DATA PROTECTION IN PERSONALIZED AI SERVICES

A DECENTRALIZED APPROACH

Vom Fachbereich Informatik der
Technischen Universität Darmstadt genehmigte

DISSERTATION

zur Erlangung des akademischen Grades
Doktor-Ingenieur (*Dr.-Ing.*)

Eingereicht von:

Christian Meurisch, M.Sc.
(*geboren am 17.09.1986 in Zell/Mosel*)

Erstreferent: Prof. Dr. Max Mühlhäuser (TU Darmstadt)
Koreferent: Prof. Dr. Schahram Dustdar (TU Wien)

Tag der Einreichung: 30.06.2021
Tag der Disputation: 23.08.2021

Fachgebiet Telekooperation
Fachbereich Informatik
Technische Universität Darmstadt
Hochschulkennziffer D 17

Darmstadt 2021

Christian Meurisch: *Data Protection in Personalized AI Services: A Decentralized Approach*
Darmstadt, Technische Universität Darmstadt
Tag der Einreichung: 30.06.2021
Tag der Disputation: 23.08.2021

URN: urn:nbn:de:tuda-tuprints-193559
URI: https://tuprints.ulb.tu-darmstadt.de/id/eprint/19355
Jahr der Veröffentlichung der Dissertation auf TUprints: 2021

In the middle of difficulty lies opportunity.

Albert Einstein

Dedicated to my (growing) family, for showing me
what is important in life and being always there for me.
You made all this possible – ***thank you.***

ABSTRACT

Advances in Artificial Intelligence (AI) have shaped today's user services, enabling enhanced personalization and new kinds of support. As such AI-based services – referred to as *AI services* in this thesis – necessarily involve (potentially sensitive) user data, the resulting privacy implications are de facto the unacceptable face of this technology: data once provided, e.g., to AI services typically running in the provider's cloud or on (third-party) edge devices, may be used for other (often commercial) purposes than originally intended, even without the user's consent or awareness. While approaches to data protection are manifold, each of them makes a certain tradeoff between personalization, privacy, and applicability – there is no practical one-size-fits-all solution.

This thesis explores a *data decentralization* approach in the context of personalized (single-user) AI services to achieve a more favorable tradeoff for users while considering the providers' interests. As a result, this work comprises *seven* (7) *major contributions*, two for the systematic understanding of data protection and privacy requirements in AI services, and five technical contributions – of the latter, three contribute protection mechanisms based on data decentralization and two pave the way for a decentralized (urban) operation. Specifically, the *first* contribution presents a user study that explores user expectations of such data-demanding AI services and the extent to which privacy concerns arise. Based on these findings, the *second* contribution classifies the related work of data protection in AI services in a novel way, highlighting the identified research gaps – some of which are addressed in this thesis, as outlined below.

While data decentralization promises users more control over their own data, it entails issues related to both efficiency and the protection of the provider's intellectual property due to the need for locally running AI services; this part of the thesis contributes three building blocks to address these issues: the *third* contribution of this thesis comprises a privacy-by-design platform, which relies on an open architecture and decentralized data-confining personal data stores with design and runtime support for AI services running locally to access user data; it forms the basis for the following building blocks. The *fourth* contribution adds a building block to ensure confidential processing of user data locally by AI services while protecting providers' intellectual property, even when both are offloaded to untrusted (third-party) edge devices. The *fifth* contribution adds a building block to address the cold-start problem and efficiency issues (e.g., caused by labeling effort for users, local resource use) specifically of AI services relying on supervised learning algorithms in local personalization.

To support mobile users in coping with resource-intensive, latency-demanding AI services and provide ambient support to them not only at home, the last part of this thesis enables a city-wide, decentralized operation of this platform. The

sixth contribution presents two economic (edge computing) infrastructure concepts, which propose to exploit existing (but originally for other purposes used) infrastructures that are predestined for this: one is based on publicly-owned augmented street lamps; the other relies on a sharing concept of privately-owned wireless home routers and their LAN-connected home resources. The *seventh* and last contribution adds a proactive deployment mechanism to efficiently conceal the inherent initialization overhead of (personalized, data-protected) AI services on nearby edge devices for mobile users.

A series of evaluations on sample AI services provides the proof of the proposed concepts—confirming the achieved *unique* tradeoff between personalization, privacy, and applicability.

ZUSAMMENFASSUNG

Fortschritte in der Künstlichen Intelligenz (KI) haben die heutigen Nutzerdienste geprägt, wodurch eine verbesserte Personalisierung dieser Dienste und neue Unterstützungsformen ermöglicht wurden. Da solche KI-basierten Dienste – in dieser Arbeit kurz als *KI-Dienste* (engl. *AI services*) bezeichnet – notwendigerweise (teils sensible) Nutzerdaten erfordern, sind die daraus resultierenden Auswirkungen auf die Privatsphäre de facto die Kehrseite der Medaille: Einmal zur Verfügung gestellte Daten, z.B. für KI-Dienste, die in der Cloud des Anbieters oder auf nahegelegenen Geräten von Drittanbietern laufen, können für andere (oft kommerzielle) Zwecke als die ursprünglich beabsichtigten Zwecke verwendet werden – auch ohne die Zustimmung oder Kenntnis des Nutzers. Obwohl bestehende Ansätze zum Schutz der Nutzerdaten vielfältig sind, geht jeder von ihnen einen gewissen Kompromiss zwischen der Personalisierung der Nutzerdienste, der Privatsphäre der Nutzer und seiner praktischen Anwendbarkeit ein – zurzeit existiert keine praktisch-einsetzbare Lösung, die diese Aspekte gleichermaßen und vollumfänglich berücksichtigt.

Diese Arbeit erforscht einen *Daten-Dezentralisierungs*-Ansatz im Kontext personalisierter KI-Dienste, um einen für den Nutzer vorteilhafteren Kompromiss zu erzielen – bei gleichzeitiger Berücksichtigung der Anbieterinteressen. Insgesamt umfasst diese Arbeit *sieben* (*7*) *Hauptbeiträge*: Zwei tragen zum systematischen Verständnis der Anforderungen an den Schutz der Privatsphäre und Nutzererwartungen an KI-Dienste bei; die restlichen fünf sind technische Beiträge, von denen drei neuartige Schutzmechanismen und zwei Konzepte für einen dezentralen (stadtweiten) Betrieb darstellen. Der *erste* Beitrag stellt eine Nutzerstudie vor, die zunächst untersucht, welche Erwartungen die Nutzer an solche personalisierten KI-Dienste haben und in welchem Ausmaß Bedenken hinsichtlich ihrer Privatsphäre bestehen. Basierend auf diesen Erkenntnissen klassifiziert der *zweite* Beitrag zunächst die relevanten verwandten Arbeiten in einer neuartigen Weise und identifiziert bestehende Forschungslücken – von denen einige in dieser Arbeit, wie nachfolgend näher erläutert, adressiert werden.

Während die Daten-Dezentralisierung den Nutzern mehr Kontrolle über ihre eigenen Daten verspricht, bringt sie aufgrund der Notwendigkeit lokal laufender KI-Dienste Probleme sowohl in Bezug auf die Effizienz als auch den Schutz des geistigen Eigentums des Anbieters mit sich. Dieser Teil der Arbeit steuert drei Bausteine bei, um diese Probleme zu adressieren: Der *dritte* Beitrag dieser Arbeit umfasst eine *Privacy by Design*-Plattform, die auf einer offenen Architektur und lokalen „daten-einsperrenden“ Datenspeicher basiert; diese bietet zudem eine Design- und Laufzeitunterstützung für lokal laufende KI-Dienste an und bildet somit die Grundlage für die folgenden Bausteine. Der *vierte* Beitrag trägt einen Baustein für die vertrauliche, lokale Verarbeitung von Nutzerdaten durch KI-Dienste bei, wobei insbesondere das geistige Eigentum der Anbieter

– auch auf nicht vertrauenswürdigen Geräten von Drittanbietern – geschützt wird. Der *fünfte* Beitrag trägt einen Baustein bei, um das Kaltstartproblem und die Effizienzprobleme speziell von KI-Diensten, die auf überwachte Lernalgorithmen bei der lokalen Personalisierung aufbauen, zu adressieren.

Um nicht nur von einer (Umgebungs-)Unterstützung zu Hause zu profitieren, ermöglicht der letzte Teil dieser Arbeit einen stadtweiten, dezentralen Betrieb dieser Plattform: Der *sechste* Beitrag schlägt zwei kosteneffiziente (*Edge Computing*-)Infrastrukturkonzepte vor, die bereits existierende (aber zweckfremde) Infrastrukturen ausnutzen: das eine Konzept basiert auf Straßenlampen in öffentlichem Besitz; das andere nutzt ein „Sharing"-Konzept, um privat-betriebene drahtlose Heimrouter und daran angeschlossene lokale Ressourcen für andere Nutzer nutzbar zu machen. Der *siebte* und letzte Beitrag trägt einen proaktiven, effizienten „Deployment"-Mechanismus bei, um den inhärenten Initialisierungsaufwand von (personalisierten, datenschützenden) KI-Diensten auf nahegelegenen Geräten für mobile Benutzer zu verbergen.

Eine Reihe von Evaluationen auf exemplarischen KI-Diensten zeigt die Machbarkeit aller Beiträge und bestätigt zugleich den erreichten *einzigartigen* Kompromiss zwischen den drei Aspekten Personalisierung, Privatsphärenschutz und Anwendbarkeit.

PUBLICATIONS

The author has already published substantial parts of this thesis in the following **48 publications** (*46 peer-reviewed*, *2 pre-published*), 32 of them as leading author.

PUBLICATIONS DIRECTLY RELATED TO THIS THESIS

These contributions form the foundation of this thesis; parts of the content of the respective publications are used verbatim (marked in *gray* in the text).

[Meu21a] Christian Meurisch. *The Trusted Edge*. 2021. arXiv: 2105.13601.

[MM21] Christian Meurisch and Max Mühlhäuser. "Data Protection in AI Services: A Survey." In: *ACM Computing Surveys* 54.2 (Mar. 2021). **JCR-IF (2019): 7.990**, 40:1–40:38. DOI: 10.1145/3440754.

[Meu+20a] Christian Meurisch, Cristina Mihale-Wilson, Adrian Hawlitschek, Florian Giger, Florian Müller, Oliver Hinz, and Max Mühlhäuser. "Exploring User Expectations of Proactive AI Systems." In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*. IMWUT 4.4 (Dec. 2020), 146:1–146:22. DOI: 10.1145/3432193.

[Meu+20b] Christian Meurisch, Bekir Bayrak, Florian Giger, and Max Mühlhäuser. "PDSProxy: Trusted IoT Proxies for Confidential Personalization of AI Services." In: *Proceedings of the 29th International Conference on Computer Communications and Networks*. ICCCN'20. IEEE, Aug. 2020, pp. 1–2. DOI: 10.1109/ICCCN49398.2020.9209655.

[Meu+20c] Christian Meurisch, Dennis Werner, Bekir Bayrak, Florian Giger, and Max Mühlhäuser. "PDSProxy++: Proactive Proxy Deployment for Confidential Personalization of AI Services." In: *Proceedings of the 29th International Conference on Computer Communications and Networks*. ICCCN'20. IEEE, Aug. 2020, pp. 1–9. DOI: 10.1109/ICCCN49398.2020.9209747.

[Müh+20] Max Mühlhäuser[1], Christian Meurisch[1], Michael Stein, Jörg Daubert, Julius von Willich, Jan Riemann, and Lin Wang. "Street Lamps as a Platform." In: *Communications of the ACM* 63.6 (June 2020). **JCR-IF (2019): 6.988**, pp. 56–64. DOI: 10.1145/3376900.

[MBM20] Christian Meurisch, Bekir Bayrak, and Max Mühlhäuser. "Privacy-preserving AI Services Through Data Decentralization." In: *Proceedings of The Web Conference 2020*. WWW'20. **acceptance rate: 19.2%**. ACM, Apr. 2020, pp. 190–200. DOI: 10.1145/3366423.3380106.

[1]Co-first authors: these authors have contributed equally to this work.

[MBM19a] Christian Meurisch, Bekir Bayrak, and Max Mühlhäuser. "EdgeBox: Confidential Ad-hoc Personalization of Nearby IoT Applications." In: *Proceedings of the 2019 IEEE Global Communications Conference*. GLOBECOM'19. IEEE, Dec. 2019. DOI: 10.1109/GLOBECOM38437.2019.9013520.

[Meu+19b] Christian Meurisch, Sebastian Kauschke, Tim Grube, Bekir Bayrak, and Max Mühlhäuser. "{P}Net: Privacy-Preserving Personalization of AI-Based Models by Anonymous Inter-Person Similarity Networks." In: *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. MobiQuitous'19. ACM, Nov. 2019, pp. 60–69. DOI: 10.1145/3360774.3360819.

[MBM19b] Christian Meurisch, Bekir Bayrak, and Max Mühlhäuser. "AssistantGraph: An Approach for Reusable and Composable Data-driven Assistant Components." In: *Proceedings of the 2019 IEEE 43rd Annual Computer Software and Applications Conference*. COMPSAC'19. IEEE, July 2019, pp. 513–522. DOI: 10.1109/COMPSAC.2019.00079.

[Meu+17a] Christian Meurisch, Maria-Dorina Ionescu, Benedikt Schmidt, and Max Mühlhäuser. "Reference Model of Next-generation Digital Personal Assistant: Integrating Proactive Behavior." In: *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers*. UbiComp'17. ACM. 2017, pp. 149–152. DOI: 10.1145/3123024.3123145.

[Meu+17b] Christian Meurisch, Julien Gedeon, Artur Gogel, The An Binh Nguyen, Fabian Kaup, Florian Kohnhäuser, Lars Baumgärtner, Milan Schmittner, and Max Mühlhäuser. "Temporal Coverage Analysis of Router-based Cloudlets Using Human Mobility Patterns." In: *Proceedings of the 2017 IEEE Global Communications Conference*. GLOBECOM'17. IEEE, Dec. 2017, pp. 1–6. DOI: 10.1109/GLOCOM.2017.8255035.

[Meu+17e] Christian Meurisch, Julien Gedeon, The An Binh Nguyen, Fabian Kaup, and Max Mühlhäuser. "Decision Support for Computational Offloading by Probing Unknown Services." In: *Proceedings of the 2017 26th International Conference on Computer Communication and Networks*. ICCCN'17. IEEE, July 2017, pp. 1–9. DOI: 10.1109/ICCCN.2017.8038406.

[Meu+17f] Christian Meurisch, Bennet Jeutter, Wladimir Schmidt, Nickolas Gündling, Benedikt Schmidt, Fabian Herrlich, and Max Mühlhäuser. "An Extensible Pervasive Platform for Large-scale Anticipatory Mobile Computing." In: *Proceedings of the 2017 IEEE 41st Annual Computer Software and Applications Conference*. COMPSAC'17. IEEE, July 2017, pp. 459–464. DOI: 10.1109/COMPSAC.2017.54.

[Meu+17i] Christian Meurisch, The An Binh Nguyen, Stefan Wullkotte, Stefan Niemczyk, Florian Kohnhäuser, and Max Mühlhäuser. "NICER911: Ad-hoc Communication and Emergency Services Using Networking Smartphones and Wireless Home Routers." In: *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. MobiHoc'17. (**Best Poster Award**). ACM, July 2017, 33:1–33:2. DOI: 10.1145/3084041.3084075.

[Meu+15b] Christian Meurisch, Alexander Seeliger, Benedikt Schmidt, Immanuel Schweizer, Fabian Kaup, and Max Mühlhäuser. "Upgrading Wireless Home Routers for Enabling Large-scale Deployment of Cloudlets." In: *Proceedings of the 7th International Conference on Mobile Computing, Applications, and Services*. MobiCASE'15. (**Best Paper Candidate**). Springer, Nov. 2015, pp. 12–29. DOI: 10.1007/978-3-319-29003-4_2.

OTHER PUBLICATIONS

The author has also (co-)authored several other publications, which are situated in related research areas and complement the contributions listed above.

[BGM20] Bekir Bayrak, Florian Giger, and Christian Meurisch. *Insightful Assistant: AI-compatible Operation Graph Representations for Enhancing Industrial Conversation Agents*. Sept. 2020. arXiv: 2007.12929 [cs.CL].

[Meu+19a] Christian Meurisch, Artur Gogel, Bekir Bayrak, and Max Mühlhäuser. "NextAct: A Hybrid Approach for High-Resolution Human Activity Predictions." In: *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. MobiQuitous'19. ACM, Nov. 2019, pp. 278–287. DOI: 10.1145/3360774.3360821.

[Raw+19] Reza Rawassizadeh, Taylan Sen, Sunny Jung Kim, Christian Meurisch, Hamidreza Keshavarz, Max Mühlhäuser, and Michael Pazzani. "Manifestation of Virtual Assistants and Social Robots into Daily Life: Vision and Challenges." In: *CCF Transactions on Pervasive Computing and Interaction* (Oct. 2019), pp. 1–12. DOI: 10.1007/s42486-019-00014-1.

[Gui+19] Alejandro Sanchez Guinea, Usman Naeem, Philipp M. Scholl, Elena Di Lascio, Pei-Yi (Patricia) Kuo, Veljko Pejovic, Alexander Seeliger, Cristina Mihale-Wilson, Muhammad Awais Azam, Max Mühlhäuser, and Christian Meurisch. "UPA'19: 4th International Workshop on Ubiquitous Personal Assistance." In: *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers*. UbiComp/ISWC'19. ACM, Sept. 2019, pp. 1099–1101. DOI: 10.1145/3341162.3347755.

[Meu+19c] Christian Meurisch, Zain Hamza, Bekir Bayrak, and Max Mühlhäuser. "Enhanced Detection of Crisis-related Microblogs by Spatiotemporal Feedback Loops." In: *Proceedings of the 2019 IEEE 43rd Annual Computer Software and Applications Conference*. COMPSAC'19. IEEE, July 2019, pp. 507–512. DOI: 10.1109/COMPSAC.2019.00078.

[Meu+18a] Christian Meurisch, Lulzim Murati, Rüdiger Eichin, Benedikt Schmidt, and Max Mühlhäuser. "Tour Guides Get Guided: Intelligent Coordination of Simultaneous Tours in Exhibition Environments." In: *Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. MobiQuitous'18. ACM, Nov. 2018, pp. 274–283. DOI: 10.1145/3286978.3286987.

[Ged+18a] Julien Gedeon, Michael Stein, Jeff Krisztinkovics, Patrick Felka, Katharina Keller, Christian Meurisch, Lin Wang, and Max Mühlhäuser. "From Cell Towers to Smart Street Lamps: Placing Cloudlets on Existing Urban Infrastructures." In: *Proceedings of the 2018 IEEE/ACM Symposium on Edge Computing*. SEC'18. Oct. 2018, pp. 187–202. DOI: 10.1109/SEC.2018.00021.

[Meu+18b] Christian Meurisch, Philipp M. Scholl, Usman Naeem, Veljko Pejovic, Florian Müller, Elena Di Lascio, Pei-Yi (Patricia) Kuo, Sebastian Kauschke, Muhammad Awais Azam, and Max Mühlhäuser. "UPA'18: 3rd International Workshop on Ubiquitous Personal Assistance." In: *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*. UbiComp'18. ACM, Oct. 2018, pp. 766–769. DOI: 10.1145/3267305.3274133.

[Rie+18] Jan Riemann, Markus Funk, Martin Schmitz, Christian Meurisch, and Max Mühlhäuser. "OverTop: Breaking the Boundaries of Tangible Tabletop Environments." In: *Adjunct Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*. UbiComp'18. ACM, Sept. 2018, pp. 231–234. DOI: 10.1145/3267305.3267559.

[Ged+18b] Julien Gedeon, Jeff Krisztinkovics, Christian Meurisch, Michael Stein, Lin Wang, and Max Mühlhäuser. "A Multi-Cloudlet Infrastructure for Future Smart Cities: An Empirical Case Study." In: *Proceedings of the 1st International Workshop on Edge Systems, Analytics and Networking*. EdgeSys'18. IEEE, June 2018, pp. 19–24. DOI: 10.1145/3213344.3213348.

[Ngu+18] The An Binh Nguyen, Marius Rettberg-Päplow, Christian Meurisch, Tobias Meuser, Björn Richerzhagen, and Ralf Steinmetz. "Complex Services Offloading in Opportunistic Networks." In: *Proceedings of the 2018 IFIP Networking Conference*. Networking'18. IEEE, May 2018, pp. 1–9. DOI: 10.23919/IFIPNetworking.2018.8696915.

[Kau+18] Fabian Kaup, Stefan Hacker, Eike Mentzendorff, Christian Meurisch, and David Hausheer. "Energy Models for NFV and Service Provisioning on Fog Nodes." In: *Proceedings of the 2018 IEEE/IFIP Network Operations and Management Symposium*. NOMS'18. IEEE, Apr. 2018, pp. 1–7. DOI: 10.1109/NOMS.2018.8406158.

[Hus+17] Rida Ghafoor Hussain, Muhammad Awais Azam, Mustansar Ali Ghazanfar, Usman Naeem, and Christian Meurisch. "Smartphone-based Robust Hierarchical Framework for Activity Recognition based on Machine Learning." In: *Proceedings of the 2017 Future Technologies Conference*. FTC'17. The Science and Information (SAI) Organization, Nov. 2017, pp. 1–5. URL: https://saiconference.com/Conferences/FTC2017Proceedings.

[Meu+17c] Christian Meurisch, Artur Gogel, Benedikt Schmidt, Timo Nolle, Frederik Janssen, Immanuel Schweizer, and Max Mühlhäuser. "Capturing Daily Student Life by Recognizing Complex Activities Using Smartphones." In: *Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. MobiQuitous'17. ACM, Nov. 2017, pp. 156–165. DOI: 10.1145/3144457.3144472.

[Bau+17] Lars Baumgärtner, Stefan Kohlbrecher, Juliane Euler, Tobias Ritter, Milan Stute, Christian Meurisch, Max Mühlhäuser, Matthias Hollick, Oskar von Stryk, and Bernd Freisleben. "Emergency Communication in Challenged Environments via Unmanned Ground and Aerial Vehicles." In: *Proceedings of the 2017 IEEE Global Humanitarian Technology Conference*. GHTC'17. IEEE, Oct. 2017, pp. 1–9. DOI: 10.1109/GHTC.2017.8239244.

[Ngu+17a] The An Binh Nguyen, Pratyush Agnihotri, Christian Meurisch, Manisha Luthra, Rahul Dwarakanath, Jeremias Blendin, Doreen Böhnstedt, Michael Zink, and Ralf Steinmetz. "Efficient Crowd Sensing Task Distribution Through Context-aware NDN-based Geocast." In: *Proceedings of the 2017 IEEE 42nd Conference on Local Computer Networks*. LCN'17. IEEE, Oct. 2017, pp. 52–60. DOI: 10.1109/LCN.2017.73.

[Meu+17d] Christian Meurisch, Usman Naeem, Philipp Scholl, Muhammad Awais Azam, Sebastian Günther, Paul Baumann, Shafiq ur Réhman, and Max Mühlhäuser. "SmartGuidance'17: 2nd Workshop on Intelligent Personal Support of Human Behavior." In: *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers*. UbiComp'17. ACM, Sept. 2017, pp. 623–626. DOI: 10.1145/3123024.3124457.

[Meu+17g] Christian Meurisch, The An Binh Nguyen, Julien Gedeon, Florian Kohnhäuser, Milan Schmittner, Stefan Niemczyk, Stefan Wullkotte, and Max Mühlhäuser. "Upgrading Wireless Home Routers as Emergency Cloudlet and Secure DTN Communication Bridge."

In: *Proceedings of the 2017 26th International Conference on Computer Communication and Networks*. ICCCN'17. IEEE, July 2017, pp. 1–2. DOI: 10.1109/ICCCN.2017.8038485.

[Meu+17h] Christian Meurisch, The An Binh Nguyen, Martin Kromm, Andrea Ortiz, Ragnar Mogk, and Max Mühlhäuser. "DisVis 2.0: Decision Support for Rescue Missions Using Predictive Disaster Simulations with Human-Centric Models." In: *Proceedings of the 2017 26th International Conference on Computer Communication and Networks*. ICCCN'17. IEEE, July 2017, pp. 1–2. DOI: 10.1109/ICCCN.2017.8038474.

[Ged+17] Julien Gedeon, Christian Meurisch, Disha Bhat, Michael Stein, Lin Wang, and Max Mühlhäuser. "Router-based Brokering for Surrogate Discovery in Edge Computing." In: *Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops*. ICDCSW'17. IEEE, June 2017, pp. 145–150. DOI: 10.1109/ICDCSW.2017.61.

[Ngu+17b] The An Binh Nguyen, Christian Meurisch, Stefan Niemczyk, Doreen Böhnstedt, Kurt Geihs, Max Mühlhäuser, and Ralf Steinmetz. "Adaptive Task-Oriented Message Template for In-Network Processing." In: *Proceedings of the 2017 International Conference on Networked Systems*. NetSys'17. IEEE, Mar. 2017, pp. 1–8. DOI: 10.1109/NetSys.2017.7903952.

[Ngu+17c] The An Binh Nguyen, Christian Meurisch, Stefan Niemczyk, Christian Klos, Doreen Böhnstedt, and Ralf Steinmetz. "Facilitating volunteer computing resources for in-network processing through message template." In: *Proceedings of the 2017 International Conference on Networked Systems*. NetSys'17. IEEE, Mar. 2017, pp. 1–2. DOI: 10.1109/NetSys.2017.7931515.

[Meu16] Christian Meurisch. "Intelligent Personal Guidance of Human Behavior Utilizing Anticipatory Models." In: *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*. UbiComp'16. ACM, Sept. 2016, pp. 441–445. DOI: 10.1145/2968219.2971355.

[Meu+16] Christian Meurisch, Usman Naeem, Muhammad Awais Azam, Frederik Janssen, Benedikt Schmidt, and Max Mühlhäuser. "Smarticipation: First Workshop on Intelligent Personal Guidance of Human Behavior Utilizing Anticipatory Models." In: *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*. UbiComp'16. ACM, Sept. 2016, pp. 1227–1230. DOI: 10.1145/2968219.2968436.

[Meu+15a] Christian Meurisch, Tahir Hussain, Artur Gogel, Benedikt Schmidt, Immanuel Schweizer, and Max Mühlhäuser. "A Spatiotemporal Approach for Social Situation Recognition." In: *Proceedings of the 7th International Conference on Mobile Computing, Applications, and Services: Poster*. Vol. 162. MobiCASE'15. Springer, Nov. 2015, pp. 309–

316. ISBN: 978-3-319-29003-4. DOI: 10.1007/978-3-319-29003-4_18.

[Meu+15c] Christian Meurisch, Ashwinkumar Yakkundimath, Benedikt Schmidt, and Max Mühlhäuser. "Upgrading Wireless Home Routers as Emergency Cloudlet: A Runtime Measurement." In: *Proceedings of the 7th International Conference on Mobile Computing, Applications, and Services: Poster*. Vol. 162. MobiCASE'15. Springer, Nov. 2015, pp. 338–340. ISBN: 978-3-319-29003-4. DOI: 10.1007/978-3-319-29003-4.

[Meu+15d] Christian Meurisch, Benedikt Schmidt, Michael Scholz, Immanuel Schweizer, and Max Mühlhäuser. "Labels: Quantified Self App for Human Activity Sensing." In: *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers*. UbiComp/ISWC'15. ACM, Sept. 2015, pp. 1413–1422. DOI: 10.1145/2800835.2801612.

[Sch+15b] Benedikt Schmidt, Sebastian Benchea, Rüdiger Eichin, and Christian Meurisch. "Fitness Tracker or Digital Personal Coach: How to Personalize Training." In: *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers*. UbiComp/ISWC'15 Adjunct. ACM, Sept. 2015, pp. 1063–1067. DOI: 10.1145/2800835.2800961.

[Meu14] Christian Meurisch. "Social Hub: Multi-source Fusion of the Personal Social Graph." MSc. thesis. TU Darmstadt, Aug. 2014. URL: http://tubiblio.ulb.tu-darmstadt.de/116926/.

[Meu+13] Christian Meurisch, Karsten Planz, Daniel Schäfer, and Immanuel Schweizer. "Noisemap – Discussing Scalability in Participatory Sensing." In: *Proceedings of the First International Workshop on Sensing and Big Data Mining*. SenseMine'13. ACM, 2013, pp. 1–6. DOI: 10.1145/2536714.2536720.

[Meu12] Christian Meurisch. "Non-monetary Incentive Systems for Mobile Participatory Sensing Apps." BSc. thesis. TU Darmstadt, 2012. URL: http://tubiblio.ulb.tu-darmstadt.de/116925/.

[Sch+12] Immanuel Schweizer, Christian Meurisch, Julien Gedeon, Roman Bärtl, and Max Mühlhäuser. "Noisemap: Multi-tier Incentive Mechanisms for Participative Urban Sensing." In: *Proceedings of the Third International Workshop on Sensing Applications on Mobile Phones*. PhoneSense'12. ACM, Nov. 2012, 9:1–9:5. DOI: 10.1145/2389148.2389157.

Feeling gratitude and not expressing it
is like wrapping a present and not giving it.

William A. Ward

ACKNOWLEDGMENTS

This thesis would not have been possible without the wonderful support of so many awesome people including my advisors, colleagues, students, friends, and my loving family. Many thanks to all of you named below.

First of all, I would like to express my deepest gratitude to *Max*, my doctoral supervisor and mentor, for his continuous and outstanding support: thank you for believing in me, influencing me strongly in my personal development, and giving me the opportunity to write this thesis. I would also like to thank *Prof. Schahram Dustdar* (TU Wien, Austria) for acting as a co-referee for my work.

Over the past years, I had the pleasure of being part of a wonderful research group – many thanks to *all of you*. Of course, a special thanks goes to my awesome colleagues of the ISY (formerly SPA) group, *Alex*, *Timo*, and *Seb*, for the countless hours we spent on new ideas, presentations, and general discussions. Without a postdoc for long stretches, we've made it (in a self-organized, mutually supporting and motivating manner) and achieved some impressive results after all. I also want to expressly thank my former office mates, *Jan*, *Sepp*, and *Mo*, with whom one can always have fun. I also really enjoyed the (personal and 'grapevine') conversations with *Martin S.*, *Tim*, *Michael*, and especially *Rolf*, who – even though he is not that aware of it – opened my eyes for a new mindset that has mostly influenced my life to this day. Many thanks also go to my first supporters, *Benedikt*, *Immanuel*, and *Flo V.*, for their encouragement and some good advice—many of them, unfortunately, I only really grasped in retrospect.

Likewise, I appreciate *all my students* for their great work. This applies especially to my 'precious' students, *Bekir*, *Adrian*, and *Bennet*: thanks for all the productive discussions about my research we had and for your incredible inspirations. You've got it, guys – keep it up! I also enjoyed working with *Stefan W.*, *Flo G.*, *Dennis W.*, *Artur*, *Maria*, *Simone*, *Wladimir*, and *Nicko*. I would also like to thank the entire *NICER team* for the good cooperation and wonderful experience as well as other external researchers, such as *Binh* and *Fabian K.*, who influenced my work.

Finally, I would like to express my sincere gratitude to my girlfriend *Micha*; my parents, *Roland* and *Christa*; my sister *Susanne* and her family (especially my always good-humored godson *Mori*); and my *friends*: thank you all so much for your continuous support and encouragement, infinite patience (especially for enduring my changing moods during stressful times), and above all, that you never stopped believing in me and that you enrich my life in so many ways.

CONTENTS

Part I

INTRODUCTION

1 INTRODUCTION

1.1 MOTIVATION

Around three decades after the invention of the World Wide Web (WWW), commonly known as *the web*, its founder Tim Berners-Lee reflects on how it has evolved. In many ways, the web has lived up his initial vision of "an open platform that would allow everyone, everywhere to share information, access opportunities and collaborate across geographic and cultural boundaries". As one point of criticism, on the other hand, Berners-Lee is particularly concerned that "we've lost control of our personal data". [Ber17]

> *The current business model for many websites offers free content in exchange for personal data. Many of us agree to this – [...] fundamentally we do not mind some information being collected in exchange for free services. But, we're missing a trick. As our data is then held in proprietary silos, out of sight to us, we lose out on the benefits we could realize if we had direct control over this data, and chose when and with whom to share it. What's more, we often do not have any way of feeding back to companies what data we'd rather not share - especially with third parties [...]. This widespread data collection by companies also has other impacts. Through collaboration with - or coercion of - companies, governments are also increasingly watching our every move online, and passing extreme laws that trample on our rights to privacy.*
>
> Tim Berners-Lee, 2017 [Ber17]

1.1.1 *Privacy: The Loss of Data Ownership*

The *loss of control over the own data*, to which Berners-Lee alludes in his review from 2017, is – besides targeted attacks (e.g., 2011 Sony PlayStation Network outage [QA11]) and accidental bugs (e.g., 2018 Amazon Alexa eavesdropping scandal [Lyn19]) – mainly caused by the way web-based services work today [Pap+17]: if *users* want to benefit from such services, they must share their data (referred to as *user data* or *personal data*) with the *service providers*, in particular to personalize or tailor these services to their tastes (referred to as *personalized services*) [CS05; Xu+11]. As services typically run in the cloud of the respective provider [Wie+17], potentially-sensitive data leaves the (metaphorically speaking) *user territory*—the digital space that is fully-controlled by the user and cannot be bypassed or restricted by the provider. As further denounced by Berners-Lee, and what represents the key problem, data once provided (or just a copy of it) may be used without the consent or awareness of the users for other purposes (e.g., for personalized advertising to finance free services) than those initially intended—leading to serious *privacy invasions* or, at worst, to personal data breaches such as the 2018 Facebook–Cambridge Analytica incident [Lan19].

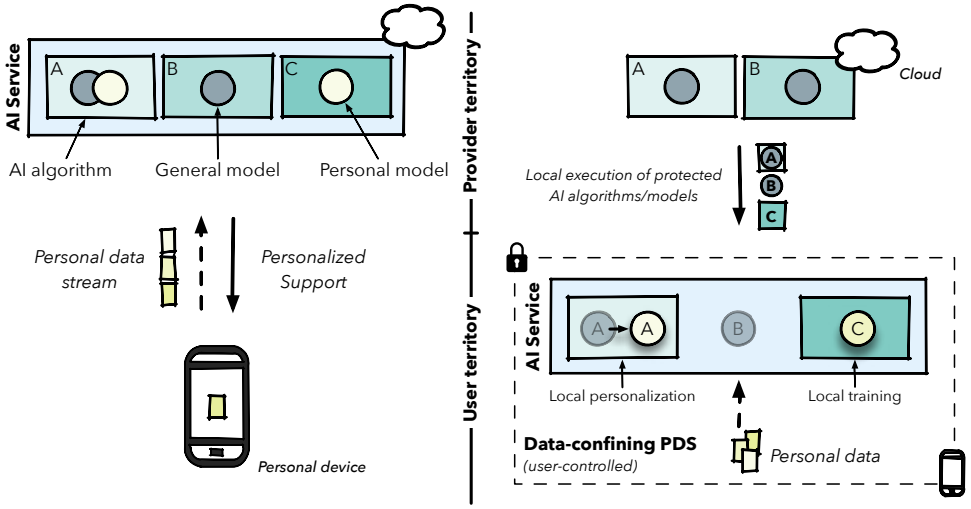


Figure 1.1: Schematic representation of a centralized AI service (*left*) and an AI service based on data decentralization (*right*). An AI service (*blue*) may consist of several possibly-protected general (*gray circle*) and personal AI models (*yellow circle*), which are trained by specific possibly-protected AI algorithms (*green*) and may require personal data (*yellow rectangles*).

1.1.2 *Personalization: Advances Through AI Services*

With the advance in disciplines such as *intelligence augmentation* (including *anticipatory computing* [PM15]) and *intelligent infrastructures* (including *edge AI* [Zho+19; An+19]), such privacy issues are further aggravated [Jor19]. Services in the former discipline increasingly base their actions on Artificial Intelligence (AI) models that `(N1)` demand an extensive and constant stream of personal data to enable enhanced personalization and better user support, e.g., through proactive intervention [Sar17] – we generally refer to them as *AI services* (see Figure 1.1, *left*, the details will be explained below). Services in the latter discipline, relying on "a web of computation, data, and physical entities" [Jor19], `(N2)` demand running (at least parts of the) AI services on nearby Internet of Things (IoT)/edge devices, e.g., to make human environments more responsive and supportive [Rag15]. We generally refer to this subset as *distributed AI services* and to the subset of distributed AI services that require a specific IoT device in the user's proximity (e.g., to access its built-in or connected sensors/actuators for providing ambient support) as *device-bound AI services*. Both general demands `(N1+2)` of such emerging AI services make privacy and data protection increasingly challenging.

1.1.3 *Tradeoffs Between Privacy and Personalization*

Approaches to protecting user data are manifold, but – besides all their advantages – each of them shows individual drawbacks with respect to `N1+2`, making a certain tradeoff between personalization, privacy, and applicability. More specifically, conventional approaches use protocols, access control mechanisms, and policies for a privacy-respecting data management [Lan02]. However, to enable

personalized AI services, either user data or the provider's *protected AI algorithms/models* must be shared to some extent out of the user territory in the former case or, in the latter case, out of the *provider territory*—analogously referring to the digital space in which the provider's confidential parts of AI services should run to ensure Intellectual Property (IP) protection. Crypto-based approaches, e.g., relying on homomorphic encryption (HE) or secure multiparty computation (SMC), can share and process personal data in encrypted format [Dow+16; Lin05]. For instance, collaborative or federated learning (with secure aggregation) can train a shared *general models* across multiple users, without sharing sensitive personal data [Har+19; Bon+17]. However, these approaches are limited either in their applicability or in the effectiveness of the resulting (general) model for some of the users. Perturbation and randomization-based approaches add noise to the data [GL08; Aba+16], which limits their usefulness for training *personal models*. Approaches involving *local training* of personal models or *local personalization* of a general model lead to issues in both efficiency and the protection of the provider's IP due to their local execution [Ser+18; KLF19]. All in all, there exists *no* practical one-size-fits-all solution that covers all the above data protection issues and at the same time meeting both demands `(N1+2)` of evolving personalized AI services. [Meu21b]

1.2 TOWARDS TRUE DATA OWNERSHIP IN PERSONALIZED AI SERVICES

To overcome these privacy issues at all, it is required that users must *always* "own their own data", including copies and derivatives of it (e.g., analysis results of AI services based on user data). Inspired by Common Law, Pentland defines *data ownership* as "the rights of possession, use, and disposal". Data ownership, in particular, implies full control over the use of the own data, i.e., data owners can distribute their data, restrict its use, and remove it at any time. [Pen09]

1.2.1 *Data Decentralization as a Solid Basis*

As a promising direction to technically achieve data ownership in personalized AI services, *data decentralization* concepts form a suitable basis. Specifically, blockchain-based approaches can ensure the immutability, integrity, and validity of data without trusting a central authority. Such approaches, for instance, enable decentralized access control and audit management for personal data [YGR17; Cho+18]. However, the genuine Blockchain concept is generally designed to make data globally accessible, which contradicts data protection – while data integrity can be guaranteed, data confidentiality cannot (e.g., data can be leaked as users interact with blockchains): all sensitive data must therefore be stored encrypted in the blockchain, as it is accessible to other peers, which in turn limits its usefulness for AI services (see crypto-based approaches).

More practical decentralization approaches are based on the following concept (see Figure 1.1, *right*, p. 2): users collect and store their digital data consistently in a single user-specified and -controlled location, their own *personal data*

store (PDS) [YGR17]. If users want to use an AI service, they will provide the service with fine-grained and audited access to their personal data—ensuring traceability, liability, and accountability [Mor+16]. In this way, the PDS concept decouples the data from the services themselves. PDS implementations are manifold, including openPDS [Mon+14], INDX [VO14], Databox [Mor+16], and Solid Pods [Sam+16], mainly differing in their access mechanisms. For example, some approaches require parts of the proprietary service (or technically speaking its code) to be executed locally to access useful personal data; only derivates and/or metadata may be shared [Mon+14]. Although current PDS approaches achieve an improved privacy through their selective sharing and audit mechanisms, they still require some degree of data or code sharing (the latter could infringe the IP rights of the service provider).

1.2.2 *A New Privacy Setting*

This thesis, in contrast, starts from the premise that 'true' data ownership can only be reliably achieved if user data (including its derivatives) *never* leaves the user territory – or at least *not* in unencrypted or reversible obfuscated form. Otherwise, it is technically almost impossible to ensure data ownership and control the flow of data. Therefore, this thesis proposes the concept of *data-confining personal data store* (DC-PDS) as a solid basis in personalized AI services. DC-PDS relies on the general concept of PDS with the premise-derived modification of *strictly enforcing the confinement of user data*. Along with it, this thesis further gives the following extended definition of PDS, newly introducing the term 'data-confining PDS' [Meu21b]:

Definition 1.2.1 (Data-confining Personal Data Store)**.** *A data-confining personal data store is a PDS that – additionally – strictly enforces the confinement of user data, which never leave the user territory in raw, unencrypted, non-anonymized or reversible obfuscated form.*

This privacy-by-design concept of DC-PDS ensures data ownership for its users per definition but (*RC-1*) its support for AI services with respect to `N1+2` remains challenging, as user data can only be accessed locally within the user territory.

1.2.3 *Inherent Research Challenges*

DC-PDS as the basis for interacting with AI services poses different challenges. In particular, they concern (i) the *fact* that at least parts of the underlying proprietary data-demanding AI services must now run locally (see Figure 1.1, p. 2, *right*), and (ii) the *extent* to which they run locally. The former issue would (*RC-2*) disclose confidential AI algorithms/models when executing them locally (outside the provider territory), and thus, infringe the IP rights of the provider; the latter issue would (*RC-3*) burden the users (e.g., a possible labeling effort) and their personal devices (e.g., resource use), leading to a cold-start problem and efficiency issues. Therefore, the overarching challenge (*RC-4*) lies in finding the best tradeoff between privacy (user data protection, provider code/IP

protection), personalization of AI services in terms of efficiency (e.g., manual labeling effort, local resource use) and effectiveness (e.g., model performance), and applicability (or generalizability) of the solutions.

To allow mobile users to benefit from such data decentralization concepts, a suitable densely-deployed (edge computing) infrastructure with high coverage is required, e.g., to offload resource-demanding AI services that overstrain mobile devices' batteries or compute power or to provide ambient support to users not limited to their homes. This leads to (*RC-5*) the classical (*application–infrastructure*) *bootstrapping problem*: without unique use cases and corresponding distributed AI services that leverage nearby resources, there is no incentive to provide appropriate infrastructures; on the other hand, without a large-enough infrastructure that provides high service coverage for mobile users, there is little incentive for developers to create those new applications or services [Sat17]. Another challenge in this context (*RC-6*) is the *ad-hoc deployment* of device-bound AI services on such decentralized infrastructures: users want to receive (ambient) support right after they reach the (IoT/edge) devices, but only within the wireless reach of the devices users can (ad hoc) deploy AI services and corresponding data protection mechanisms or personalize the AI services with user data – this may lead to a *timing issue*, which negatively affects the user experience.

1.2.4 *Contributions of This Thesis*

Based on the above-mentioned research challenges, this thesis comprises **seven (7) main contributions** for data protection in personalized AI services. In particular, two (2) of them contribute to the systematic understanding of data protection and privacy requirements in AI services, and five are technical contributions – of the latter, three (3) contribute protection mechanisms based on data decentralization in AI services, and two (2) pave the way for a decentralized (urban) operation of distributed AI services.

In detail, the *first* contribution presents a user study that explores user expectations of (personalized, data-demanding) AI services with special attention to the privacy aspect [Meu+17a; Meu+20a]. To this end, this thesis proposes a new study method that situates the participants in the likely context of use in order to get more realistic answers about users' (initial) expectations towards such AI services—leading to a novel understanding of the extent to which privacy concerns arise. Considering these findings, the *second* contribution is a novel classification of data protection research in AI services [MM21]. Specifically, the approaches considered are reviewed at three different levels, namely management, system, and AI levels. The review reveals that not all of them meet the established requirements of evolving AI services. Based on the review, this contribution also highlights open research gaps. Most notably, in the edge AI context, is the generally neglected challenge of protecting proprietary AI algorithms/ models locally or 'at the edge' [Meu21a].

To address these issues identified, this thesis adds the following integrated technical contributions [MBM20]. The *third* contribution of this thesis comprises a privacy-by-design platform (namely `PrivAI`), addressing *RC-1*; it incorporates an open architecture concept (namely `OAII`) and the *data-confining* personal data store concept (already introduced as `DC-PDS`). The former concept enables providers to connect to the platform and operate their AI services there, both in a loosely-coupled way [Meu+17f]. The latter concept also provides design and runtime support to increase the efficiency of locally-running AI services, relying on a graph-based approach (namely `openAIgraph`) [MBM19b]. All in all, `PrivAI` follows the data decentralization paradigm, forming the basis for the following contributions or building blocks. The *fourth* contribution adds the building block (namely `PDSProxy`) for confidential processing of user data while keeping the provider's AI algorithms/models confidential (IP protection) [Meu+20b]. For both, `PDSProxy` can expand the territories of the users and providers to untrusted (third-party) edge devices, addressing (*RC-2*) the three-party confidentiality challenge: neither party can disclose confidential data/code from another party; nonetheless, AI services may temporarily access user data under strict supervision, and the hosting system/device is able to run AI services without accessing the (unencrypted) code [MBM19a]. The *fifth* contribution adds the building block (namely `PNet`) to (*RC-3*) address the cold-start problem of local personalization for new users by proposing a community-based approach [Meu+19b]. Specifically, this approach enables accurate and privacy-enhancing personalization of AI services that rely on supervised learning algorithms while keeping the burden to users (in terms of labeling effort) and local devices (in terms of local resource use) lower than comparable work. All contributions together achieve (*RC-4*) a *unique* tradeoff between personalization, privacy, and applicability.

Especially distributed and device-bound AI services (as defined) require nearby (IoT/edge) devices to benefit from low-latency offloading to edge resources and to provide ambient user support, respectively. To address (*RC-5*) the introduced (application–infrastructure) bootstrapping problem for such decentralized concepts, starting with urban environments, the *sixth* contribution comprises two complementary infrastructure concepts. For economic realization, both propose to exploit existing infrastructures, which are predestined for a required city-wide (decentralized) operation: one is based on publicly-owned augmented street lamps (coined as `SLaaP`) [Müh+20]; the other one relies on a sharing concept based on privately-owned wireless home routers and possible LAN-connected home resources (coined as `RaaP`) [Meu+15b; Meu+17i; Meu+17b]. The *seventh* and last contribution adds a proactive deployment mechanism (namely `PDSProxy++`) for device-bound AI services based on user mobility. In this way, personalized yet data-protected AI services can efficiently deployed on nearby (IoT/edge) devices and initialized with user data just in time, shortly before mobile users would use them [Meu+20c]. A complementary assessment approach allows to check the target devices for their suitability in advance [Meu+17e]. Both approaches together allow to (RC-6) conceal the inherent initialization overhead from the

mobile user, which increases the user experience in ad-hoc deployment scenarios.

Finally, it is important to note that only *single-user services* (that only support individuals separately, without sharing data with other users) are taken into account. All concepts proposed are *evaluated* with implemented proof-of-concept prototypes and compared against baseline approaches, which demonstrate their feasibility and efficacy. In addition, some of the proposed concepts (e.g., the IP protection on untrusted devices or the proactive deployment/initialization of services based on the individual mobility of the users) can be directly *transferred* to other research fields such as edge computing to solve related issues. Still others are *complementary* to existing approaches like federated learning that can train a general model in a privacy-preserving way, from which the proposed approaches start their personalization process.

Remarks: All contributions of this thesis or concepts proposed have already been published at international peer-reviewed conferences or journals. Before each part or chapter, the publications that have shaped it and in which the concepts have been published are listed – a full list of the author's publications is given at pages ix ff. In agreement with all authors involved, this thesis may use parts of the content of the respective publications verbatim, which are marked in *gray* (text). This also includes other material presented in this thesis (comprising figures, tables, and algorithms), which may be based on or taken from the author's publications listed in the respective *contribution statement* of the chapter.

1.3 STRUCTURE

This thesis comprises five main parts, structured as follows:

PART I (INTRODUCTION) – which is this part – first motivated this thesis, placing it in the context of personalized AI services and defining its scope (see CHAPTER 1).

PART II (BACKGROUND & RELATED WORK) gives the reader an systematic understanding of existing approaches, research gaps identified and requirements to be considered – organized in the following two chapters:

CHAPTER 2 provides the *background information* required for this thesis; it also contributes a *user study* that explores user expectations of such data-demanding AI services and the extent of which privacy concerns arise.

CHAPTER 3 discusses and classifies the *related work* on data protection in AI services in a novel way, highlighting the identified research gaps – some of which will be addressed in the following two parts.

PART III (PROTECTION MECHANISMS THROUGH DATA DECENTRALIZATION) covers the main platform concept based on data decentralization and the newly-integrated mechanisms (representing the building blocks) for enabling data protection in personalized AI services – organized in the following three chapters:

CHAPTER 4 proposes the *core platform* that follows a privacy-by-design approach by making the assumption of a data-confining PDS and provides *design and runtime support* for locally-running AI services—building the solid and efficient foundation for the following building blocks.

CHAPTER 5 adds a building block to ensure *confidential processing* of user data locally by AI services while protecting the intellectual property of providers, even when both (user data, provider code) are offloaded to untrusted edge devices.

CHAPTER 6 adds a building block for *privacy-enhancing personalization* of locally-running AI services; it addresses in particular the cold-start problem and efficiency issues specifically of AI services relying on supervised learning algorithms in local personalization.

PART IV (DATA DECENTRALIZATION SUPPORTIVE CONCEPTS) presents concepts to pave the way for a decentralized operation of the proposed platform on suitable and ubiquitous (IoT/edge) infrastructures – organized in the following two chapters:

CHAPTER 7 presents two economic (edge computing) *infrastructure concepts*, which propose to exploit existing urban infrastructures that are predestined for this: one is based on publicly-owned augmented street lamps; the other relies on a sharing concept of privately-owned wireless home routers and their LAN-connected home resources.

CHAPTER 8 contributes a *proactive deployment mechanism* to efficiently conceal the inherent initialization overhead of (personalized, data-protected) AI services for mobile users on the above decentralized infrastructures.

PART V (EPILOG) concludes this thesis with a brief summary of the core contributions, a discussion of their implications, and an outlook on possible future work (see CHAPTER 9).

Part II

BACKGROUND & RELATED WORK

Contribution Statement: This part is based on the following (3) publications:

- Christian Meurisch and Max Mühlhäuser. "Data Protection in AI Services: A Survey." In: *ACM Computing Surveys* 54.2 (Mar. 2021). **JCR-IF (2019): 7.990**, 40:1–40:38. DOI: 10.1145/3440754
- Christian Meurisch, Cristina Mihale-Wilson, Adrian Hawlitschek, Florian Giger, Florian Müller, Oliver Hinz, and Max Mühlhäuser. "Exploring User Expectations of Proactive AI Systems." In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*. IMWUT 4.4 (Dec. 2020), 146:1–146:22. DOI: 10.1145/3432193
- Christian Meurisch, Maria-Dorina Ionescu, Benedikt Schmidt, and Max Mühlhäuser. "Reference Model of Next-generation Digital Personal Assistant: Integrating Proactive Behavior." In: *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers*. UbiComp'17. ACM. 2017, pp. 149–152. DOI: 10.1145/3123024.3123145

I (*Christian Meurisch*) led the process of idea generation, literature review, conceptual design, study design, evaluation, and writing. *Cristina Mihale-Wilson* contributed to the study design, the evaluation, and the writing process. The master students *Maria-Dorina Ionescu*, *Florian Giger*, and *Adrian Hawlitschek* supported the study design and implemented the study tools. *Oliver Hinz*, *Benedikt Schmidt*, *Florian Müller*, and *Max Mühlhäuser* provided helpful critique and comments on the study design; the latter also contributed to structuring the survey article.

2

PERSONALIZED AI SERVICES: BACKGROUND AND USER EXPECTATIONS

This chapter presents background information around personalized AI services, conveying the foundations of this thesis. Specifically, Section 2.1 first introduces the terms and useful categorizations for personalized AI services. One important issue (relevant for this thesis) is privacy, as personalized AI services inevitably require access to user data for the purpose of personalization. Section 2.2, therefore, provides background information on privacy and discusses users' privacy concerns in this context. In Section 2.3, we close the gap that previous studies confirming these privacy concerns for AI services focused mainly on reactive ones: this thesis explores privacy concerns along with user expectations for emerging proactive AI services, which are particularly targeted by the technical contributions due to their data-demanding nature. Section 2.4 points out the implications for this thesis.

2.1 TERMINOLOGY AND CATEGORIZATION

We first elaborate on the *terminology* around personalized AI services and their underlying concepts. Given the ambiguous use of the terms in the literature, this section also clarifies their use in this thesis and introduces useful categorizations.

2.1.1 *Artificial Intelligence* (*AI*)

The term 'artificial intelligence', or 'AI', is used ambiguously today, without a clear definition [RN16]. Historically, AI was coined in 1956 to refer to the process of enabling computer systems to simulate the human brain function and handle any general cognitive task in any setting (termed *human-imitative AI*) [Jor19]. These days, AI refers more generally to an area of computer science concerned "with designing intelligent computer systems" [BF81] or "with the automation of intelligent behavior" [LS93], encompassing a huge variety of subfields (from general-purpose areas to specific tasks). However, the term 'intelligent' is controversial and not yet well-defined[2] here, as even the definition of human intelligence is difficult and controversial.

For this reason, a more pragmatic approach is used in this thesis: AI has also always been defined by AI researchers themselves in terms of the set of fundamental methods and concepts considered to belong to this field. These were, for example, simple neural networks and especially logic programming in the first rising wave of AI. *Machine learning* (ML) followed, and its great successes and further acceleration through advances in deep neural networks were the

[2] http://jmc.stanford.edu/artificial-intelligence/what-is-ai (retrieved 06/30/2021)

key drivers of the second rising wave of AI. This development was particularly boosted by the availability of massive computational power and huge data sets (the big data hype), as it became possible to apply (resource-intensive) statistical learning methods to large amounts of data. [Den18]

Although ML cannot be equated with AI, it has been shaping the greatest advances and the broadest application of AI for several years. We will, therefore, primarily have ML in mind, when we refer to AI in this thesis; at the same time, we will pay special attention to the need of modern AI/ML methods for collecting and processing large amounts of data.

Machine Learning (ML)

ML, a subfield of AI, refers to statistical methods or 'tools' that are used to empower computer systems to learn and automatically improve through experience/ data without being programmed explicitly – a comprehensive introduction to machine learning is given by [Mit97]. For the sake of brevity and due to their relevance for this thesis, we only introduce relevant terms here and focus on two categories of ML methods that rely on specific learning styles, namely *supervised* and *unsupervised learning*.

In supervised learning, the algorithms have access to 'correct' input-output pairs during the learning (training) phase, i.e., the input data is annotated with an expected target value, the so-called *class* or *label*. However, obtaining such ground-truth data (aka *labeling*) is laborious work, especially if humans have to annotate the data manually (*explicit feedback*); sometimes labeling can be obtained by actions that humans would perform anyway (e.g., selecting suggestions in mobile keyboards [Har+19]), regardless of the actual labeling intention (*implicit feedback*); and sometimes it is just infeasible. In unsupervised learning, the algorithms have access to (non-annotated) input data only, finding underlying patterns in it. The most common technique here is *clustering*, the process of finding similarities in the data and grouping them together. This training phase is followed by the inference phase, in which the trained model can be used to make predictions on new (unseen) data.

At the beginning of each chapter presenting original scientific contributions of this work, we will discuss to what extent these contributions are applicable to AI beyond ML or, on the other hand, to what extent it focuses on a subset of ML.

2.1.2 *AI Service, Algorithm, and Model*

In terms of user services in the AI context, the last two decades have seen major progress in two complementary disciplines, namely *intelligence augmentation* (IA) and *intelligent infrastructures* (II). In the former, "computation and data are used to create services that augment human intelligence and creativity". The latter describes "a web of computation, data, and physical entities that makes human environments more supportive". [Jor19] Here, the involvement of physical

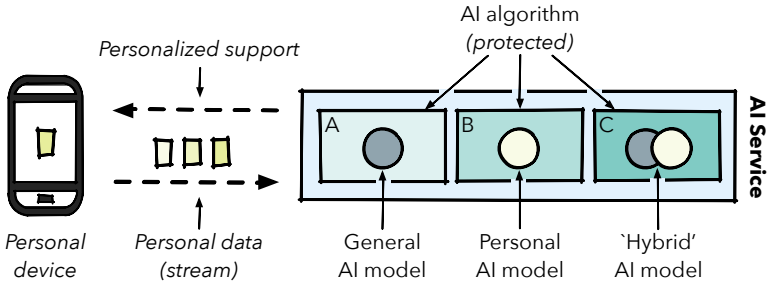


Figure 2.1: Schematic representation of a sample personalized AI service (*blue*) with three different kinds (*A*-*C*) of underlying AI algorithms (*green*); *yellow* marks target user data or its required involvement in model training (*circles*).

entities, which often refer to mobile or IoT/edge devices[3], is crucial for such distributed services (e.g., in terms of the interaction with humans, the additional available data streams, or a local execution environment) [Rag15].

We now introduce the following terms to refer in this thesis to user services in the respective discipline. As introduced in Section 1.1.2, the term 'AI services' will be used to denote any AI-powered system that provides some kind of direct support to a single user (referring to IA). The term 'distributed AI services' (a subset of AI services) will be used to refer to those AI services that require parts of them to run on (at least two) different devices, e.g., locally on personal devices for privacy reasons or 'at the edge' to benefit from short-delay computing (referring to II/EI). We use the phrase 'device-bound AI services' to refer to the subset of distributed AI services that additionally require access to sensors and/or actuators on specific (IoT/edge) devices in the physical proximity of the user in order to provide ambient support.

Figure 2.1 shows a schematic representation of such a sample AI service (*blue*). According to our definition, an AI service incorporates at least one *AI model* (*circle*), which processes input data to make a prediction (output) for a specific task. AI models can operate hierarchically, i.e., an output of one AI model can serve as an input of another. An AI model is trained by using a specific *AI algorithm* (*green*), which is usually protected, as it is part of the IP of the provider.

For the sake of completeness, an AI service may incorporate additional non-AI components, e.g., a *sensing* unit to collect data required for the underlying AI models or a *decision logic* to interpret the output of AI models and trigger appropriate actions to *actuators* [PM15]. In this thesis, for the sake of simplicity, such components are abstracted in the AI service component (*blue*), which can run distributed on several physical devices.

[3] A related (sub-)discipline dealing with AI on (edge) devices is represented by *edge AI* or *edge intelligence* (EI), with a focus mainly on aspects of resource management and communication [Zho+19; An+19].

2.1.3 *Personalization*

In the AI/ML context, *personalization* (broadly known as *customization*) is the process of tailoring an AI service, or more precisely, its underlying AI models to an individual user and her setting in order to enhance the service experience/value specifically for this user. Technically speaking, personalization aims at improving an AI model w.r.t. specific performance metrics (e.g., accuracy) or its *effectiveness* for individuals using their personal data. In the following, this section defines what constitutes *personal data* (or how it is treated in this thesis) and its value for personalization in the context of AI services. Following this, the different *model training* practices relevant to this thesis are introduced.

2.1.3.1 *Personal Data*

"The term 'personal data' has several meanings" [Sch+11]. This thesis broadly defines personal data as "[digital] data related to a natural person that can be possibly linked to that person", i.e., the digital record of "everything a person makes and does online and in the [physical] world" [HSP19, p. 284]. According to [Sch+11], personal data can be defined more precisely as "[digital] data (and metadata) created by and about people", encompassing the following three groups of data:

- *Voluntary data* is created and explicitly shared by individuals.
- *Observed data* is captured by recording the actions of individuals (e.g., locations, accelerometer data, communications data).
- *Inferred data* (or *derivatives*) is higher-level data about individuals based on the analysis of the two previous groups of data (e.g., physical activities derived from observed accelerometer data).

The following—partly interdependent—*four aspects of personal data* that affect the degree of personalization are identified:

- The *type of data* allows AI algorithms to draw different personal inferences about users, ranging from low-level (e.g., location data) to high-level data (e.g., depressive state) [Wie+17; WW14] – an initial list of categories (e.g., digital identity, communications data) is given in [HSP19, p. 285f.].
- Both the *quantity* and *quality of data* (and especially of its labels) have a direct influence on the performance of the resulting AI models [Ser+18]. Typically, more data leads to better results, up to a certain saturation point. On the other hand, bad data or erroneous signals (e.g., a small bias in a sensor) can lead to incorrect results of the AI models, regardless of the volume of data. Consequently, only enough high-quality data ensures meaningful and accurate insights.
- The *time frame of data* allows AI models to make different kinds of user/behavior modeling and predictions, on which AI services base their support:

— *present data* allows the interpretation of data only, such as in location-based services [Lan+10] or voice-controlled information retrieval [Sar17];

— *past data* allows an in-depth understanding of user behavior and user contexts, such as in fitness and health applications [Rab+15; Sch+15b];

— *anticipated future data* mostly relies on past data and allows an anticipatory behavior of AI services based on future states, such as in anticipatory mobile computing applications [PM15].

2.1.3.2 *Model Training*

In general, the need for *personal* or *personalized AI models* (*yellow circle* in Figure 2.1) stems from the fact that a *general AI model* (*grey circle* in Figure 2.1) can work well for many users but not for all [Ser+18]. This thesis basically defines the following three different types of model trainings that are widely practiced today:

(A) *general model training* relies on data from many users but not from the target user;

(B) *personal model training*, in contrast, only relies on data from the target user;

(C) *'hybrid' model training* is a combination of both, considering data from other users but also from the target user, e.g., in a subsequent *personalization step* using transfer learning [PY09].

The personalization degree of AI models depends on the personal data available to them. Thus, (A) cannot achieve personalization in the training phase. On the other hand, (B+C) come with a so-called inherent *cold-start problem* for new users: AI models cannot daw any personalized inferences for users about whom they have not yet collected enough personal data [LKH14]. By its nature, (B) is therefore more affected by this cold-start problem than (C).

2.1.4 *Degree of Proactivity*

This section introduces an adapted and refined classification of AI services to create a common understanding and to better classify and reference them in the thesis when discussing the contributions. More precisely, inspired by the proactivity continuum ranging from zero to full initiative automation of user support [Sar17], this thesis proposes the following three classes of how AI services operate and support users (*reactive–proactive–autonomous*).

Reactive Support

Reactive AI services only *respond to commands from users* in order to support them, currently representing the most widespread class of AI services. Commands can be entered through different input modalities (e.g., voice, gesture) [JS07]. Probably, the most prominent representatives of this class are conversational agents [GJ19], which have made the transition from research prototypes to

consumer products in the last decade: Apple Siri (2011), Google Assistant (2012), Microsoft Cortana (2015), Amazon Alexa (2015), to name a few [LS16]. With such reactive AI services, users can trigger information searches or simple task executions (e.g., finding the fastest route) [Bar+20].

Proactive Support

From this class onwards, AI services *proactively take actions without an explicit user request*. For this (e.g., to provide helpful advices or recommendations proactively), AI services need to access and monitor a user's data (stream). Examples include location-aware services that can send updates to weather and traffic conditions, among other things, which may be relevant for users in their current situation. [Sar17] Fitness and health applications in this class, for instance, can monitor a user's physical activity to strategically suggest changes to this behavior (e.g., "take a break" or "walk for 30 minutes") for a healthier lifestyle [Rab+15] or meeting a specific goal [Sch+15b].

In general, the actual actions must still be confirmed or taken by the users. To engage users in these actions, latest AI services also rely on predictive models to find opportune moments to notify [PM15] and intervene users [MM17] or incorporate mechanisms to persuade them [Fog99; Lan+10]. However, the key challenge in determining the type and timing of suggestions is still a hot research topic, as there are associated costs with them (e.g., if the action, relevance, or timing of the notification is wrong).

Autonomous Support

AI services in this class *act autonomously, making decisions and taking the actual actions on behalf of users without confirmation*. For instance, such AI services enter our lives in the form of digital agents [Yor+12], (un-)manned vehicles (e.g., drones [Erd+17], autonomous driving [Mau+16]), or (social) robots [Pen+19]. Even though preliminary work and first prototypes exist, this class still holds many open challenges to achieve its breakthrough. In the coming decades, this class will increasingly become the focus of research, shaping our lives the most – even having the potential to revolutionize them.

2.1.5 *Application Domains and Examples*

For practical reasons (and to refer to them consistently throughout this thesis), we will roughly categorize AI applications into three domains. This partitioning is guided by three purposes: (i) it should cover most of the users' daily life; (ii) the domain categories should be marked by AI services that provide direct support to users; (iii) the domain should be subject to ongoing AI research. If applicable, some domains are further divided into subareas [Meu+17a]. For each domain, representative application examples of AI services that are either highly cited or widely known are named to give the readers a better understanding – Table 2.1 (p. 18) gives a more detailed overview of these examples using the

terms and concepts introduced; we will use this table in particular in Section 2.4 to discuss the implications for this thesis.

Healthcare & Well-being

Well-being refers to "diverse and interconnected dimensions of physical, mental, and social well-being that extend beyond the traditional definition of health." [NI15] Along this definition, this domain is categorized into the following three subareas.

Physical health covers all directly perceivable and measurable (physical) states and body conditions of an individual, "taking into consideration everything from the absence of disease to fitness level."[4] State-of-the-art AI services can detect and analyze a user's state with respect to certain factors, and intervene or make recommendations if either a worsening is imminent or an improvement is to be achieved – many of such works can be found in the related discipline of digital behavior change interventions [Lat+13]. For instance, `MyBehavior` automatically learns a user's physical activity and dietary habits to strategically suggest changes to this behavior for a healthier lifestyle [Rab+15]. Schmidt et al. present a digital coach that automatically identifies the user's strengths and weaknesses; on that basis, it creates appropriate training plans to motivate and help a user in achieving his fitness goals [Sch+15b].

Mental health covers all emotional and psychological states of an individual, including "subjective well-being, perceived self-efficacy, autonomy, competence, inter-generational dependence, and self-actualization of one's intellectual and emotional potential, among others" – according to the World Health Organization (WHO) [Org01]. This also includes coping with stress and productive work. In literature, there are several AI services/ systems that support these factors. For instance, `InterruptMe` represents an AI service that automatically infers opportune moments for interruption [PM14]. Pielot et al. further identify opportune moments in which users are particularly open to engage with suggested content [Pie+17]. Other AI services can detect and monitor depressive states [CM15] and emotions [Rac+10], just to name a few.

Social well-being covers all social dimensions, including social acceptance, integration, and the interaction with others [TR05]. For instance, `SociableSense` models the 'sociability' of users based on their co-location and interaction patterns; it provides users with real-time feedback to foster and improve social interactions [Rac+11]. Similarly, `BeWell` calculates well-being scores to raise a user's awareness, helping users to manage their overall well-being; it also includes social interactions detected by inferences from sensor data of users' smartphones (e.g., microphone data) [Lan+14].

[4] https://www.eupati.eu/glossary/physical-health (retrieved 06/30/2021)

Table 2.1: Overview of selected state-of-the-art examples of user-supporting AI services

Sample	Subarea(s)	Issue(s) addressed	Proactivity level	Personalization	PoP	Personal data (*voluntary/observed*)	Personal data (*inferred*)
Health & Well-being							
`MyBehavior` [Rab+15]	physical	activity level, diet	💬	🟡	🔒☁	GPS, accelerometer (acc), photo	phys. activities, places, food intake
Digital fitness coach [Sch+15b]	physical	activity level	💬	🟡	☁	physical activities	workouts
`InterruptMe` [PM14]	mental	interruptibility	💬	🔵🟡	📱	GPS, acc, emotions, BT/WiFi fingerprints, notifications	places, activities, co-located people, engagement
`StressSense` [Lu+12]	mental	stress	(💬)	🔵	📱	microphone (mic)	voice-based stress
Detecting & monitoring depressions [CM15]	mental	depression	(💬)	🟡	📱	GPS, phys. activities	depressive states
`EmotionSense` [Rac+10]	mental/ social	social interactions	(💬)	🔵	📱	GPS, Bluetooth (BT) fingerprint, acc, mic	places, emotions, in/outdoor, co-located people, speaker
`SociableSense` [Rac+11]	social	sociability	🔔	🟡	📱/☁	acc, mic, BT fingerprints	co-located people, social interactions
`BeWell` [Lan+14]	*all*	activity level, sleep, social interactions	›_	🔵	📱☁	GPS, acc, mic, phone recharging	social interactions, phys. activities, sleep patterns
Activity Support							
`Google Assistant` [PM15]†	physical	daily routine organization‡	💬	🔵	☁	location, calendar	places
Guiding experiments [SWV15]	physical	experiment execution	›_	🔵	📱/☁	mic, camera, acc.	voice commands, gestures, activities
Autonomous driving [Mau+16] (e.g., `Tesla` autopilot†)	physical	driving‡	⚙	🔵	📱/☁	GPS, (*video, radar, car telemetry, ...*)	mobility patterns, (*driver/car states,..*)
Conversational agents [GJ19] (e.g., `Amazon Alexa`†)	digital	info retrieval, task exec.‡	›_	🔵	☁	mic, *task-specific data*	voice commands (intents)
`Google Duplex` [LM18]†	digital	appointments	›_/⚙	🔵	☁	—"—	—"—
Spam filtering [BB08]	digital	organization of emails	⚙	🔵🟡	☁	emails	contacts, categories
Ambient Intelligence							
Smart speakers [GJ19] (e.g., `Apple HomePod`†)	smart home	home control‡	›_	🔵	☁	mic, *task-specific data*	voice commands (intents)
`SHE` vision [Che+17]	smart home	energy management	(⚙)	🟡	📱☁	electricity, GPS, activities, calendar	usage patterns, daily demands
Detecting drowsiness [CK17]	smart car	drowsiness	(🔔)	🔵🟡	(📱)	camera	drowsiness level
`SLaaP` vision [Müh+20] (e.g., *safe guidance*)	smart city	nightly risk areas‡	(💬)	🔵	🔒☁	(depth) camera	location, activity, (*risk assessment*)

Encoding: **proactivity levels** (see Section 2.1.4): ›_–reactive, proactive (🔔—referring to 'inform the user'; 💬—referring to 'make personalized recommendations'), ⚙–autonomous, ()—indicating a possible use of the system, which is, however, not realized in the respective paper; **personalization** (see Section 2.1.3): 🔵–general model, 🟡–personal model, 🔵🟡–hybrid model (e.g., with a subsequent personalization step); **place of processing (PoP)**: 📱–local, ☁–remote, 📱☁–hybrid (local–remote), 🔒☁–hybrid but confidential (personal data remains local); **misc.**: †–proprietary examples; ‡–only these selected issues are discussed

Activity Support

AI services can support users with activities or take them over to achieve the desired result. This kind of support is grouped into the following two subareas.

Support with physical activities covers all physical actions of an individual, e.g., movements, transportation, or cooking. For instance, Google Assistant supports users in organizing their daily lives by suggesting when to leave places, finding the fastest routes, and identifying the best mode of transportation. [PM15]. Other AI services can also support specific professional activities, such as conducting experiments in a laboratory [SWV15] or detecting nurse activities in a hospital [Ino+16].

Support with digital activities covers all actions related to an individual's digital world. Probably the best known example of this area is email spam filtering—concerning the processing of emails to organize them according to specified criteria [BB08]. Other AI services help users by automatically rescheduling appointments or organizing tasks [Yor+12].

Ambient Intelligence

This last category refers to the support of users through the environment [Sad11; Wei91]. Examples of such smart environments include *smart home*, *smart car* (/*mobility*), and *smart city*. In the former, AI services can, for instance, automatically regulate air and light conditions based on the users' preferences [DH08]. In smart car environments, several AI services support the driver in various situations—such as parking aid, collision avoidance, drowsiness detection—or through infotainment assistance [MZH19; KBS16]. In the smart city context, recent AI-based approaches and visions rely, for instance, on so-called 4D models captured and processed at nearby smart street lamps to offer user services for Augmented Reality (AR), civil protection, or emergency assistance [Müh+20].

2.2 PRIVACY IN AI SERVICES

As illustrated in Table 2.1 (p. 18), personalized AI services can address various user-related issues, having the potential to make the everyday life of users easier and better. However, such services inevitably require access to appropriate user data, which can be highly sensitive (see the last two columns of Table 2.1). Therefore, *user privacy* is an essential requirement. This section will first introduce definitions and provide background information on privacy, followed by an analysis of users' *privacy concerns* identified in the literature.

2.2.1 *Definitions and Background*

Privacy has been defined and interpreted in several ways by researchers from different communities or by different governments—i.e., there exists no common, overarching concept [Moo03; Solo6]. In the following, terms around privacy are

clarified as they are used in this thesis.

2.2.1.1 *Privacy*

Westin defines *privacy* in terms of information control as the right "to control, edit, manage, and delete information about themselves and decide when, how, and to what extent that information is communicated to others" [Wes68]. This definition applies to any personally identifiable data that in any way reveals information about an individual, including situations in which the actions of some individuals affect the privacy of others (termed *interdependent privacy*). The latter can be quite challenging, as individuals do not always have full control over the sharing policies of such interdependent data (e.g., a photo with several – possibly unwilling or unsuspecting – individuals, taken and shared by only one individual). [HTH19] With the General Data Protection Regulation (GDPR)[5] and The California Consumer Privacy Act (CCPA)[6], first governments—the European Union (EU) and the US state of California, respectively—have established appropriate, strong legal frameworks for the protection of their citizens' privacy.

2.2.1.2 *Data Protection and Ownership*

Whereas "data privacy is all about authorized access, concerning who has it and who defines it", *data protection* is essentially a technical issue of what data privacy dictates, concerning the securing of data against unauthorized access (data security) [Hov17]. Closely related to data protection is data ownership [HSP19]. Derived from English common law on ownership rights, Pentland defines *data ownership* as follows: individuals (1) have the right to *possess* their personally identifiable data, (2) must have full control over the *use* of their data, and (3) have the right to *dispose* or *distribute* their data [Pen09].

2.2.1.3 *The Privacy Paradox*

The *privacy paradox* is a phenomenon concerning the online behavior of users, which describes the discrepancies between user attitude and their actual behavior: although users claim to be very concerned about their privacy, they do very little to protect their personal data [NHH07; JW18]. There are multiple theories explaining the privacy paradox [Sol21]; the most convincing of which is probably the *privacy calculus*: users decide whether to disclose personal data on the basis of a risk-benefit calculation, in which the benefits is ultimately preferred to the risks [BD17]. The reasons for the latter include a lack of awareness and uncertainty, missing rationality, context dependency (e.g., social setting), and malleability of preferences [GGV18].

A special subcategory of the privacy paradox is the *personalization-privacy paradox* [Xu+11; CS05], which is now discussed in the context of AI services.

[5] https://gdpr.eu (retrieved 06/30/2021)
[6] https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375 (retrieved 06/30/2021)

Providing more, appropriate, and better personal data to AI services usually leads to a higher personalization degree of them—especially in the training phase (see types B and C in Section 2.1.3.2, p. 15)—and thus, increases their value for the user. At the same time, providing personal data out of the user territory (e.g., to the provider's cloud) requires the user to sacrifice a certain level of privacy. Consequently, users always have to make a compromise between the personalization of AI services and their privacy, in which personalization tends to negate the downside of perceived risks to privacy. In particular, it is often difficult for users to assess the actual risk to their privacy – personal data can be of varying *sensitivity to privacy* depending on the extent of the individual factors described in Section 2.1.3.1 (p. 14).

Solove, in turn, argues in 'The Myth of the Privacy Paradox (2021)' that "it is perfectly rational for people [..] to fail to make good assessments of privacy risks and to fail to manage their privacy effectively." Giving individuals more responsibility for managing their privacy, as privacy regulations such as the recent CCPA often seek to do, imposes on them a vast, complex, and never-ending project that does not scale, as we are currently experiencing with cookie settings, for example – this strategy will not provide effective privacy protection. Instead, he concludes, regulations should focus more on "regulating the architecture that structures the way information is used, maintained, and transferred." [Sol21] This thesis will make a technical contribution to this by proposing a suitable architecture with incorporated data protection mechanisms that give users more control over their personal data without placing a greater burden on them.

2.2.2 *Privacy Concerns of Users*

We now have a look at the specific *privacy concerns* of users when using data-demanding services in general – these can be transferred to (increasingly data-intensive) AI services. For this, the following taxonomy adapted from Wang et al. [WLW98] is used to better name and classify these privacy concerns:

- *Improper acquisition* concerns the gathering of personal data without notice or acknowledgment from users, e.g., through unauthorized access to it, additional data collection beyond the service, or unnoticed monitoring of users' physical and digital activities.
- *Improper use* concerns the *analysis* and *transfer of personal data*. In the former case, personal data is analyzed without proper notice to draw conclusions about user behavior. In the latter case, personal data is transferred to the cloud or third parties without users' notice and acknowledgement.
- *Improper storage* is related to the concerns about data confidentiality and integrity.
- *Privacy invasion* mostly concerns unwanted solicitation (e.g., sending information to potential users) and unwanted executions of tasks without users' acknowledgement or permission.

According to [AEY10], the main concern of users is the improper use of their data, in particular *data transfer* or *data sharing*. In particular, users are mostly concerned about the resulting disclosure of their behavior and the trading/selling of personal data to third parties—often leading to loss of data ownership without the awareness and consent of users. This second point (improper use) is closely connected with the first point (improper acquisition) and the third point (improper storage), which should therefore be addressed together. In this thesis, we mainly focus on these privacy concerns and address them by proposing different data protection approaches geared to each other (see Part III).

2.3 USER EXPECTATIONS OF AI SERVICES EXPLORED THROUGH A STUDY

A variety of studies confirms these privacy concerns from Section 2.2.2 specifically for AI services as well, but mostly only for reactive ones [BC11; LS16; Kis+16; Cow+17; LQG17; DHA18; Bri18; Ben+18] or only for very specific proactive ones [BD03; MZH17]. There is still a lack of studies for a wide range of proactive or even autonomous AI services in this context. However, such AI services in particular are of high relevance to this thesis due to their data-demanding nature—e.g., they typically need to *continuously* analyze user data in order to act proactively (see Section 2.1.4).

In light of this deficit, and before moving on to the technical contributions dealing specifically with such (proactive, data-demanding) AI services, this thesis contributes to a better understanding of the extent of which privacy concerns arise in such AI services. As proactive AI services are not yet widespread in the consumer market, we take a broader perspective in this section and first explore the user expectations—representing "an individual's prediction or anticipatory judgment about what they should or will receive through the performance of a product or service" [Bri18]—to inquire about privacy concerns on this basis.

2.3.1 *Methodology*

To explore the user expectations of proactive AI services, a user study with 272 participants was conducted within the scope of this thesis. More precisely, the study examines 23 selected use cases or scenarios in which user support is provided by AI services (see Table A.1 in the Appendix), covering different application domains (see Section 2.1.5 and Table 2.1, p. 16 ff.).

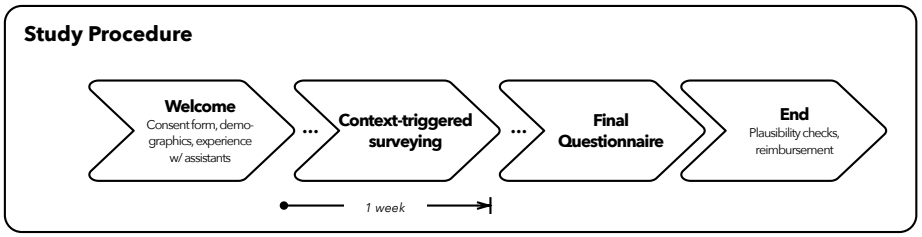
We opted for an *in-the-wild* approach, as it provides more reliable results compared to an online survey or vignette study [Fel+12; BFK17]: participants do not have to answer questions about fully-hypothetical situations but experience the real (physical, cognitive) situation in which the questions for the specific use case fit. However, the usual methods for such a study, such as experience sampling [BFK17], cannot be applied, as proactive AI services are not yet widespread and implementing a variety of use cases for this study is too complex and laborious. Therefore, we propose a new study method in which participants

Table 2.2: Description of the developed *context-based triggers*, which can be combined depending on the target use case

Group	Possible trigger criteria
Time	Timestamp, time range, time of day, on weekdays/weekend, day of the week
Location	Entry/exit event for a geo-fence, detected places of the user (home, work, others), en route
Activity	Still, walking, running, on bicycle, in vehicle, sleeping
App usage	Opening/leaving event for specific apps or categories
Wake-up event	First smartphone use after sleep activity

are asked about hypothetical use cases, but only when they experience the likely context of use, i.e., the situation in which the AI service provides its support.

Especially for this study method, we implemented a smartphone app that is able to recognize the relevant situations and trigger in-situ questionnaires about the corresponding hypothetical use cases (see Figure A.1 in the Appendix). For this, we developed different *context-based triggers*, which continuously analyze sensor data from the following five groups: *time*, *location*, *activity*, *app usage*, and *wake-up events* (see Table 2.2). These context-based triggers can be combined (rule-based) as needed to cover all use cases of the study. For instance, to trigger an in-situ questionnaire for the *social health support* use case *"You have an appointment at 7 p.m. with some friends. Today, the roads will be very busy. So, if you leave home too late, you might not make it in time to your appointment."* (see Use Case #9, Table A.1 in the Appendix), the app combines the time (*6 p.m. - 6:45 p.m.*) and the location (*home*) triggers. For further technical details, the reader is referred to [Meu+20a].



2.3.1.1 *Study Procedure*

The study ran for one week. The procedure of the study was as follows:

1) Welcome. The participants expressed informed consent. Then, they completed a demographics questionnaire and reported their experience with digital assistants – the term 'assistant' is used as a substitute for 'AI service' to better communicate the more technical concept of AI services to the participants.

2) Context-triggered Surveying. In this step, the app triggered in-situ questionnaires for a specific use case when participants came into relevant situations in their

daily life during a period of one week (see Figure A.1b in the Appendix). Unseen or dismissed questionnaires are rescheduled when the participants are back in the corresponding situation. To avoid overburdening or disturbing the participants, a maximum of eight questionnaires were triggered per day. Each questionnaire comprises three parts:

USE CASE DESCRIPTION The participants see the explanation of the use case scenario in which an AI service could support the participants in their current situation (see Figure A.1c in the Appendix, p. 253).

ASSESSING THE DEGREE OF PROACTIVITY The participants are given four different proactivity options ranging from reactive to autonomous support (see Section 2.1.4), plus a *no support*-option; they are asked to make their choice for one of these options (see Figure A.1d in the Appendix, p. 253).

QUESTIONS FOR UNDERSTANDING THE CHOICE Unless participants opted for the highest proactivity level, they are asked to rate a set of questions (see Figure A.1e in the Appendix, p. 253), which comprises six different statements (see Table A.2 in the Appendix). These questions help us to better understand why the participants made exactly this choice and did not opt for a higher level of proactivity. Based on this, we explore the user concerns with this type of AI services.

3) Final Questionnaire. The participants received a final questionnaire asking the hypothetical use cases that they had not answered in situ (e.g., because the use case was simply not triggered by the participant's daily behavior).

4) End. To ensure high data quality, rough plausibility checks based on completion times and response patterns are applied. Finally, we reimbursed the participants who completed the study with $8 each.

2.3.1.2 *Recruitment and Participants*

We recruited a sample of 272 participants residing in different countries. For the recruitment and compensation processes, we used the online platform *Prolific*[7].

The participants were on average 35.0 years old (SD = 9.8, Min = 18, Max = 64). About half of the participants identified as female (N = 138), 134 identified as male. From the participants, around a quarter of the participants (N = 70) reported that they had never used an assistant. In contrast, the majority of the participants uses at least one assistant from time to time (56.3%), often (14.7%), or very frequently (3.3%).

2.3.2 *Findings and Discussion*

This section presents only the study results from the in-the-wild study that are relevant to this thesis and discusses them in this context. The first part will

[7]https://prolific.ac (retrieved 06/30/2021)

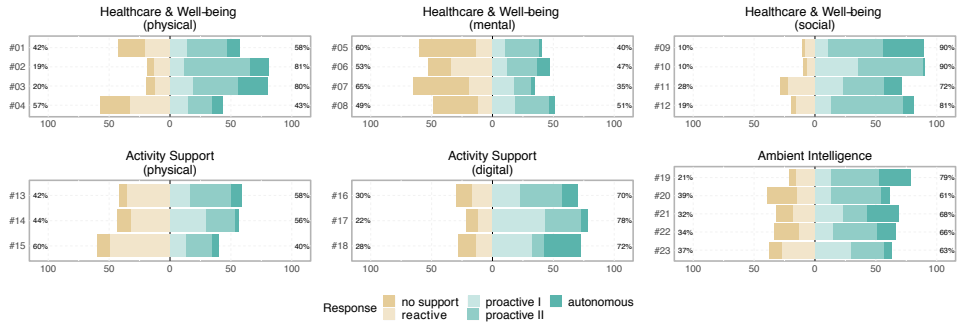


Figure 2.2: The in-situ responses of the participants to the *expected proactivity levels* for the different use cases. The use cases are abbreviated by their identifiers starting with # (see Table A.1 in the Appendix) and grouped by the application domains introduced in Section 2.1.5. The respective summed percentages per use case (row) indicate the ratio between the non-proactivity levels (left, yellow ochre tones) and the proactivity levels (right, turquoise tones).

contribute to a better understanding about users' attitudes towards AI services, representing the preparation for the actual analysis of the user concerns, which follows in the second part.

2.3.2.1 *Understanding the Expected Proactivity*

We first examines the measurement effect of the in-situ design of the study. To quantify this effect, the proactivity levels are encoded with the following ordinal scale: *no support*→*0*; *reactive support*→*1*; *proactive support I*→*2*; *proactive support II*→*3*; and *autonomous support*→*4*. As the data is not normally distributed (Shapiro–Wilk test, $p < .001$) [McC08], this analysis applies a non-parametric Kruskal–Wallis test to test the null hypothesis stating that "the population medians for the two methods, namely the proposed method and conventional online surveys, are equal". The *in-situ* reported proactivity levels are used for the former, while the reported proactivity levels *at the end of the study* are used for the latter. According to the results, $\chi^2(1) = 14.3, p < .001$, there exists a statistically significant difference between both measurement methods on capturing participants' expectations, and hence, leads to the rejection of the null hypothesis. Given this difference, and because several studies have shown that answers given in situ are more reliable than answers to purely hypothetical questions [Fel+12; BSR18; BFK17], we will therefore only consider the in-situ responses (N=3,168) in what follows.

We now investigate in an exploratory way whether and what kind of support users expect from the respective AI services. Across all use cases, the most frequent choice is *proactive support II* ($\mu = 32.5\%$, $\sigma = 12.5\%$), followed by *reactive support* ($\mu = 20.1\%$, $\sigma = 10.5\%$) and *proactive support I* ($\mu = 19.7\%$, $\sigma = 9.0\%$). However, in some cases ($\mu = 15.5\%$, $\sigma = 12.8\%$) participants opted not to be supported at all – neither reactive nor proactive. Interestingly, the least preferred choice of participants is *autonomous support* ($\mu = 12.2\%$, $\sigma = 9.6\%$).

Figure 2.2 depicts these user expectations per use case (row) and application domain (plot) – use cases are abbreviated by their identifiers (see Table A.1 in the Appendix); the *yellow ochre tones* on the left indicate the desire to receive no proactive support (i.e., no support or only reactive support) while *turquoise tones* on the right show expectations for proactive support. In over 78% of all use cases (18 out of 23), the majority of participants opted for proactive support by AI services. Especially for the use cases #9 and #10, which concern appointments and recurring events in the *social* context respectively, more than 90% of the participants would like to receive proactive support. Specifically, participants would like to receive autonomous support for use cases such as (#18) 'do not disturb'/availability handling and (#19) smart home control, i.e., light and climate control – these use cases relate to *digital activity support* and *ambient intelligence* respectively. In contrast, many participants want to receive no support or only reactive support in the (#15) shopping use case and in most of the *mental health* related use cases (#5-8).

In terms of application domains, as introduced in Section 2.1.5, a non-parametric Kruskal–Wallis test reveals significant differences between these areas with $\chi^2(5) = 194, p < .001$. To further investigate these differences, we applied Dwass-Steel-Critchlow-Fligner (DSCF) pairwise comparison tests with Bonferroni correction to account for multiple testing. The tests reveal significant differences between the desired proactivity levels for *mental health support* ($\mu = 1.40$, $\sigma = 0.94$) and those of all other areas ($p < .001$) – the average preferred level in the former case corresponds to the reactive support. In the areas of *physical health support* ($\mu = 2.01$, $\sigma = 0.88$) and *physical activity support* ($\mu = 1.99$, $\sigma = 0.78$), participants rather prefer a more proactive support – there is no statistically significant difference between these two groups ($p = .999$). However, there are significant differences ($p \leq .004$ for all comparisons) to the remaining three areas, namely *ambient intelligence* ($\mu = 2.24$, $\sigma = 0.97$), *digital activity support* ($\mu = 2.28$, $\sigma = 0.79$), and *social well-being support* ($\mu = 2.66$, $\sigma = 0.94$). There is no significant difference between the first two areas ($p = 1.000$), but there are differences between the first two areas and the latter area ($p < .001$ for both).

All in all, these results show the predominantly positive attitude of users towards AI services, including for increasingly emerging proactive (data-demanding) services, almost for all application domains – mental health support in particular represents an exception here. This first part prepared the actual analysis of the user concerns, which follows next.

2.3.2.2 *Understanding the Reasons for the Choice*

We now analyze the potential user concerns about these AI services by exploring the reasons for the choice of the desired proactivity level and why more proactivity was not desired. Figure 2.3 shows the ratings of the participants for the set of questions grouped by proactivity levels (plot); the questions (row) are abbreviated with Q1-Q6 (see Table A.2 in the Appendix).

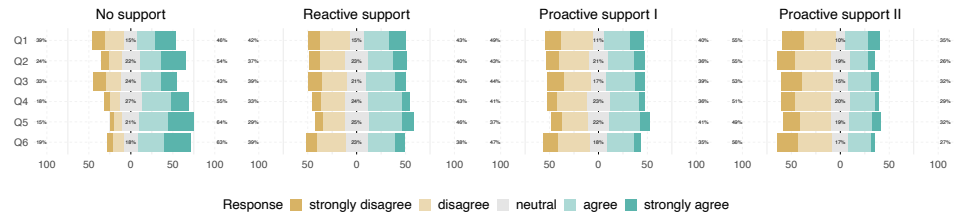


Figure 2.3: The responses to the additional questions asked immediately after the user's choice for a proactivity level lower than 'autonomous support'. The questions are abbreviated with Q1-Q6 (see Table A.2 in the Appendix). The respective summed percentages per row reflect the ratio of responses indicating disagreement (left, yellow ochre tones), neutrality (middle, gray), and agreement (right, turquoise tones).

The first two questions deal with the relevance and the importance of the use case. Nearly half (46%) of the participants who desired no support stated that the use case was not relevant in their lives (Q1), and they can therefore not imagine any appropriate support. Just over half (54%) of this group felt that the use case would be too important to be supported by an AI service (Q2). The reasons for this may be found in part in the stated concerns of users: nearly two thirds of the participants who opted for the 'no support'-option are concerned that AI services with a higher degree of proactivity would be too intrusive (64%, Q5) or violate their privacy (63%, Q6); these are followed by concerns about trust in the abilities of such AI services (55%, Q4), especially that AI services with a higher proactivity level would make too many mistakes (43%, Q3). The more open participants are to a higher level of proactivity, the fewer expressed such concerns (e.g., less than one-third of the participants who opted for 'proactive support II').

As this thesis is primarily concerned with the *user privacy concerns* (Q6), we will limit our discussion here to the corresponding responses and look at these in more detail. Specifically, it is remarkable that participants who desired a higher proactivity level are less concerned that AI services might violate their privacy. A non-parametric Kruskal–Wallis test on the underlying non-normally distributed data (Shapiro–Wilk test, $p < .001$) confirms significant differences between these desired levels with $\chi^2(3) = 534$, $p < .001$. To further investigate these differences, we applied DSCF pairwise comparison tests with Bonferroni correction to account for multiple testing. We found significant differences between all groups ($p = .001$ for the comparison between 'reactive support' and 'proactive support I'; for all others, $p < .001$).

Figure 2.4 depicts the expressed privacy concerns for the different use cases (see Table A.1 in the Appendix) and application domains (introduced in Section 2.1.5). Accordingly, most privacy concerns were expressed for use cases relating to mental health (by up to 63% of the participants), physical health (by up to 58% of the participants), and digital activity support (by up to 48% of the participants). This may explain the low level of proactivity desired by the participants in the area of mental health; on the other hand, these

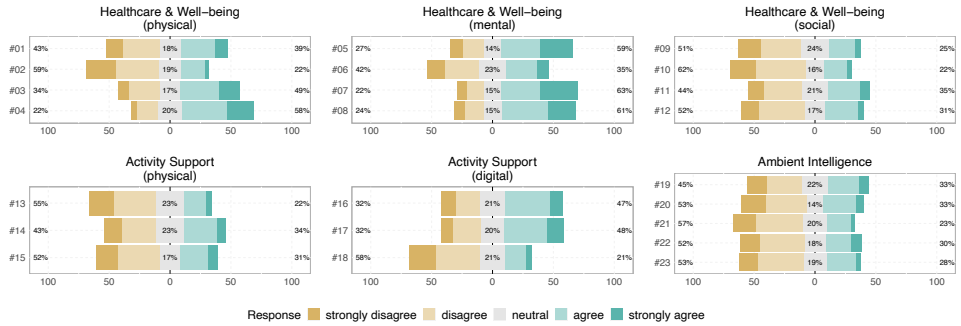


Figure 2.4: The responses of the participants to question Q6 regarding *privacy concerns* (see Table A.2 in the Appendix) for the different use cases. The use cases are abbreviated by their identifiers starting with # (see Table A.1 in the Appendix) and grouped by the application domains introduced in Section 2.1.5. The respective summed percentages per row reflect the ratio of responses indicating disagreement (left, yellow ochre tones), neutrality (middle, gray), and agreement (right, turquoise tones).

privacy concerns may be attributed to the fact that AI services in these areas require access to very sensitive user data (see Table 2.1, p. 18). In all other areas, participants with privacy concerns were in the minority, but there were some, namely up to one third of the participants.

All in all, these results confirm privacy concerns as one of the reasons when participants either do not want support from AI services or do not desire a higher level of proactivity. This, in turn, suggests that addressing privacy concerns is one of the key building blocks to make AI services more appealing to reluctant users, especially in areas like mental health in which AI services need access (and that continuously) to very sensitive data. In the next section, we will discuss the implications in the context of this thesis.

2.4 IMPLICATIONS FOR THIS THESIS

This chapter presented background information around personalized AI services, conveying the foundations of this thesis. More precisely, we introduced the terms and useful categorizations for AI services and explained the underlying concepts to establish **a common understanding**. This chapter also added a user study that explores the users' expectations and concerns about such increasingly emerging services. Complementing previous studies in the literature, which are largely limited to reactive AI services, the study results confirmed in particular (i) a predominately positive user attitude towards a variety of proactive AI services but also (ii) privacy concerns as one of the reasons for reluctance among some users. These results underscored the need for appropriate data protection mechanisms specifically for AI services and the **high relevance** of this topic.

This chapter specifically pointed out the following **characteristics of AI services** that data protection mechanisms must take into account in this context:

– AI services require (partly even continuous) access to user data to be useful to the user. For example, user data is needed for AI services to act proactively or for the personalization step (see Sections 2.1.3 and 2.1.4). Table 2.1 (p. 18) showed examples of the data-demanding nature of AI services: user data can be of different types (see also Section 2.1.3.1) and highly sensitive, especially in the mental health domain. In particular, this may lead to user privacy concerns regarding data acquisition and analysis (see Section 2.2.2).

– AI services can be distributed across multiple locations, e.g., on personal devices, edge devices (we refer to them as 'local share') or in the cloud (we refer to this as 'cloud share'). In Table 2.1 (p. 18), this can be observed mainly in the mental health domain (due to the sensitive data) and in the ambient intelligence domain (due to the support type). In particular, this may lead to user privacy concerns regarding data transfer and storage (see Section 2.2.2).

– Both personal model training (see Point B in Section 2.1.3.2) and the local personalization step in 'hybrid' model training (see Point C in Section 2.1.3.2) require the providers to perform (at least parts of) their AI services locally or outside their territory. This can lead to issues with both IP rights (e.g., regarding protected AI algorithms/models) and efficiency (e.g., regarding local resources).

– For personalization of AI services, enough and appropriate user data is needed; in the case of supervised learning, ground-truth data or labels are also needed 2.1.3.2). If such data is not directly available, this typically leads to a cold-start problem and, in the case of explicit feedback to obtain ground-truth data, to a burden on the user (see Section 2.1.1).

In the next chapter, which is also the last chapter of Part II, we will first categorize and review existing data protection approaches in the context of (data-demanding, single-user) AI services. Based on the established requirements for such approaches, open issues are identified that inform the contributions of this thesis. Part III then presents the (technical) core contributions of this thesis.

3

DATA PROTECTION RESEARCH IN AI SERVICES: CLASSIFICATION AND SURVEY

Before this thesis presents the technical contributions to data protection research in AI services (see Part III), this chapter gives an overview of existing relevant approaches and discusses them. Section 3.1 first introduces a categorization scheme based on the possible threats at different levels—namely the *management*, *system*, and *AI levels*. The following three sections (3.2–3.4) reflect this structure. Each section establishes a set of category-specific requirements and assesses the relevant research with regard to these requirements. Section 3.5 summarizes the open issues identified, which inform the contributions of this thesis.

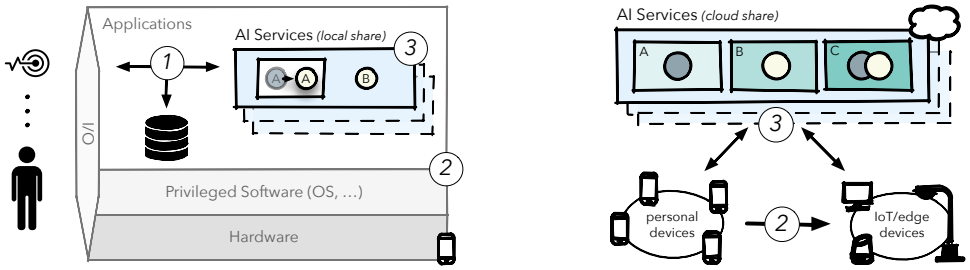
3.1 CLASSIFICATION OF DATA PROTECTION RESEARCH

A generic pipeline in information systems represents *input–processing–output* (IPO). Since the input and output (I/O) protection in such systems are particularly associated with challenges in sensing [Lan+10] and Human-Computer Interaction (HCI) [DTS19], this thesis focuses on the processing aspect only. In the context of this thesis, this concerns mainly the data access protection. This middle part of the simplified IPO model may also include *storing* and *networking* capabilities, especially in distributed computing systems.

Figure 3.1 shows a simplified IPO example of an AI service environment. Based on this environment, this thesis proposes the following categorization scheme using three different technical perspectives:

1. *Management Perspective: Data Access Rights & Storage Protection*. After sensing (*input*), user-related data enters the system/platform; it is either stored in the data storage and/or immediately processed by the AI services. This first category of data management (①) is concerned with protecting this data from *unauthorized access* when storing, retrieving or acting on it. For this, the approaches in this category assume trusted underlying systems/devices.

2. *System Perspective: Data Protection against System Threats/Attacks.* While the approaches in the former category assume trusted underlying systems, we now look at these systems when they are untrusted, e.g., when users use cloud-based or distributed AI services hosted by third parties. Specifically, this category (②) is concerned with the protection of user data and the locally-running proprietary code of the providers against *threats and attacks* from the system – e.g., from the hardware and privileged software (e.g., OS, hypervisor). Therefore, protection approaches in this category must support secure processing not only locally on personal devices (see



(a) Local view (*personal* or 'nearby' *IoT/edge* devices) (b) Global view

Figure 3.1: Schematic representation of an example AI service environment, categorized along three technical perspectives: ① protection at management level, ② protection at system level, and ③ protection at AI level. The notation/color coding for the AI services was introduced in Section 2.1.2.

Figure 3.1a, ②) but also on third-party IoT/edge devices (see Figure 3.1b, ②). It is important to note that this thesis considers mechanisms for protecting both data 'at rest' (i.e., when it remains on the personal device) and data 'in motion' (i.e., when it is communicated across devices) – the latter is also particularly relevant in edge computing scenarios [Sat17].

3. *AI Perspective: Data Leakages & Improper Use.* AI services need to access and process the data in order to provide personalized support (*output*). However, conventional AI algorithms can suffer from inherent 'data leakages', depending on their use and split between local sites (see Figure 3.1a, ③) and the provider's cloud (see Figure 3.1b, ③). Accordingly, this category (③) is concerned with data protection challenges in data sharing with authorized but untrusted AI services, i.e., when data (or metadata) leaves the user territory or when they draw new conclusions about users from this data – even beyond their intended use [Lan19]. Like ①, this category also assumes trust in the underlying systems and devices.

In general, data protection approaches for AI services must take the inherent nature of AI algorithms/models into account in order to avoid performance issues w.r.t. efficiency or effectiveness. All categories proposed above are non-exclusive but rather complement and overlap each other. Indeed, comprehensive protection (without trust assumptions) can only be ensured, if approaches in the various categories are reasonably combined and work together synergistically.

The relevant data protection approaches are categorized on the basis of the above scheme (see Figure 3.2 for an overview). The following three sections (3.2-3.4) reflect this structure. Each section first introduces a set of category-specific requirements. The protection approaches relevant to the respective category are then surveyed and assessed against the established requirements. The open challenges identified from this assessment provide this thesis with a framework for its contributions.

It is important to note that (i) some of the approaches presented in the following have not been developed directly for AI services; (ii) others may address several

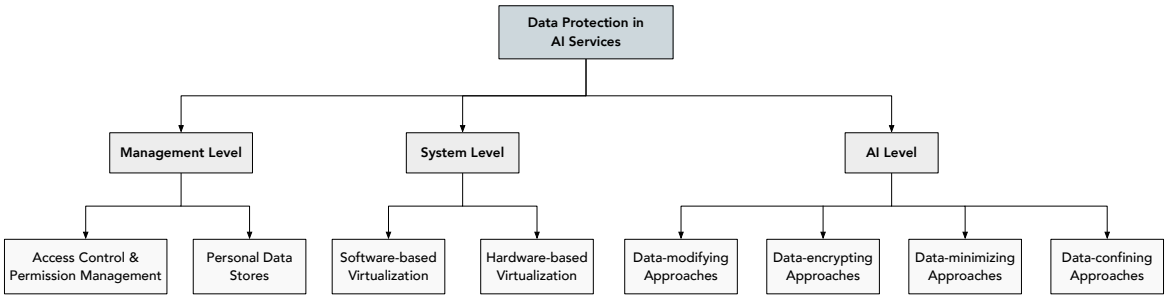


Figure 3.2: Categorization scheme of data (access) protection approaches in AI services

aspects and thus fall into more than one category. In the former case, the approach is discussed from the perspective of its applicability in the context of AI services. In the latter case, the approach is assigned to a category based on its primary protection aspect or primary challenge addressed.

3.2 PROTECTION AT MANAGEMENT LEVEL

This first category contains data protection approaches *at management level* (see Figure 3.1, ①), i.e., protection against *unauthorized access* ***within the system*** when storing, retrieving or acting on user data. Specifically, approaches in this category assume trust in the underlying systems/ devices and the authorized services providers.

3.2.1 *Specific Requirements*

This section first establishes category-specific requirements that can be addressed by approaches under the given trust assumptions in this category. The requirements are divided into three groups, namely *data protection* aspects, *runtime* aspects, and *applicability* aspects. The group concerning data protection aspects, which have a direct impact on user privacy, contains the following requirements:

- *Authentication:* The system should be able to verify that a user or service entity is what it claims to be. After successful authentication, fine-grained authorization can follow.
- *Authorization:* The system should be able to determine whether an authenticated user or service entity has access to particular resources and to what extent.
- *Data confidentiality (observed/inferred/metadata):* The system should protect personal data – both observed and inferred, as well as metadata – against disclosure, theft, and unintentional, unlawful or unauthorized access. This thesis argues that when personal data leaves the user territory without appropriate modifications (e.g., encryption), it cannot be reliably kept confidential.

The group concerning runtime aspects, which have a direct impact on possible architectures and the resulting performance, contains the following requirements:

- *Privacy-preserving processing:* Personal data should be processed in a privacy-preserving environment that is either fully under the control of the users or ensures that no data can be leaked by an (untrusted) service.
- *Data store (central/per-app):* Personal data can be stored centrally and/or per service. Depending on the design, the availability of personal data can be better ensured.
- *Low performance overhead:* Additional or complementary data protection mechanisms should not affect performance or entail only a low overhead.

The group concerning applicability aspects, which have a direct impact on whether and how simply a provider can apply the approach to its AI services, contains the following requirements:

- *Support of any data type:* Protection mechanisms should be designed to support all types of data, so that providers are not restricted in their AI services.
- *Support of any AI algorithm:* Protection mechanisms should be designed to support all underlying AI algorithms, so that providers can easily deploy their unmodified AI services.
- *Proof-of-concept prototype (implemented/evaluated):* The proposed data protection approach/ concept should have been implemented and evaluated to show its feasibility.
- *Low platform-specific dependence:* Protection mechanisms should be designed in such a way that AI services do not need to integrate platform-specific programming.
- *Cross-site deployment/use (local/nearby/cloud):* Depending on the use case (e.g., due to latency or resource demands), distributed AI services may make use of multiple sites (e.g., local, nearby IoT/edge devices or the cloud). Protection mechanisms should support such deployment at multiple sites (typically, this will imply the use of one site as a kind of 'base station').

3.2.2 *Protection Approaches*

According to the underlying mechanisms, the different protection approaches at management level are classified into two sub-categories as follows.

3.2.2.1 *Access Control & Permission Management*

Conventional access control mechanisms such as discretionary, role-based, identity-based, mandatory, attribute-based, and task-based access control are widely used in centralized databases [BGK11]. However, the administration of such access control mechanisms is complex and critical; it is therefore usually handled by security experts. Another – and probably the most critical – issue is that personal (raw) data needs to be stored in centralized databases to which the

AI services get partial access. The adaptation of such access control mechanisms to ubiquitous or IoT environments is quite limited, as the access control policy is often part of the database schema definition [TAP17]. In contrast, fully decentralized approaches such as Pretty Good Privacy (PGP) [Gar95], Web of Trust (WOT) [Zim95], and Friend of a Friend (FOAF) [GCB07] are more flexible sharing models. However, such approaches require individuals to define each basic rule manually, which places a heavy burden on them and is more prone to error. Also, not all of such approaches (e.g., FOAF) are well suited for personalized (single-user) AI services.

In the following, selected approaches that are more user-friendly and better applicable for ubiquitous/ IoT environments and especially for AI services are surveyed. Early approaches are still based on *policy enforcement*. For instance, `pawS` (presented in 2002) is proposed as a privacy infrastructure, which relies on privacy beacons and proxies [Lan02]: nearby beacons announce the data collection of each service and their policies; the user's mobile device delegates this request to the user's cloud-based privacy proxy that contacts the corresponding service proxy; both negotiate the collection and use of data. However, the confidentiality of the data cannot be guaranteed, as the data leaves the user territory; also, AI services in `pawS` must be based on a platform-specific infrastructure. Another approach, `KP-ABE`, uses attribute-based encryption to enforce fine-grained sharing of encrypted data [Goy+06]: while it allows selective sharing of data and granting access to it, this data is decrypted and used outside the user territory. The next approach in this category, `SemaDroid`, extends Android's sensor management framework to enforce a more fine-grained access control [XZ15]: so-called *SemaHooks* are placed within existing OS components, intercepting sensor access requests from apps to enforce new policies during data collection and to monitor the data usage of apps. However, `SemaDroid` is limited to Android OS and specific sensor data. A similar, more general concept constitutes *privacy mediators*, which is expected to run on edge devices [Dav+16]. Neither approach can fully ensure data confidentiality and integrity: raw data is directly passed to authorized apps in `SemaDroid`; in *privacy mediators*, the data can still be aggregated or obfuscated before it is ultimately released to the app or the associated cloud.

To address integrity and auditing issues, *Blockchain-based access control* approaches were proposed – in-depth surveys on this subject can be found in [ZXL19; Zhu+19; Hua+21]. Examples of such *decentralized* approaches include `Enigma` [ZNP15], `ControlChain` [PGD17], and *Blockchain-based audible store*. The former uses distributed hash tables (to store secret data), an external blockchain (to manage access control and serve as a tamper-proof event log), and secure multi-party computations (to process the data). In this way, `Enigma` "removes the need for a trusted third party, enabling autonomous control of personal data" [ZNP15]. `ControlChain` further distributes this management on four blockchains—namely (1) one for storing contextual information, (2) one for storing public credentials and entity relationships, (3) one for storing the authorization policies, and (4) one for logging and auditing

data accesses [PGD17]. Authorization decisions can then be made based on contextual information and the existing entity relationship. Similar to `Enigma`, the latter of the three Blockchain-based approaches also decouples the control and data planes [Sha+17]. The main difference is that the *Blockchain-based audible store* works with data streams: permissions are per stream; each data stream is chunked at pre-defined lengths and end-to-end (i.e., users ↔ AI services) encrypted. However, the former two of these approaches are neither implemented nor evaluated. In general, the genuine Blockchain concept is designed to make data globally accessible, which contradicts data protection – while data integrity can be guaranteed, data confidentiality cannot (e.g., data can be leaked as users interact with blockchains) [Hua+21].

The following approaches primarily propose environments to ensure *privacy-preserving processing*. `SWYSWYK` (*share what you see with who you know*) "allows each user to physically visualize the net effects of sharing rules" on the personal data management [TAP17]. With isolated and secure execution environments, `SWYSWYK` can enforce the sharing policies and ensure data confidentiality and integrity – but untrusted code can still leak data. Also, the system is only used locally. `FlowFence` requires providers to split their AI services into (i) a set of 'quarantined modules' that operates on sensitive data and (ii) code that orchestrates the execution by chaining the former [Fer+16]. `FlowFence` achieves a high level of protection but, on the other hand, introduces a performance overhead and high platform-specific dependencies. A completely new direction of data ecosystem is given by `Data Cooperatives`, which refer to "the voluntary collaborative pooling by individuals of their personal data for the benefit of the membership of the group or community" [HP19]. Specifically, individuals with their data stores can become members of a data cooperative—the legal entity in charge. External entities (called *queriers*, e.g., AI services) can then interact with such 'cooperatives' and query data or execute algorithms on them. Despite the many advantages of this new data ecosystem, `Data Cooperatives` are neither implemented nor evaluated, and focus on the common good rather than on providing AI services to individuals.

3.2.2.2 *Personal Data Stores*

In contrast to traditional solutions (e.g., Google Drive, Microsoft OneDrive), which are hosted in the provider's cloud [WW14], and self-hosted solutions (e.g., ownCloud), where users can keep their own data storage at home under their control [Nar+12], so-called *personal data stores* (PDS) advocate the golden mean [VO14]: they introduce protected data stores and managed application execution environments, overcoming data confidentiality issues of standalone access control mechanisms. Various PDS implementations have been proposed with their individual pros and cons; the most relevant ones (in terms of being highlighted in related work discussions and surveys) are discussed and evaluated in the following.

PDV (*personal data vault*), presented in 2010, is one of the first PDS implementations, only supporting specific data types (e.g., location) [Mun+10]. Nevertheless, PDV demonstrates the decoupling of collection and storage from the sharing of personal data: this data is selectively filtered by the user before being passed on to AI services – but the usability remains quite low. P3 enables two-party secure photo sharing by splitting a photo into two parts: a public part (containing the volume of the photo needed for server-side scaling and mass storing), and a secure private part (only containing the meaningful, privacy-sensitive information of the photo) that only the selected recipients can view [RGO13]. However, P3 is limited to one specific data type only, i.e., JPEG-compliant photos. πBox is more general, supporting arbitrary data types and even any AI algorithm [Lee+13]. πBox addresses the issue of untrusted AI services by sandboxing both the local and the remote parts of them, shifting the data protection responsibilities to the trusted platform. Personal data is stored in per-sandbox, per-user stores in the cloud with a sharing channel between them. Besides all advantages, the AI services must use platform-specific programming. OMS (*open mustard seed*) presents a visionary framework that should give individuals "control over their own digital identity in a secure, transparent and accountable way" by introducing so-called *Trusted Compute Cells* (TCC) [HDC14]. Each TCC is a network of virtual resources under control of a registry to integrate trusted *group-based* functionalities – but OMS has limited support for AI services, and is neither implemented nor evaluated.

Other important contributions in this category are openPDS [Mon+14; YGR17], Databox [Cha+15; Cra+16; Per+17], and Solid Pod [Sam+16]. The former newly introduces so-called *SafeAnswers*: parts of the code run locally in a secure environment of the PDS and have full access to personal data; only the aggregated results are then returned. Although openPDS achieves greater control over user data, it has a high platform dependency and limited support for AI algorithms due to the necessity of pre-approved queries. Also, the confidentiality of the data cannot be guaranteed, as (aggregated) data leaves the user territory. Databox, similar to πBox, overcomes this data confidentiality issue by using trusted sandboxes to establish connected local and virtual environments for 'divided' applications [Mor+16]. Further, a mix of a central and per-app data stores allows only authorized applications to access personal data. However, AI services in Databox must partly rely on platform-specific programming, which limits their flexibility; a proof-of-concept prototype has also not yet been evaluated. The latter—Solid Pod—allows users to have multiple data stores (called *pods*) from different providers: users can control access to their data and have the ability to switch between applications at any time. However, Solid Pod is designed for social and web apps; personal raw data needs to be shared to a certain extent. A proof-of-concept implementation has been initiated but not evaluated.

Table 3.1: Summary of data protection approaches *at management level* discussed in the context of AI services

	Data Protection			*Runtime*			*Applicability*				
Approach	Authentication	Authorization	Data Confidentiality (*observed/inferred/metadata*)	Privacy-preserving Processing	Data Store (*central/per-app*)	Low Performance Overhead	Support of Any Data Type	Support of Any AI Algorithm	Proof-of-Concept Prototype (*implemented/evaluated*)	Low Platform-specific Dependence	Cross-site Deployment/Use (*local/nearby/cloud*)
AC & Perm. Mgmt. (9)	3	9	3/3/3	3	8/1	9	9	9	5/3	6	8/1/6
`pawS` [Lan02]	○	●	○/○/○	○	◐/○	●	●	●	●/○	○	◐/○/★
AC for Enc. Data [Goy+06]	○	●	○/○/○	○	●/○	●	●	●	○/○	●	○/○/★
`SemaDroid` [XZ15]	○	●	○/○/○	○	○/○	●	◐	●	●/○	○	★/○/○
`Enigma` [ZNP15]	◐	●	○/○/○	○	●/○	◐	●	●	○/○	●	☆/○/☆
`FlowFence` [Fer+16]	○	◐	●/●/◐	●	◐/◐	◐	●	●	●/●	○	★/○/◐
`ControlChain` [PGD17]	○	●	○/○/○	○	●/○	◐	●	●	○/○	●	★/○/○
BC Audit. Store [Sha+17]	●	●	○/○/○	○	◐/○	◐	●	●	◐/◐	●	★/◐/◐
`SWYSWYK` [TAP17]	○	●	●/●/◐	●	●/○	●	●	●	◐/◐	◐	★/○/○
Data Cooperatives [HP19]	●	●	◐/◐/◐	◐	●/○	●	●	◐	○/○	●	◐/○/★
Personal Data Stores (7)	6	6	6/4/4	6	6/5	7	5	7	6/3	3	7/1/7
`PDV` [Mun+10]	●	●	◐/○/○	○	●/○	●	○	●	●/●	○	◐/○/★
`P3` [RGO13]	○	○	◐/○/○	●	●/○	◐	○	◐	●/●	○	◐/○/★
π`Box` [Lee+13]	●	◐	●/●/◐	●	◐/●	◐	●	●	●/●	◐	●/○/★
`OMS` [HDC14]	●	●	◐/◐/◐	●	○/◐	◐	●	●	○/○	◐	★/○/◐
`openPDS` [Mon+14]	◐	●	●/◐/◐	●	●/◐	◐	●	◐	◐/○	○	★/○/◐
`Databox` [Cha+15]	◐	●	●/●/●	●	●/●	◐	●	●	●/○	○	★/◐/●
`Solid Pod` [Sam+16]	●	●	○/○/○	◐	●/●	●	●	●	◐/○	●	★/○/◐
Total Count (*16*)	9	15	9/7/7	9	14/6	16	14	16	11/6	9	15/2/13

Note: There may be approaches that address several aspects and thus fall into more than one category or overlap with other categories (see Section 3.1). In such cases, the approaches are assigned to the categories based on the primary protection goal or challenged addressed.

Encoding: **fulfillment of requirements** (see Section 3.2.1): ●–fulfilled, ◐–partially fulfilled, ○–little or not fulfilled; **deployment**: ★–base deployment. The numbers in parentheses indicate how many approaches are considered (per category or in total). The numbers per requirement column indicate how many of these approaches fulfill this requirement (per category or in total) – partial fulfillment is also counted.

Abbreviations used in the table: AC–access control; BC–Blockchain; enc.–encrypted; perm. mgmt.–permission management

3.2.3 *Open Challenges*

Table 3.1 provides a summary of data protection approaches *at management level*. Most of the data protection mechanisms discussed in this category are presented in the context of generic systems and not specifically for AI services, which can limit their usefulness or applicability (e.g., if AI services cannot make use of the resulting accessible data, as may be the case with `openPDS`: only aggregated data from pre-approved queries is accessible in a predefined way.). Also, not all of the proposed concepts were implemented, and even fewer were evaluated with a proof-of-concept prototype. We can further see that none of the presented approaches can *fully* fulfill all of the category-specific requirements. In particular, access control and permission management approaches are not sufficient to ensure the confidentiality of data, as authorized service entities ultimately gain full access to (unmodified) user data. On the other hand, personal data stores complement authentication mechanisms and introduce privacy-preserving processing environments – both local and virtual (e.g., `Databox`). Only, the latter enable the confidentiality of data in most approaches (e.g., `FlowFence`), as AI services only access personal data locally within a sandboxed environment.

Combining both types of data protection mechanisms to exploit their respective advantages is very promising but also an open challenge due to performance overhead and/or platform-specific dependencies. Another open challenge is the (ad-hoc) deployment of distributed (possibly latency-demanding) AI services on nearby devices, which is not supported by almost all approaches with the same protection level. However, the most critical point is that approaches in this category require the assumption that the underlying (possibly third-party) systems/devices are fully trusted. This assumption does not always hold in real-world settings (e.g., in open decentralized IoT environments). If this assumption does not hold true, this could have serious consequences for the *confidentiality*, *integrity*, and *availability* of user data and AI services.

3.3 PROTECTION AT SYSTEM LEVEL

While the approaches of the previous category assume trust in the underlying (possibly third-party) systems/devices, this category contains approaches that reliably solve this potential trust issue *at system level* (see Figure 3.1, ②).

3.3.1 *Specific Requirements*

This section first establishes category-specific requirements. Similar to the previous category, the requirements are divided into the following three groups: *data/code protection* aspects, *performance* aspects, and *applicability* aspects. The group concerning data and code protection aspects, which have a direct impact on user privacy and providers' IP rights, contains the following requirements:

– *Integrity (code/data):* Protection mechanisms should be able to ensure the accuracy and consistency of data and code—including the proprietary AI

algorithms/models of providers—over their lifecycles, i.e., unauthorized or untrusted service entities should not be able to modify or tamper user data and the underlying provider's code of AI services.

- *Data confidentiality (code/data):* Protection mechanisms should protect personal data – both observed and inferred, as well as metadata – and code (including protected AI algorithms/ models) against disclosure, theft, and unintentional, unlawful or unauthorized access. This thesis argues that when personal data leaves the user territory or code leaves the provider territory without appropriate modifications (e.g., encryption), it cannot be reliably kept confidential.
- *Protection against untrusted code:* protection mechanisms should be resistant against untrusted code/ AI services, i.e., code that processes user data should not be able to disclose this data.

The group concerning the system performance aspects, which have a direct impact on the user experience, contains the following requirements:

- *Low performance overhead:* Additional or complementary data protection mechanisms should not affect the performance of AI services or entail only a low overhead.
- *Initialization optimizations:* Protection mechanisms should optimize their initialization overhead (if any), so that the user experience of AI services is not affected.
- *Runtime optimizations:* Protection mechanisms should optimize their runtime overhead (if any), so that the user experience of AI services is not affected.

The group concerning the applicability aspects, which have a direct impact on whether and how simply a provider can apply the approach to its AI services, contains the following requirements:

- *Data streaming (local/1-/n-hop):* Data protection mechanisms should be designed to support continuous and confidential data streaming on local devices (e.g., mobile device) and to 1-/n-hop nearby IoT/edge devices (e.g., public displays[8] [Mik+19; Müh+20]).
- *Support of any AI algorithm:* Protection mechanisms should be designed to support all underlying AI algorithms, so that providers can easily deploy their unmodified AI services.
- *Support of any use case:* Protection mechanisms should be designed to support arbitrary use cases, so that providers are not restricted in their AI services.
- *Support for ad-hoc deployment:* Protection mechanisms should support ad-hoc personalization of AI services on nearby devices, i.e., AI services should be deployed there with the same level of protection, and personal data should be transferred there confidentially.

[8] https://petras-iot.org/project/displays-and-sensors-on-smart-campuses-dissc (ret. 06/30/2021)

This section does not consider *availability* of the AI services, as untrusted systems/devices can easily terminate all processes at any time – this issue and possible solutions are further discussed in Section 3.3.3.

3.3.2 *Protection Approaches*

According to the underlying mechanisms, the different data protection approaches at system level are classified into the following two sub-categories.

Software-based Virtualization

To achieve trust *at system level*, early approaches rely on *privilege software enhancements*. For instance, `InkTag` extends "a standard hypervisor to monitor the untrusted OS using paravirtualized device drivers and virtualization hardware" [Hof+13]. In this way, trusted AI services can be run in high-assurance processes, which are isolated from the untrusted OS, to protect code, data, and control flow. While `InkTag` provides a high level of protection against the OS, such protection is not sufficient against side-channel attacks; it also requires customized hypervisors. `Virtual Ghost` integrates a hardware abstraction layer between the kernel and the hardware to provide a set of trusted services to both the kernel (for manipulating the hardware) and the applications (e.g., memory encryption, key management). By this means, `Virtual Ghost` outperforms `InkTag` in several benchmarks [CDA14] – but it requires recompiling the entire OS (including its kernel). `CQSTR` [Zha+16] allows users to release their data in so-called data buckets and send a reference to the service provider; services are then released to a group of tenant *virtual machine* (VM) instances with restricted networking capabilities but with access to the respective data bucket; the output is returned to the users. However, `CQSTR` requires the cloud provider as the root of trust – similar to proprietary cloud-based solutions like *Microsoft Azure Confidential*[9].

Another approach in this category, `MiniBox`, combines two mechanisms, proposing a two-way sandbox: (1) a hypervisor-based memory isolation mechanism to protect AI services from the OS using TrustVisor; (2) a one-way sandbox to protect the OS and user data from untrusted AI services using Google Native Client (*NaCl*) [Li+14]. However, `MiniBox` assumes a trusted hypervisor, i.e., it cannot provide protection against attackers who have control over the hypervisor.

Hardware-based Virtualization

Only complex *hardware enhancements* can provide such protection and achieve 'secure' memory isolations defining a trusted boundary, but they required specialized hardware in the early days – a survey of early memory encryption approaches can be found in [HT14].

[9] https://azure.microsoft.com/en-us/solutions/confidential-compute (retrieved 06/30/2021)

A game changer in this category was the release of SGX (*Software Guard Extensions*) in 2015 with the sixth-generation Intel Core microprocessors [CD16]; first prototypes and emulations of SGX were already available since 2014. With SGX, required *hardware enhancements* for trusted computing got widely available, even on commodity processors. Specifically, SGX uses hardware-protected memory regions (so-called *enclaves*) to execute code isolated from any other code in the system including privileged software (e.g., OS, BIOS, hypervisors)—i.e., only code running in an enclave can access data in this enclave [CD16]. Using *remote attestation*, the desired code can be proven to actually run "securely and unmodified within the enclave of a particular device" [Zhe+17; Abe+16].

Early approaches relying on SGX are `Haven` [BPH14] and `VC3` [Sch+15b]. The former (`Haven`) provides a two-way protection: (1) a trusted execution environment (*TEE*) through an enclave to run and protect unmodified apps from the host; (2) conversely, a secure isolation container with no access to traditional OS services to protect the host from untrusted AI services. The latter (`VC3`) uses the enclave concept for protecting user data in distributed *MapReduce* computations in the cloud. Both can truly ensure the integrity and confidentiality of code and data, even if the underlying system is not trusted. However, `VC3` does not support arbitrary AI algorithms and does not protect against untrusted code. Both approaches (`Haven` and `VC3`) introduce a considerable performance overhead during initialization and at runtime, and thus, they are suitable for an ad-hoc deployment to a limited extent.

The following approaches are for the most part only a 'variant' of these two approaches, improving on certain aspects. Therefore, we will only briefly discuss the novel features. For instance, `SCONE` combines container-based virtualization (e.g., LXC, Docker) with enclaves to increase the integrity and confidentiality of the former and its corresponding use cases [Arn+16] – but it can only be used locally, as remote attestation is not supported. `Graphene-SGX` combines *Graphene*—a library OS, similar to unikernels—with enclaves, providing integrity support for dynamically-loaded libraries and secure multi-processes [TPV17]. `Ryoan` enables distributed trusted sandboxes—relying on *NaCl* that confines data processing modules—over trusted hardware using enclaves [Hun+16] – but `Ryoan` is not designed for stream processing, only supporting request-oriented data models. `Opaque` enables distributed enclave-protected data analytics on the cluster-computing framework *Spark* from Apache [Zhe+17].

Only some approaches in this category specifically consider the nature of AI services and their underlying algorithms. For instance, `Chiron` distinguishes two types of enclaves: (1) several concurrent *training enclaves* and (2) a *parameter enclave* for exchanging AI model parameters between the former. In each training enclave, a service provider can run its AI models with a standard ML toolchain and access user data within an extended `Ryoan` sandbox [Hun+18] – both user data and AI models are kept confidential from other parties but at the expense of performance. To reduce the inherent performance overhead while still providing *secure data analysis*, Chandra et al. incorporate data randomization

mechanisms [Cha+17]. While this mechanism shows higher computational efficiency and is more resistant to side-channel attacks, it adds noise to the data—resulting in less usefulness of this data for AI services. Ohrimenko et al. propose a *multi-party ML* algorithm that is *data-oblivious*, i.e., attackers interacting with it and observing these interactions learn nothing except possibly the public parameters [Ohr+16]. Specifically, multiple users not trusting each other send their encrypted data to an enclave for a joint ML task; so-called *data-oblivious primitives* are used to careful eliminate data-dependent accesses. The latter makes the algorithm more resistant to side-channel attacks but at the expense of performance. Also, the latter two approaches are limited to specific – simple and modified – AI algorithms.

There are also some approaches in this category that are specifically designed and optimized for specific data types. For instance, `VoiceGuard` exploits the secure channel established for remote attestation to subsequently load AI algorithms/models into the enclave [Bra+18] – but `VoiceGuard` is only evaluated with voice data, introduces a high performance overhead, and does not provide runtime protection against untrusted code. `SafeKeeper`, on the other hand, is specially targeting server-side password protection, which limits its applicability for AI services considerably [Kra+18].

There is one approach in this category (namely `Ekiden`) that is particularly different from the others: it particularly addresses the *availability* and *fault tolerance* issues by combining TEEs with decentralized blockchains [Che+19]. In particular, it decouples consensus from data processing, enabling confidentiality-preserving and stateless TEEs that can be set up on several distributed nodes on demand (or in case of failure) while each persistent state is stored in the blockchain.

The latest approach in this category newly addresses issues for mobile use. `OfflineModelGuard` (`OMG`) goes for lighter *ARM TrustZone* to make mobile devices with ARM architecture accessible for the required data protection mechanisms [Bay+20]. Among other things, `OMG` also optimizes the initialization steps by caching encrypted AI models locally – but it is only designed and evaluated in the context of speech processing.

3.3.3 *Open Challenges*

Table 3.2 provides a summary of data protection approaches *at system level*. These approaches can actually ensure *confidentiality* and *integrity* of code and user data (two of the three properties of the CIA triad) by addressing the trust issue of (third-party) systems. On the other hand, we can see that none of the presented approaches can fulfill *all* of the category-specific requirements. In particular, most approaches rely on SGX hardware support to address the trust issue, which has an inherent performance overhead at setup and runtime. While first approaches—such as `OMG` [Bay+20]—already

Table 3.2: Summary of data protection approaches *at system level* discussed in the context of AI services

	Data/Code Protection			*Performance*			*Applicability*			
Approach	Integrity (*code/data*)	Confidentiality (*code/data*)	Protection Against Untrusted Code	Low Performance Overhead	Initialization Optimization	Runtime Optimization	Data Streaming (*local/1-/n-hop*)	Support of Any AI Algorithm	Support of Any Use Case	Support for Ad-hoc Deployment
SW-based Virtualization (*5*)	3/3	2/2	3	5	1	3	4/2/1	5	5	1
Container-based Virtualization ➢	○/○	○/○	◐	●	●	●	●/●/●	●	●	●
`InkTag` [Hof+13]➴	○/○	○/○	○	●	○	○	●/○/○	●	●	○
`Virtual Ghost` [CDA14]⅄	◐/◐	◐/◐	◐	●	○	◐	●/○/○	●	●	○
`MiniBox` [Li+14]➴≺	◐/◐	◐/◐	●	●	○	◐	●/○/○	●	●	○
`CQSTR` [Zha+16]➴	◐/◐*	○/○*	○	●	○	○	○/◐/○	●	●	○
HW-based Virtualization (*13*)	13/13	13/13	4	13	1	5	8/10/0	11	12	0
`Haven` [BPH14]≮	●/●	●/●	●	◐	○	○	◐/◐/○	●	●	○
`VC3` [Sch+15b]≮	●/●	●/●	○	◐	○	○	◐/◐/○	○	●	○
`SCONE` [Arn+16]≮➢	●/●°	●/●	◐	◐	○	○	◐/◐/○	●	●	○
`Ryoan` [Hun+16]≮	●/●	●/●	●	◐	○	○	◐/◐/○	●	●	○
Multi-party ML [Ohr+16]†≮	●/●	◐/◐	○	◐	○	◐	○/◐/○	◐	◐	○
Sec. Data Analytics [Cha+17]≮†	●/●	◐/◐	○	●	○	●	○/◐/○	◐	●	○
`Graphene-SGX` [TPV17]≮	●/◐	●/◐	○	◐	○	◐	●/○/○	●	●	○
`Opaque` [Zhe+17]≮	●/●	●/◐	○	◐	○	◐	○/◐/○	●	●	○
`Chiron` [Hun+18]†≮	●/●	●/●	●	◐	○	○	◐/◐/○	●	●	○
`VoiceGuard` [Bra+18]†≮	●/●	●/◐	○	◐	○	○	●/○/○	●	◐	○
`SafeKeeper` [Kra+18]≮	●/●	●/●	○	●	○	○	○/◐/○	○	○	○
`Ekiden` [Che+19]≮	●/●	●/◐	○	◐	○	◐	○/◐/○	●	●	○
`OMG` [Bay+20]†≮	●/●	●/◐	○	●	●	○	●/○/○	●	◐	○
Total Count (*18*)	16/16	15/15	7	18	2	8	12/12/1	16	17	1

Note: There may be approaches that address several aspects and thus fall into more than one category or overlap with other categories (see Section 3.1). In such cases, the approaches are assigned to the categories based on the primary protection goal or challenged addressed.

Encoding: **fulfillment of requirements** (see Section 3.3.1): ●–fulfilled, ◐–partially fulfilled, ○–little or not fulfilled (see *BL*–baseline); **primary development context**: †–specially designed for AI algorithms; **trust anchor**: ∗–third-party system/platform; **remote attestation**: °–not supported; **HW/SW prerequisites**: ≮–TEE (e.g., SGX, TrustZone), ≺–sandbox, ➢–container, ‡–PKI, ➴–VM/hypervisor, ⅄–hardware abstraction layer (recompiling of the OS incl. kernel required). The numbers in parentheses indicate how many approaches are considered (per category or in total). The numbers per requirement column indicate how many of these approaches fulfill this requirement (per category or in total) – partial fulfillment is also counted.

Abbreviations used in the table: SW–software; HW–hardware; sec.–secure; VM–virtual machine

optimize the setup phase by securely preloading AI services/models, a *proactive deployment* of AI services and *runtime optimizations* are necessary to better support or enable latency-demanding and real-time use cases. Examples include use cases in which users are on the move and have short contact times to device-bound AI services (e.g., located on drones or in the infrastructure [Müh+20]). In such use cases, a reliable, fast and zero-configuration *ad-hoc deployment* of AI services still represents an open challenge. However, addressing these challenges should not be at the expense of supporting different AI algorithms. Indeed, protection approaches should be *specifically designed for AI services* to reduce the attack surface and increase the potential for optimization.

Untrusted code or AI services inside enclaves can endanger data confidentiality. Only a few approaches (e.g., `Ryoan` [Hun+16]) include protection mechanisms against such data leaks, whereby approaches incorporating an additional sandbox are the most promising. Data leakage detection mechanisms that verify the contextual integrity of data (e.g., `VACCINE` [Shv+19]) can further complement them. Another open challenge is the remaining property of the CIA triad, *availability*, which cannot be guaranteed in most of the discussed approaches, since a malicious/untrusted hosting system can terminate an enclave at any time. Only one approach—namely `Ekiden` [Che+19]—addresses this availability issue by maintaining redundant and decentralized stateless processing. However, there is still a conflict between the goals: low performance overhead and high availability.

On a side note, a general challenge with such trusted computing approaches is the coping with side-channel attacks [Ole+18; Bra+19] – this issue is a subject of intensive ongoing research in the security community and is beyond the scope of this thesis.

3.4 PROTECTION AT AI LEVEL

Approaches of the previous two categories ensure the protection of user data against *unauthorized* access by AI services within the system (see Section 3.2) and the underlying system itself (see Section 3.3). We now look at approaches that enable the protection of user data *at AI level* (see Figure 3.1, ③) or, more precisely, against its use by *authorized* service providers that goes beyond the intended scope of the AI service.

3.4.1 *Specific Requirements*

This section first establishes category-specific requirements. Similar to the previous categories, the requirements are divided into the following three groups: *data protection* aspects, *personalization* aspects, and *applicability* aspects. The group concerning data protection aspects, which have a direct impact on user privacy, contains the following requirements:

– *Integrity (data):* Protection mechanisms should ensure the accuracy and consistency of personal data along their lifecycle.

– *Data confidentiality (observed/inferred/metadata):* Protection mechanisms should protect personal data – both observed and inferred, as well as metadata – against disclosure, theft, and unintentional, unlawful or unauthorized access. This thesis follows the obvious reasoning that when personal data leaves the user territory without appropriate modifications (e.g., encryption), it cannot be reliably kept confidential.

The group concerning personalization aspects of AI services, which has a direct impact on the user experience and the resulting benefit to users, contains the following requirements:

– *Performance (w.r.t. accuracy):* Protection mechanisms should not negatively affect the resulting performance of AI services, e.g., in terms of accuracy.

– *Personalization capabilities:* Protection mechanisms should continue to provide AI services with access to sufficient and accurate personal data to enable them to adapt to the user.

– *Low personal data involvement:* Data-protecting AI services should require less personal data, thus minimizing (i) the risk of leaking sensitive data and (ii) the inherent cold-start problem.

– *Low labeling effort:* Data-protecting AI services should require the user to label less personal data, thus reducing the burden on the user and improving the user experience.

– *Low local resource usage:* Data-protecting AI services should save local resources as far as possible, thus reducing the burden on the personal devices and improving the user experience.

The former two refer to the *usefulness* of personal data; the latter three refer to the *efficiency* of personalization. It is important to note that the main challenge is to find the best tradeoff between data protection aspects, the usefulness of personal data, and the efficiency of the personalization process (see Section 2.2).

In addition, the approach should still be applicable to AI services. The group relating to applicability aspects contains the following identified requirements:

– *Support of any data type:* Protection mechanisms should be designed to support all types of data, so that providers are not restricted in their AI services.

– *Support of multiple AI algorithm:* Protection mechanisms should be designed to support not just one AI algorithm but multiple underlying AI algorithms. This increases the applicability of the protection mechanisms for the providers to their AI services.

– *Low algorithm-specific dependence:* Protection mechanisms should be designed in such a way that AI services do not need to integrate specific algorithms.

– *Easy deployment and integration:* Protection mechanisms should be easy for providers to deploy and integrate into their AI services .

– *General model learning/ improvement capabilities:* Protection mechanisms should be designed to support learning and improving the general model, thereby alleviating the cold-start problem.

3.4.2 *Protection Approaches*

The different protection approaches *at AI level* are classified according to the following *four* specific data handling techniques that enhance user privacy.

3.4.2.1 *Data-modifying Approaches*

Approaches in this category modify or sanitize user data so that it cannot be linked to specific individuals—leading to an inherent conflict between both goals: *data protection* and *usefulness* of personal data. A key concept is *k-anonymity* that addresses the risk of re-identification of individuals within a dataset [Agg+05]. *k-anonymity* can be used as a quality measure for privacy guarantees: data for an individual contained in the dataset cannot be distinguished from at least *k*-1 individuals whose data also appears in the dataset. This means that a k-anoymized dataset has the property that each data record is similar to at least *k*-1 other records. To achieve *k*-anonymity, some of the record entries within the dataset are either suppressed or generalized (e.g., an age of 31 could be changed to the age range of 30-35) [Agg+05]. Using *k-anonymity*, for instance, Gedik et al. proposed an approach in 2007 for *protecting location privacy*, allowing users to specify *k* according to their individual privacy preferences [GL08]. Although "the goal is to lose as little information as possible while ensuring that the" dataset is *k*-anonymous, this comes at the expense of the usefulness of personal data for AI services [Agg+05]. Moreover, it has been shown that such anonymization techniques like *k-anonymity* and its variants like *l-diversity* [Mac+07] or *t-closeness* [LLV07] are vulnerable to composition attacks [Kan+18] – surveys related to such early anonymization approaches can be found in [AS00; AP08].

Approaches in this category that have been highlighted in the last decade are now briefly surveyed. In particular, obfuscation- or perturbation-based approaches such as *differential privacy* were prevailing [JLE14]. In short, *differential privacy* [Dwo+06] mathematically guarantees that, for instance, the output of a query is insensitive to the presence or absence of data about an individual in a dataset [XYW19]. The *privacy loss* at a differential change in a dataset can be measured by the privacy parameter ϵ (also called *privacy budget*) – the smaller the value, the better the privacy protection but the larger the perturbation noise. `RAPPOR` from Google is an example that enables differential privacy in a practical (real-world) setting, allowing to collect statistics from end-users (clients) with strong privacy protection using randomized responses, eliminating the need for a trusted third party [EPK14]. The `SuLQ` (*sub-linear queries*) framework

further modifies the privacy analysis to real-valued functions and arbitrary data types, achieving *sub-linear* computational performance with (slightly) noisy replies [Blu+05]. To further reduce the perturbation errors introduced by differential privacy mechanisms, Fan et al. propose two sampling-based methods—namely *simple random algorithm* and *hand-picked algorithm*—to obtain a subset of individual data for *privacy-preserving data analytics* [FJ15]. Xu et al. and Abadi et al. enable the use of differential privacy in *logistic regression* [XYW19] and *deep learning* [Aba+16] respectively – but both are limited to their specific AI algorithms.

Other approaches in this category with less restrictive assumptions than those for differential privacy are `Incognito`, `Obfuscation At-source`, and `Pickle`. The former two are focused on obfuscating web data [Mas+18] and location data, respectively [Kan+18]. While these approaches achieve a good data protection, they are limited to certain types of data, which restricts their range of use. The latter (`Pickle`), in contrast, is more generic: it works in two phases—namely a *regression phase* and a *training phase*—for privacy-preserving collaborative learning [Liu+12]. In the regression phase, each user u_i first sends summary statistics (mean and covariance matrix) derived from a fixed small number of the user's training feature vectors to the cloud. Based on these statistics and an equally-weighted Gaussian mixture model, `Pickle` randomly generates so-called public feature vectors. Each user u_i then perturbs a set of these public feature vectors by a private transformation matrix M_{p_i}; this way, `Pickle` can learn *approximation* functions f_μ to calculate the desired mathematical relationships between the (original and perturbed) feature vectors using regression methods. In the training phase, users can then send their M_{p_i}-perturbed training feature vectors with labels to the cloud; there, a collaborative model is learned, taking f_μ into account (to compensate the distortion introduced by perturbation).

All these data-modifying approaches are suitable for learning a general model in a more or less privacy-friendly way. By design, these approaches are bound to exhibiting comparatively low usefulness for personalized AI services.

3.4.2.2 *Data-encrypting Approaches*

This category comprises protection approaches that work with encrypted user data, ensuring *integrity* and *confidentiality* when sharing data. In particular, two complementary encryption techniques shape this category, namely *homomorphic encryption* [Gen09] and *secure multi-party computation* (SMC) [Pin02]. The former "makes it possible to analyze or manipulate encrypted data without revealing the data" [Mar19] – but low computational efficiency and limited operations limit its applicability. The latter is a cryptographic protocol that enables secure and private computations on distributed data without ever revealing or moving them outside the territory of the parties involved – but SMC entails a high communication and computational overhead.

This section further briefly discusses relevant approaches for AI services, many of which are based on the two techniques above. For instance, Barni et al. propose a hybrid protocol based on a combination of homomorphic encryption and garbled circuits to classify encrypted electrocardiogram signals from users [Bar+11]. Another approach, `CryptoImg`, relying on homomorphic encryption allows processing (e.g., image adjustment, spatial filtering, edge sharpening) on encrypted images [Zia+16]. However, both approaches are limited to specific data types and AI algorithms. `ML Confidential` and `CryptoNets` are more general, working with different data types but only with specific AI algorithms. The former proposes an homomorphic encryption based confidential protocol for AI tasks and develops appropriate confidential AI algorithms for binary classification based on their polynomial approximations [GLN12]. The latter further demonstrates the use of homomorphic encryption with trained neural networks [Dow+16] – but efficiency is still challenging for both.

Other approaches in this category aim at the collaborative learning of a general model. For instance, `SecureML` [MZ17] proposes protocols for specific AI algorithms using SMC and the stochastic gradient descent method—often used to train AI models, including neural networks. Specifically, `SecureML` distributes user data among two non-colluding servers, which then train various AI models on the joint data using SMC. Also using SMC but with one trusted server only, Bonawitz et al. propose a *practical secure aggregation* mechanism that allows to compute sums of model parameter updates from local AI models of individuals (see `Federated Learning`, p. 50) in a secure and more efficient way than before [Bon+17]. Last but not least, `DeepSecure` uses the garbled circuits protocol to securely perform scalable deep learning execution over distributed data from individuals, assuming an honest-but-curious adversary model [RRK18] – the efficiency is maintained. However, the latter three approaches are not easy to deploy and integrate, and getting a high accuracy is still challenging.

3.4.2.3 *Data-minimizing Approaches*

Approaches in this category aim to increase efficiency by minimizing the amount of personal data required. Depending on the setup, the current practice of *general model training* does not require personal data of individuals during training – if any are required, then only during the inference phase. This practice usually relies on voluntary data, and the training is typically performed in the cloud. While this practice achieves high efficiency (low to no burden on local resources and users) and is easy to implement, the resulting general model may have low accuracy compared to locally-trained (personal) models, as it works well for many users but not for all – we use this practice as the *baseline* for this category.

To address efficiency issues (in terms of local resource use), the first type of approaches in this category proposes the *partitioning of AI algorithms*. For instance, `Neurosurgeon` is an approach that splits the training of neural networks between the cloud and the users with layer granularity; it further identifies the optimal points for this split, taking into account the latency and energy

consumption of personal devices [Kan+17]. Similarly, Osia et al. propose *hybrid deep learning*, in which the first layers of a layer-separated, pre-trained Siamese neural network are trained locally and the output (intermediate layer) is sent to the cloud share to complement the remaining layers [Osi+17]. However, both approaches still require labeled data, which results in a cold-start problem for new users.

The next type of approaches relates to *collaborative learning* of a shared model. For instance, Shokri et al. propose a practical system for *privacy-preserving DL* [SS15]: it can jointly learn a neural network among multiple users by running the stochastic gradient descent method with global parameters on local data and selectively updating them back. Probably the best known approach in this category is `Federated Learning` from Google [AI19; Har+19; Yan+19]: each user learns a personal model or adapts a general model locally; the provider then learns or improves this general model by aggregating locally-computed model updates (e.g., weights) using a newly-introduced *averaging* method [McM+17]. However, the average accuracy of federated learning across different users remains similar to that of the baseline, but it does not require raw volunteered data. An extended, more practical version of federated learning represents `When Edge Meets Learning` [Wan+18c], allowing such distributed federated learning algorithms to be adapted for resource-limited devices.

To overcome the low effectiveness of a general model and the cold-start problem of personalized models for new users, *community-based approaches* form an appropriate tradeoff. For instance, `CSN` [Lan+11; ALC12] incorporates three kinds of inter-person similarity measurements—namely physical, lifestyle, and sensor data similarities—into the cloud-based training process. In particular, `CSN` builds a personalized model for a user, using a combination of labeled data of this user and other 'similar' users. However, `CSN` cannot guarantee integrity and confidentiality, as personal data leaves the user territory and is stored in 'data silos' within the cloud.

3.4.2.4 *Data-confining Approaches*

This category comprises AI approaches that do not require sharing personal data outside the user territory. In this way, these approaches ensure data *integrity* and *confidentiality*; they are also most *effective* in personalization due to the full access of personal data locally. On the downside, approaches in this category have low *efficiency* due to the high burden put on users and their personal devices; in addition, they do not contribute to the improvement of the general model, and personal data cannot be used for commercial purposes (e.g., advertising), thereby reducing the benefits for providers. Roughly speaking, standard AI/ML algorithms (e.g., Random Forest) that can run locally and need to re-train a personal model when new data becomes available have these properties more or less in common – we subsume these approaches under the term *local training* (*re-trained*), which represents the baseline for this category.

Table 3.3: Summary of data protection approaches *at AI level* discussed for AI services

	Data Protection		Personalization					*Applicability*				
Approach	Integrity (*data*)	Data Confidentiality (*observed/inferred/metadata*)	Performance (*w.r.t. accuracy*)	Personalization Capabilities	Low Personal Data Involvement	Low Labeling Effort	Low Local Resource Usage	Support of Any Data Type	Support of Multiple AI Algorithm	Low Algorithm-spec. Depend.	Easy deployment and Integration	GM Learn./ Improv. Capabilities
Data-modifying Approaches (7)	6	7/7/6	0	6	0	0	7	5	6	7	7	4
Protecting location privacy [GL08]	◐	◐/◐/◐	○	◐	○	○	●	○	●	●	◐	◐
`Pickle` [Liu+12]	○	◐/◐/○	○	○	○	○	●	●	●	●	●	◐
`RAPPOR` [EPK14]	◐	◐/◐/◐	○	◐	○	○	●	●	●	●	●	◐
Priv.-prsvn. data analytics [FJ15]	◐	◐/◐/◐	○	◐	○	○	●	●	●	●	●	◐
DL w/ differential privacy [Aba+16]	◐	◐/◐/◐	○	◐	○	○	●	●	○	◐	●	○
`Incognito` [Mas+18]	◐	◐/◐/◐	○	◐	○	○	●	◐	●	●	●	○
`Obfuscation At-source` [Kan+18]	◐	◐/◐/◐	○	◐	○	○	●	○	●	●	●	○
Data-encrypting Approaches (7)	7	7/7/7	5	5	7	4	7	5	3	3	4	3
Private ECG classification [Bar+11]	●	●/●/●	◐	●	●	○	◐	○	○	◐	◐	○
`ML Confidential` [GLN12]	●	●/●/●	◐	●	●	○	◐	●	○	○	◐	○
`CryptoNets` [Dow+16]	●	●/●/●	◐	●	●	○	◐	●	◐	○	◐	○
`CryptoImg` [Zia+16]	●	●/●/●	◐	●	●	●	◐	○	○	○	◐	○
`SecureML` [MZ17]	◐	◐/◐/◐	○	○	◐	◐	●	●	◐	◐	○	●
Secure aggregation [Bon+17]	●	●/●/●	○	○	◐	◐	●	●	◐	◐	○	●
`DeepSecure` [RRK18]	●	●/●/●	◐	◐	◐	◐	◐	●	○	○	○	●
Data-minimizing Approaches (7)	6	6/6/2	3-6	3	4	5	6	7	3	5	4	6
General model training (*cloud-based*)	◐	◐/◐/◐	○	○	●	●	●	●	●	●	●	◐
`CSN` [Lan+11; ALC12]	○	○/○/○	●	●	◐	◐	●	●	●	◐	◐	◐
Privacy-preserving DL [SS15]	●	●/●/◐	○-◐	○	○	◐	●	●	○	◐	○	●
`Neurosurgeon` [Kan+17]	●	●/◐/○	●	◐	○	○	◐	●	○	○	◐	○
`Hybrid Deep Learning` [Osi+17]	●	●/◐/○	●	◐	○	○	◐	●	○	○	◐	◐
`Federated Learning` [McM+17]	●	●/●/○	○-◐	○	◐	◐	○	●	○	◐	○	●
Edge Meets Learning [Wan+18c]	●	●/●/○	○-◐	○	◐	◐	◐	●	◐	◐	○	●
Data-confining Approaches (3)	3	3/3/3	3	3	2	2	1	3	3	2	3	0
Local training (*re-trained*)	●	●/●/●	●	●	○	○	○	●	●	●	●	○
Local training (*incremental*) [Ser+18]	●	●/●/●	●	●	◐	◐	○	●	◐	○	●	○
`Patching` [KF18]	●	●/●/●	●	●	◐	◐	◐	●	◐	●	●	○
Total Count (24)	22	23/23/18	11-14	17	13	11	21	20	15	17	18	13

Note: There may be approaches that address several aspects and thus fall into more than one category or overlap with other categories (see Section 3.1). In such cases, the approaches are assigned to the categories based on the primary protection goal or challenged addressed.

Encoding: **fulfillment of requirements** (see Section 3.4.1): ●–fulfilled, ◐–partially fulfilled, ○–little or not fulfilled; **performance scale** (see Section 2.1.3): ○ ≅ general model, ● ≅ local model. The numbers in parentheses indicate how many approaches are considered (per category or in total). The numbers per requirement column indicate how many of these approaches fulfill this requirement (per category or in total) – partial fulfillment is also counted.

Abbreviations used in the table: GM–general model; DL–deep learning; ECG–electrocardiogram

Other approaches in this category are mainly aimed at increasing efficiency with the same or similar usefulness of the data. In particular, *transfer* and *incremental learning* algorithms are a promising direction. The former refers to the ability to use knowledge acquired in the past to learn new tasks (with less personal data)—reducing the burden on users (e.g., lower labeling effort). The latter refers to the ability to train an existing AI model incrementally only on the basis of newly available data—reducing the burden on personal devices (e.g., lower resource use). Both can also be combined, as demonstrated by the following two exemplary approaches. Servia et al. propose a neural network architecture that is trained in the cloud and *incrementally* adapted to a user in a subsequent local personalization step by readjusting the model parameters and weights [Ser+18]. Although this approach enables deep neural networks with less personal data, it is limited to only neural network based AI algorithms. In contrast, the idea of `Patching` [KF18] is more general and thus not limited to neural networks: a general (cloud-based) 'black-box' model—that can be immutable and inscrutable—is adapted to new user data (locally) by observantly inferring and fixing error regions of this new instance space, in which the model is error-prone. In this way, `Patching` (a kind of meta-algorithm) requires less personal data and is theoretically applicable to different underlying AI/ML models – however, its applicability and usefulness have only been evaluated in a few cases. In Chapter 6, `Patching` is explained in more detail, as one contribution of this thesis is based on it.

3.4.3 *Open Challenges*

Table 3.3 provides a summary of data protection approaches *at AI level*. We can see that none of the presented approaches can fulfill all of the category-specific requirements. In particular, while data-modifying approaches have a low overhead, they have an inherent conflict between the usefulness of personal data and user privacy that remains unsolved. Data-encrypting approaches are perfectly suited to ensure confidentiality and integrity of data – but they are limited in their applicability. A combination with data-minimizing approaches is currently the more promising direction. For instance, shared model parameters/weights of locally trained models (to improve the cloud-based general model) can be securely aggregated (through data-encrypting approaches) [Bon+17]. On the other hand, to achieve high personalization accuracy while preserving user privacy, local approaches are best suited, as they have full access on personal data that never leaves the user territory – but low efficiency (i.e., labeling effort, local resource use) remains an open challenge.

All in all, a promising future way to achieve a new tradeoff between user privacy and personalization is a proper combination of multiple steps and different approaches, which still needs to be explored.

3.5 IMPLICATIONS FOR THIS THESIS

Tables 3.1-3.3 summarize all relevant data protection approaches for this thesis, showing an overall comparison of them based on the established category-specific requirements. We can say that there is *no one-size-fits-all solution* that completely fulfills all requirements for AI services. Indeed, many challenges are either studied separately, just optimizing certain aspects only, or fragmentarily by different communities, most of which are not yet interconnected. Future approaches should address all category-specific requirements, and a proper combination of these approaches across the different levels, which is not trivial, should be explored to achieve comprehensive protection (without any trust assumptions). Further, today's protection approaches are mostly either very limited to one specific data type or AI algorithm (see Section 3.4) or too generic (beyond AI), which in turn leads to performance issues (see Section 3.3). Either way, future approaches to data protection require both a *stronger specialization on AI services*, taking into account their nature to cope with the inherent performance overhead while still supporting a wide range of data types and state-of-the-art AI algorithms.

In the light of these deficits, this thesis contributes the following data protection mechanisms to each level (see Part III), addressing the open issues identified:

AT MANAGEMENT LEVEL Chapter 4 presents `PrivAI`, a privacy-by-design platform that follows the data decentralization paradigm and relies on a data-confining PDS concept (`DC-PDS`) with integrated design and runtime support. The former ensures confidentiality and integrity of the user data by confining data-accessing AI services (or parts of them) locally. The latter addresses local performance issues of such AI services by eliminating redundancies. `PrivAI` forms a solid and efficient foundation for the following building blocks.

AT SYSTEM LEVEL Chapter 5 presents `PDSProxy`, a building block for PDS-internal/-external confidential processing based on a combination of SGX enclaves and sandbox approaches. In particular, `PDSProxy` addresses the (often neglected) IP protection issue in data-confining approaches; it also enables an ad-hoc deployment on untrusted (nearby) devices by addressing the inherent initialization overhead.

AT AI LEVEL Chapter 6 presents `{P}Net`, a building block to address the cold-start problem for new users in AI services relying on supervised learning algorithms (especially if explicit labeling is required). To this end, `{P}Net` combines a data-confining approach with a new confidential community-based sharing concept; in this way, it achieves a unique personalization tradeoff between effectiveness and efficiency.

These contributed approaches are complementary to each other; their successive integration as part of this thesis provides synergistic benefits—achieving a *unique tradeoff* between personalization of AI services and user privacy while

protecting the IP rights of providers at all times.

In Part IV, this thesis further proposes new infrastructural concepts, outlining ways how the aforementioned concepts for data decentralization can be applied in the everyday life of users. In particular, this thesis shows that a close integration with two proposed (decentralized, city-wide) infrastructures (Chapter 7) can address the open availability and performance issues of data-protecting distributed AI services (Chapter 8).

Part III

PROTECTION MECHANISMS THROUGH DATA DECENTRALIZATION

Contribution Statement: This core contribution part is mostly shaped by the following publication, which synergistically *combines* and *integrates* the individual contributions of the chapters:

- Christian Meurisch, Bekir Bayrak, and Max Mühlhäuser. "Privacy-preserving AI Services Through Data Decentralization." In: *Proceedings of The Web Conference 2020*. WWW'20. **acceptance rate: 19.2%**. ACM, Apr. 2020, pp. 190–200. DOI: 10.1145/3366423.3380106

I (*Christian Meurisch*) led the process of idea generation, conceptual design, evaluation, and writing. *Bekir Bayrak* implemented the proof-of-concept prototype and conducted parts of the evaluation. *Max Mühlhäuser* provided helpful critique and comments on the conceptual design.

Note: Since the contribution above is divided among the following chapters, the (supplementary) contribution statements in the individual chapters further complement the above.

4

A PRIVACY-BY-DESIGN PLATFORM FOR AI SERVICES

The previous part (Part II) gives an introduction to personalized AI services, identifies open privacy challenges (for both, see Chapter 2), and reviews the related work on data protection (see Chapter 3). This part (Part III) presents the technical contributions of this thesis on data protection: it adds, in particular, new data protection mechanisms at each of the previously-defined levels, synergistically combining them into a *privacy-by-design platform*—termed `PrivAI` [MBM20]—for single-user AI services.

Starting with the fundamental concept, this chapter contributes to the protection against unauthorized access and use of personal data *at management level* – the categorization scheme is described in Section 3.1 (see also Figure 3.2, p. 33).

> *"[In the traditional web,] we are pointing to this location [(the cloud)] and pretending [the information] exists in only one place. And from this comes this whole monopolization that has followed...because whoever controls the location controls access to the information."*
>
> Protocol Labs, 2018 [Cor18]

With these words, Protocol Labs outlines the fundamental issue when using web-based services. Typically, service *providers* (or even third parties) have control over the 'location' in which both the service runs and the user data is stored. This leads to the loss of data control and ownership for users, as providers have the opportunity of accessing this data (e.g., to copy and use it for other purposes than originally intended) outside the *user territory*—referring to the digital space that is controlled by the user (i.e., the data subject) and cannot be bypassed or restricted by the provider, as introduced in Section 1.1.1. In general, user data leaving the user territory can only be kept confidential in encrypted or irreversible obfuscated form, which in turn limits its usefulness for AI services (see Section 3.4.2). Therefore, the user territory must include appropriate protection mechanisms (e.g., authentication, authorization, privacy-preserving execution environment) to control access to the *confidential user data* and ensure its use only for the intended purpose – all requirements identified in this category have already been introduced in detail in Section 3.2.1.

As shown in the review of related work in Section 3.2.2, none of the latest data protection approaches can fulfill the category-specific requirements listed there at the same time. In particular, conventional *access control* (e.g., DAC, MAC, TBAC [BGK11]) and *permission management* (e.g., SemaDroid [XZ15]) approaches are not sufficient to ensure the confidentiality of user data: authorized parties get access to user data, but in a way that allows them to continue using it beyond the actual authorized scope of use. *Blockchain* (*BC*)-based approaches (e.g., ControlChain [PGD17], Enigma [ZNP15]) decouple this control plane

from the data plane, ensuring data integrity and allowing data access auditing – but they cannot guarantee the confidentiality of this user data and face inherent efficiency issues; in addition, the corresponding publications often lack a proof of concept. *Personal Data Store* (*PDS*)-based approaches (e.g., openPDS [Mon+14], Databox [Mor+16]), on the other hand, provide an additional secure execution environment: data-accessing services—or at least parts of them—run locally inside it, and user data—or at least raw data—never leaves the user territory; this condition guarantees the confidentiality of (raw) user data. However, not all approaches were evaluated together with a proof-of-concept implementation. Also, efficiency is still an open challenge, as AI services must run locally to a certain degree.

In the light of these deficits, this thesis pursues a suitable tradeoff between the competing data protection, efficiency, and applicability requirements at management level. To this end, this chapter proposes an approach to data decentralization that moves and *enforces* the user data store and the execution environment for AI services in the user territory. Specifically, this chapter deals with the protection of user data against unauthorized access within the user territory, assuming that the underlying system is trusted (Chapter 5 then adds data protection mechanisms against untrusted system). The contribution of this chapter is twofold. First, this chapter contributes the data decentralization basis of `PrivAI`, which is based on the PDS concept but with a *strict data confinement policy*—termed `DC-PDS` (see Section 4.1); it includes a managed execution environment with provided runtime support through a graph-based approach (termed `openAIgraph`) to cope with the inherent efficiency issues of locally-running data-accessing AI services (see Section 4.2). Second, this chapter reports and discusses the evaluation results of a proof-of-concept (PoC) implementation of `openAIgraph`—demonstrating the unique tradeoff between the category-specific requirements (see Sections 4.3–4.5).

Contribution Statement: This chapter is based on the (2) publications below:

- Christian Meurisch, Bekir Bayrak, and Max Mühlhäuser. "AssistantGraph: An Approach for Reusable and Composable Data-driven Assistant Components." In: *Proceedings of the 2019 IEEE 43rd Annual Computer Software and Applications Conference*. COMPSAC'19. IEEE, July 2019, pp. 513–522. DOI: 10.1109/COMPSAC.2019.00079
- Christian Meurisch, Bennet Jeutter, Wladimir Schmidt, Nickolas Gündling, Benedikt Schmidt, Fabian Herrlich, and Max Mühlhäuser. "An Extensible Pervasive Platform for Large-scale Anticipatory Mobile Computing." In: *Proceedings of the 2017 IEEE 41st Annual Computer Software and Applications Conference*. COMPSAC'17. IEEE, July 2017, pp. 459–464. DOI: 10.1109/COMPSAC.2017.54

I (*Christian Meurisch*) led the process of idea generation, conceptual design, evaluation, and writing. The students *Bennet Jeutter*, *Wladimir Schmidt*, *Nickolas Gündling*, and *Bekir Bayrak* implemented the PoC prototypes; together with *Fabian Herrlich*, they conducted parts of the evaluation. *Benedikt Schmidt* and *Max Mühlhäuser* provided helpful critique and comments on the conceptual design.

4.1 DATA-CONFINING PDS CONCEPT

This section presents the data-confining PDS concept (`DC-PDS`), which constitutes the data decentralization basis of `PrivAI`. `DC-PDS` extends the PDS concept with a new privacy policy that *strictly enforces the confinement of confidential user data at any time* – see Definition 1.2.1 (p. 4). While this data decentralization concept ensures user privacy, it has two implications for AI services [Meu21b]: (1) the `DC-PDS` decouples the user data from the AI services, enabling to store and manage them separately and independently of any AI service in a decentralized way. This means that the required data are provided to the active AI services instead of AI services having to collect the data themselves separately. (2) AI services (or at least their data-accessing parts) must 'move to the data' – and not vice versa, as usual. Thus, these (parts of the) AI services get access to confidential user data only within the user territory. This, in turn, may allow users to give AI services more liberal permissions to their confidential (unmodified) data, increasing the usefulness to AI services.

To cope with these implications for AI services and be compliant with the data confinement policy above, this thesis proposes an open AI infrastructure concept (namely `OAII`). Figure 4.1 depicts the general architecture of the OAII and the respective decentralized *PDSs* of the users. Both will be explained next – the realization of a proof-of-concept prototype will be described in Section 4.3.

4.1.1 *Open AI Infrastructure*

The OAII comprises three different roles, namely the *users*, the service *providers*, and a trusted *global instance* (GI). Specifically, each user operates an own personal data store, which is hosted in the user territory – more details are given below in Section 4.1.2. The providers offer the AI services to users via the *Service Store*; they *can* also share parts of the AI services as so-called *AI modules* with other providers via the *Module Store*. The GI operates both stores. The detailed functioning will be explained in Section 4.2.

Inspired by `ENTOURAGE` [ZHK19], a privacy and security reference architecture for digital assistants (a subcategory of AI services; see Section 2.1.4), OAII establishes an X.509 Public Key Infrastructure (PKI) [Hol+11] to interconnect the parties involved: the GI acts as a trust anchor—i.e., as the trusted Certificate Authority (CA); each AI service (including AI module) and each PDS get their own (client/server) certificate issued by the GI after registering in the OAII; the GI stores all public keys of these registered entities in its trust store. This enables strong *certificate-based mutual* (*two-way*) *authentication* as follows [Cha+12]: (1) an AI service s sends a request to the user's PDS p for local resources; (2) the PDS responds with its certificate c_p; (3) the AI service retrieves the PDS public key k_p^{pu} from the GI trust store and thereby verifies c_p. The next two steps are analogous to steps (2)+(3), except for the swapped roles for mutual authentication: (4) the AI service sends its own certificate c_s to the PDS; (5) the PDS retrieves the public

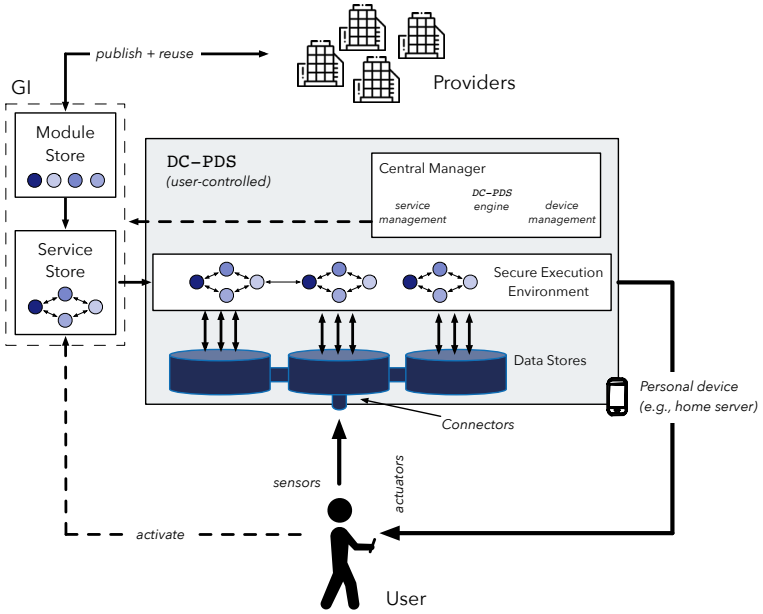


Figure 4.1: An open AI infrastructure with the respective decentralized (data-confining) personal data store of a user

key of the AI service k_s^{pu} from the GI trust store and thereby verifies c_s. Once mutual authentication is successfully completed, the AI service can access the resources of the user's PDS. For fine-grained authorization, DC-PDS employs a data flow based permission management – explained in detail in Section 4.2.2.1.

4.1.2 *The Core PDS Architecture*

Each user operates an own PDS in which the user stores *all* digital data. This PDS is located in the user territory (e.g., on a home server or a NAS) and is fully controlled by the user. In essence, each PDS consists of the following four components, namely the *connectors*, the *data stores*, a *secure execution environment*, and the *central manager*.

Connectors

A *connector* connects a sensor or an actuator to the PDS; a sensor or actuator can be physical (e.g., accelerometer, display) or virtual (e.g., activities, calendar) [SS14a]. In the case of sensors, the connector reads out the data from the sensors and passes them to the PDS; if possible, it can adjust the sensor operation, e.g., the sampling rate and the accuracy of the data. In the case of actuators, the connector translates the commands from the PDS and passes them to the actuators.

Data Stores

A *data store* ultimately stores the user data coming from the connectors or AI services; this data can be of different types (see Section 2.1.3.1). Depending on

the data type and the source, the data store has a different access policy based on the permission management proposed in Section 4.2.2.1. In short, AI services or at least the data-accessing parts of them (AI modules) can store the computation results (high-level data) in their own data store or database space—isolated from other AI services. On the other hand, they can access the user data from (i) the main data store to which they have inherited or direct permissions from the user, (ii) their own data store, and from (iii) the data stores of other AI services/modules to which they subscribe. In addition to the access layer, a PDS uses a query layer that allows the AI services/modules to define the structure of the required data, which is then returned in the same structure. [MBM20]

Secure Execution Environment

The *secure execution environment* relies on a lightweight sandboxed container runtime in the user space, i.e., it runs as a normal, unprivileged process. This environment can run each AI services and AI modules as separate and sandboxed containers: they are separate to prevent other AI modules (possibly from other providers) from accessing them; they are sandboxed to prevent malicious behavior and data leaks to the outside, which can be achieved by intercepting application system calls through the runtime. However, these locally-running AI modules or AI services may contain confidential business logic such as proprietary AI algorithms/models, which can be accessed by the user and the underlying system. This could lead to infringements of the provider's intellectual property, which would be a reason for providers to reject such decentralization approaches – to address this issue, this thesis will contribute a protection mechanism against untrusted systems or other parties in Chapter 5.

Central Manager

The *central manager* takes care of the three relevant management tasks, namely *service management*, *device management*, and the control of the secure execution environment – the latter is carried out by the so-called `DC-PDS` *engine*. The service management is responsible for managing and verifying the AI services activated by the user, i.e., it also downloads the service and associated required modules from the stores and executes them in the secure execution environment. The device management is responsible for controlling and monitoring all installed connectors to registered devices and services – for the latter, it can accept and execute predefined commands from AI services in the secure execution environment. The engine handles the AI module management for the AI services and the interconnection between them (i.e., the data and control flow) within the secure execution environment – the detailed functioning is explained next.

4.2 OPEN AI GRAPH CONCEPT

The previous section introduced the underlying data confinement concept and the general architecture. This section explains the functioning and the interplay of the architectural components. In particular, this thesis contributes

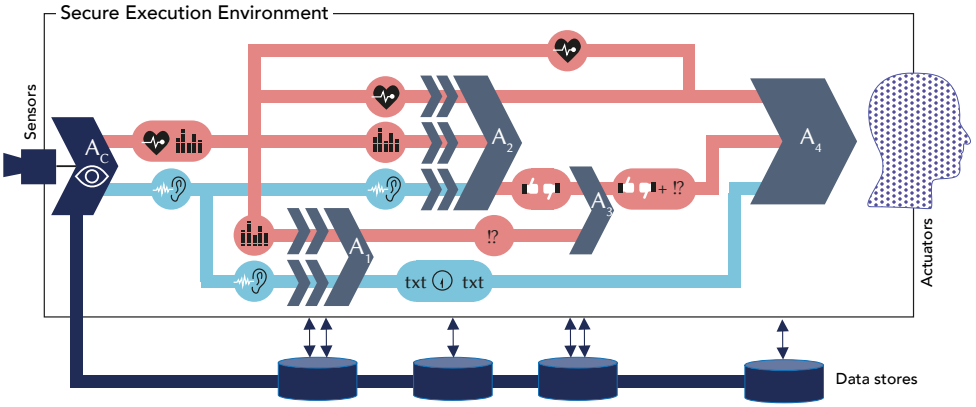


Figure 4.2: A complex application sample of an interlaced `openAIgraph` that runs in the secure execution environment of the `DC-PDS` and is managed by the *engine*—using the example of a video meeting with two input data streams (*red*, *light blue*) and individual persistent data stores (*dark blue*). The graph comprises several data-driven AI services (*A1-4 chevrons*) and their underlying AI modules (smaller *chevrons*), which are published in the OAII.

the `openAIgraph`[10], a new concept for managing reusable and composable data-driven (possibly cross-provider) AI services; it proposes a way to organize and operate AI services locally. The `DC-PDS` *engine* provides the runtime environment for this concept.

Figure 4.2 illustrates an application sample of an interlaced `openAIgraph` for an AI-supported video meeting: the user-activated AI services (A1-A4) process the respective data streams and support the meeting participants. We can see that such AI services are partly based on the same processing steps (or AI algorithms/models) on the same data. To convey the general idea of `openAIgraph`, image a simple (constructed) example in this context of a video meeting with three AI services: one blurs all (privacy-critical) objects in the scene, one highlights all these objects, and one replaces them by others. All three AI services are based on the *same* computation in the first step, namely detecting objects in the *same* input video stream. In monolith implementations, this computation is executed three times. The same is true when computation is packaged into an AI module and shared among the AI services (e.g., typical edge computing approaches would initiate one *instance* of the module for each service). For single-user AI services (as addressed in this thesis) that operate on the same user-related input data, this leads to redundancies and inefficiencies. Running only one instance of the module ('object detection') and sharing only the results ('detected objects') would be most efficient (e.g., similar to the pub/sub principle). However, AI services would then only have to subscribe on the data ('detected objects'), which can exist multiple times as a result of different computations from different providers; an AI service would have to trust on the data only regardless of the (decoupled) underlying computations.

[10]The `openAIgraph` adapts the `AssistantGraph` concept from [MBM19b], generalizing it to the concepts of this thesis.

For such *single-user environments* (e.g., within the user's PDS), with `openAIgraph`, we follow the concept of still sharing the modules among the AI services, but running only one instance of them (in case of identical modules) at a time and sharing the results only. In a larger view, `openAIgraph` can be a free and open-source software (FOSS) with which developers can quickly develop complex and customized AI services by using trusted modules (from the *Module Store* in the `OAII`) in the design and development phase. In this way, developers benefit from reusability and flexibility while the local user's PDS benefits from reduced redundancy (as only one shared instance of the same module runs at runtime) and improved efficiency. A company can also use this concept to build its internal AI services or ecosystem in this way.

4.2.1 *Designing AI Services*

As mentioned above, the `openAIgraph` concept allows developers to benefit from the reuse of shared (already existing) AI modules in the design and development stage – we will now explain this procedure.

4.2.1.1 *Reusing Shared AI Modules*

`openAIgraph` proposes a concept of how (data-driven) AI modules can be organized and interact with each other in order to provider runtime support, which will be described in Section 4.2.2. We will now first give a concrete example, which we will then use to explain the design process within `openAIgraph`. Staying with the video meeting example above, two sample AI services—one just blurring all privacy-sensitive objects in the scene (e.g., family photos hanging on the wall), and the other replaces very privacy-critical ones of them (e.g., account statements lying visible on the desk) by others neutral objects—rely on the same following computations or share the same two AI modules: (1) an *object detection* module can detect all object in a scene; it has the video stream (representing sensor data captured by the video camera) as input and outputs all detected objects. (2) The next module can *classify all objects in terms of their privacy sensitivity*, taking the output ('detected objects') of the previous ('object detection') module as its input – we refer to this as a data dependency edge that developers must specify when designing their AI services. Last, both AI services then use the output ('privacy-sensitive objects') of the previous module in (2) as their input in addition to the video stream; these AI services can modify the video stream, which may then be sent on to the device actuators (e.g., display) or to the other participants' devices.

Conceptually, an AI service describes these data dependencies between the required and optional AI modules and can connect to the actuators via the central manager. Optional modules are only active if the user gives the AI service the required permissions, which will be described in more detail in Section 4.2.2.1. An AI module in general contains the code (including the AI algorithms/models), the optional converter files required for ensuring version backward compatibility (see Section 4.2.1.2) and the so-called *manifest* file required for the service

management; the manifest file representing the service description comprises the following attributes:

- *UUID:* A 'universally unique ID' is required to link to the module.
- *Version number:* This attribute specifies the current version of the module.
- *Description and Categorical Classification:* This attribute group gives a short description of the AI module and the associated category; it helps developers to find a module.
- *Creator:* This attribute specifies the creator of the module; it can be relevant for a developer with regard to the trustworthiness of the creator.
- *Input/Output:* An array each specifies the required input or output data types that a code expects or outputs.
- *Data Dependencies:* If the module is based on other modules, the corresponding hierarchical data dependencies to these modules can be specified here by their UUIDs.

The final AI module is published in the *Module Store*. An AI service extends an AI module, also containing the code and a manifest file with the same attributes as above (possible with further store specific attributes relevant for the user). For example, a developer who wants to develop the above described AI service that blurs all privacy-sensitive objects in the scene proceeds as follows: (1) the developer specifies the manifest file and assembles the AI modules ('object detection' and 'classification of objects w.r.t. privacy sensitivity') needed for the service; the developer can search for these modules in the *Module Store* and specify the UUIDs of these modules in which creators the developer has trust. (2) the AI service gets as inputs (i) the outputs of the AI modules ('privacy-sensitive objects') specified accordingly in the manifest and (ii) the video stream; the developer can develop the code based on these inputs (i.e., modifying the video stream by blurring all privacy-sensitive objects) and then release the command or result (i.e., the modified video stream in this example). (3) The final AI service ('blurring privacy-sensitive objects in a video stream') is published in the *Service Store* and represents a *complete* service that users can activate in this store (e.g., via a smartphone client).

After the user activated an AI service in the *Service Store*, the central manager automatically downloads this AI service and resolves the data dependencies by downloading all AI modules (that are not already running locally) from the *Module Store*, as specified in the given manifests. The central manager deploys the AI service and all these AI modules in the secure execution environment and connects them via an asynchronous messaging system (see `DC-PDS` engine in Section 4.1.2) according to the data dependency graph – based on this graph, `openAIgraph` can provide runtime support as described in Section 4.2.2.

Figure 4.3 shows a schematic representation of a more complex sample data dependency graph with two AI services a_1 and a_2 (possibly from different

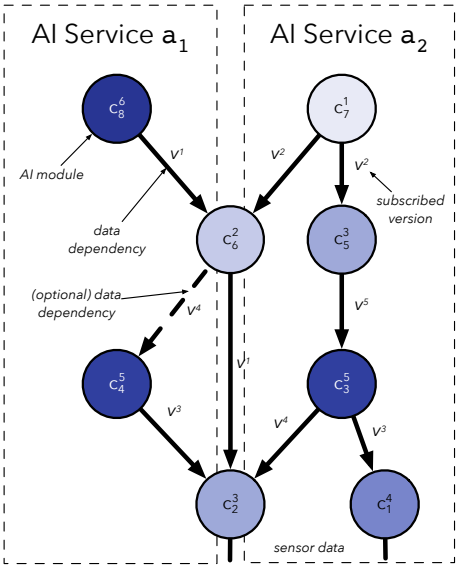


Figure 4.3: A schematic representation of a sample data dependency graph G with two AI services $\{a_1, a_2\}$ and eight AI modules $\{c_1, .., c_8\}$; the superscripted numbers in the module names indicate the currently running version (also color coded); the superscripted numbers in the edge names indicate to which versioned module a data dependency exists.

providers) and with eight AI modules $c_1 - c_8$, whereby two (c_2 and c_6, possibly also from other providers) are shared – the superscripted numbers in Figure 4.3 are the version numbers of the AI modules, which will be described in Section 4.2.1.2. This sample is used below to formally describe the graph – this basic notation introduced here will be also used in the following sections. Let's consider an AI service $a \in \mathcal{A}$ as a directed acyclic graph $G_a = (C_a, E_a)$ with a set of AI modules $C_a = \{c_{a_1}, ..., c_{a_n}\}$ (*vertices*) and data dependencies $E_a = \{e_{a_1}, ..., e_{a_m}\}$ (*edges*) between these AI modules. Each AI module $c_{a_j} \in C_a$ can have multiple inputs, collectively referred to as i_{a_j} (also corresponds to the *outgoing data dependency edges* $E_{a_j\rightarrow}$), and multiple outputs, collectively referred to as $o_{a_j} = c_{a_j}(i_{a_j})$ (also corresponds to the *incoming data dependency edges* $E_{a_j\leftarrow}$). Each data dependency edge $e_{a_{q\rightarrow p}} \in E_a$ is a tuple

$$e_{a_{q\rightarrow p}} = \{c_{a_q}, c_{a_p}, \omega_{qp} \mid c_{a_q} \in C_a, c_{a_p} \in C_a, \omega_{qp} \in \{0, 1\}\}, \tag{4.1}$$

where ω_{qp} specifies an optional (0) or required (1) directed data dependency between two AI modules, namely from c_{a_q} to c_{a_p}. Using the example of Figure 4.3, instead of running two separate AI services a_1 with $c_{a_1} = \{c_2, c_4, c_6, c_8\}$ and a_2 with $c_{a_2} = \{c_2, c_1, c_3, c_5, c_6, c_7\}$, `openAIgraph` runs only one instance per shared AI module (c_2 and c_6) and shares the result according to the data dependencies. In this way, the number of resource-intensive AI modules that need to run locally is reduced (from 10 to 8 in the example) while the number of data dependency edges may remain constant.

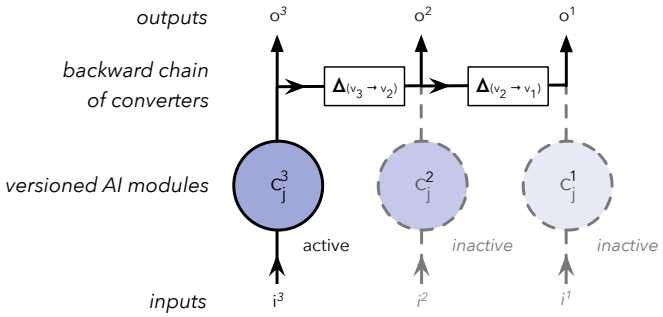


Figure 4.4: Backward chain of converters shown by the example of three versions (v_1–v_3) for one AI module c_j; only the latest version (v_3) needs to run.

4.2.1.2 *Ensuring Backward Compatibility in Versioning*

`openAIgraph` also enables providers to dynamically change their AI modules as they evolve (e.g., if the new version achieves better results and is more efficient, or if security gaps are fixed) while ensuring backward compatibility.

Problem Description

Providers may update their AI modules, publishing new versions of them to the *Module Store*. Hence, an AI module c_j can have multiple versions

$$C_j = \{c_j^1, ..., c_j^k\}. \tag{4.2}$$

Let $c_j^l \in C_j$ be an AI module c_j with the version l and incoming dependency edges from other modules $C_{j\leftarrow}^l \subset C$. Thus, a module $c_q \in C_{j\leftarrow}^l$ is adjacent to the versioned module c_j^l and particularly depends on its outcome. Let us now assume an evolution of the AI module c_j from the current version l ($l \geq 1$) with input i^l and output $o^l = c_j^l(o_l)$ to a newer version $l+1$ with input i^{l+1} and output $o^{l+1} = c_j^{l+1}(i^{l+1})$. Then all modules $C_{j\leftarrow}^l$, which have subscribed to the output of c_j^l, either have to update to the new version $c_j^{l+1} \in C_j$ (which might require an unacceptable adjustment effort for other providers) or all versions C_j of the AI module c_j should still be running (which is very resource-intensive for local environments – and especially unnecessary, as similar and possibly redundant computations are performed).

Backward Chain of Converters

To address the above-described versioning issue, `openAIgraph` includes a helpful versioning convention concept with *backward chain of converters* for each AI module. This concept is inspired by two software engineering design patterns, namely adapters and chain of responsibility [Gam95]—a related implementation to the proposed one is given by [KML06] but in the context of web service evolution. This concept can be applied in particular (i) if the outputs do not change fundamentally after an update of the AI module, e.g., both versioned AI modules still outputs scene objects, or (ii) if they can be transferred to the new format or simply (iii) if new outputs are added. In case of fundamental changes,

Algorithm 1 Algorithm of the proposed *backward chain of converters* to ensure backward compatibility from the latest version k of the module c_j back to all older versions $l \geq 1$ of the same module c_j that are still subscribed by other modules $C_{j:l\leftarrow} \subset C$ of the graph G – all required outputs are included in O

```
1:  function BackwardConverting(G, c_j, k, o^k)
2:      initialize set O
3:      c_j^k ← getVersionedModule(c_j, k)
4:      C_{j:k←} ← G.adjacentVertices_←(c_j^k)
5:      if |C_{j:k←}| > 0 then
6:          O ← O ∪ {o^k}
7:      end if
8:      if k == 1 then
9:          return O
10:     end if
11:     Δ_{c^k→c^{k-1}} ← loadConverter(c_j, k, k − 1)
12:     o^{k-1} ← Δ_{c_j^k→c_j^{k-1}}(o^k)
13:     return O ∪ BackwardConverting(G, c_j, k − i, o^{k-1})
14: end function
```

a new AI module should be created.

Figure 4.4 illustrates the functioning of a backward chain, using the example of three versioned AI modules (c_j^1, c_j^2, c_j^3) – only the latest module c_j^3 requires to run. To ensure backward compatibility over several versions, a backward chain consists of a concatenation of several converters – defined as follows:

$$\Delta_{c^k \to c^l}(o^k) = (\Delta_{c^k \to c^{k-1}} \circ .. \circ \Delta_{c^{l+1} \to c^l})(o^k). \tag{4.3}$$

In practice, providers who want to update their module c_j^{k-1} with a newer module c_j^k must take two steps: (1) publishing the new module c_j^k, which replaces the currently-running module c_j^{k-1} – this old module becomes inactive, the new one becomes active; (2) providing a *converter* $\Delta_{c_j^k \to c_j^{k-1}}(o^k) = o^{k-1}$ that converts the output o^k from the new version c_j^k to the output o^{k-1} of the next older version (c_j^{k-1}). Colloquially, a converter is a blackbox, containing code that 'translates' the outputs between two consecutively-versioned AI modules – it can simply be a 'pass-through function' (if the output format remains the same or similar) or a template processor. At runtime, whenever new data is output by any AI module, the engine executes such a backward chain of converters for it, following Algorithm 1. This algorithm detects older, still subscribed AI modules and converts the current output o^k back to all these older versions (without still running the resource-intensive AI modules).

4.2.2 *Runtime Support*

While the previous section covers the design aspect of `openAIgraph`, this section contributes mechanisms for runtime support. Specifically, during runtime, the

Algorithm 2 Algorithm for determining the optional (p_o) and required (p_r) permissions of an AI service a with the root module c_{a_j} within the overall graph G based on the optional and required dependencies $\omega \in \{0, 1\}$

function PERMISSIONDETERMINATION($G, c_{a_j}, \omega = 1$)
  $C_{j\rightarrow} \leftarrow G.adjacentVertices_{\rightarrow}(c_{a_j})$
  if $|C_{j\rightarrow}| == 0$ **then**
    if $\neg\ \omega$ **then**
      return $\{readPermissions(c_{a_j}), \varnothing\}$
    else
      return $\{\varnothing, readPermissions(c_{a_j})\}$
    end if
  end if
  initialize sets p_o, p_r
  for each c_i in $C_{j\rightarrow}$ **do**
    $\{e_{a_j \rightarrow i}, \omega_{ij}\} \leftarrow G.edge(c_{a_j}, c_i)$
    $\{p_{o_i}, p_{r_i}\} \leftarrow PermissionDetermination(G, c_i, \omega_{ij} \wedge \omega)$
    $p_o \leftarrow p_o \cup p_{o_i}$
    $p_r \leftarrow p_r \cup p_{r_i}$
  end for
  return $\{p_o, p_r\}$
end function

`openAIgraph` concept supports providers in the operation of their AI services, especially with regard to data and control flow.

4.2.2.1 *Data Flow: Permission Management*

Starting with the data flow and the access to it, the `DC-PDS` engine manages both. Specifically, the engine ensures that the data is passed between the modules according to the internal graph representation of the respective AI service. For explanation, the data flow is in the opposite direction to the direction of the (data dependency) edges between modules; these modules contain the computations, receiving data messages/streams from the predecessor modules and sending data messages/streams to the successor module in a data-driven manner. Some modules may only send data (*source modules* with a connection to sensors), others may only receive data (*sink modules* with a connection to actuators); the rest typically sends data in response to received data.

For data authorization, the `openAIgraph` concept proposes a data flow based permission management. To determine the permissions required for an AI service, the engine performs the Algorithm 2: a depth-first search over the graph G, recursively aggregating all required (p_r) and optional (p_o) permissions of the subscribed modules – these permissions must be granted by the users, e.g., when they activate the AI service in the *Service Store*. The time complexity of the algorithm is $\mathcal{O}(|C| + |E|)$; this is mainly characterized by the underlying depth-first search algorithm.

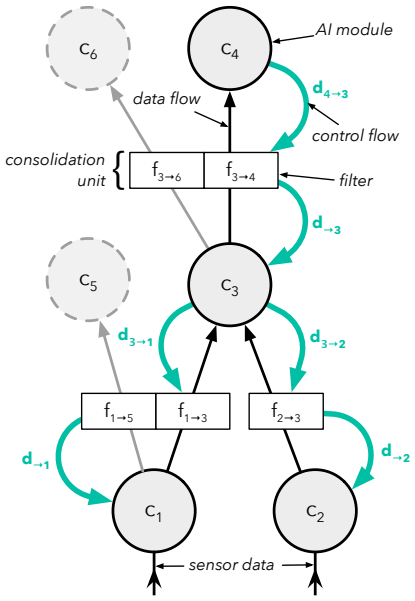


Figure 4.5: Recursive filter adaptation by consolidated demands; illustrated by the example of an initial demand request $d_{4\to3}$ (*green*) from the AI module c_4, which traverses the subgraph $\{c_1, c_2, c_3, c_4\} \subset G$. For the sake of simplicity, the version numbers of the AI modules are omitted (*gray color* instead of blue shades).

4.2.2.2 *Control Flow: Filters and Demands*

To better control the data flow and meet the demands of the individual AI modules, the `openAIgraph` concept introduces *recursive filtering on demand*; it contains two control mechanisms, namely *filtering* and *demanding*, to adjust the data flow (and indirectly influence the execution of the data-driven modules). The former filters the data flow between AI modules; the latter demands changes in the data flow (acting in the opposite direction of the former). Figure 4.5 illustrates the working of both, which are explained in detail below.

Each data dependency edge $e_{i\to j} \in E$ between two AI modules (c_i, c_j) has a *filter* $f_{j\to i}$ (managed by the engine); it filters the data flow at its corresponding edge based on the specified filter criteria (e.g., data frequency, data confidence) or filter functions – both allow to support the filtering of not only numerical values but also different data types and formats (e.g., event streams, photos, video and audio streams). To adjust these filters or request changes in the data flow, a module c_i can send a *demand* $d_{i\to j}$ with its desired attributes a_{ij} (e.g., a minimum frequency of $1/30Hz$ and a maximum accuracy of $25m$ for *location* data or passing only 'place entry' events with the place equal to 'home') to a predecessor module c_j. In particular, for a simple filter criterion, an attribute specifies the data type (e.g., location), the property (e.g., frequency), and its desired value (e.g., 1/30 [Hz]). For a more complex filter, a filter function for a specific data type (e.g., provided by the community) can be registered. By sending $d_{i\to j}$ over the edge $e_{i\to j}$, the filter $f_{j\to i}$ of this edge reads out and stores the desired attributes of $d_{i\to j}$; it also adapts its filtering criteria only for this data

Algorithm 3 Recursive filter adaptation and demand consolidation at the edges by sending a so-called *demand* $d_{i\to j}$ from module c_i to a child module c_j within graph G

1: **function** FILTERADAPTATION($G, c_i, c_j, d_{i\to j}$)
2: $e_{i\to j} \leftarrow G.edge(c_i, c_j)$
3: $f_{j\to i} \leftarrow getFilter(e_{j\to i})$
4: $adjustFilter(f_{j\to i}, d_{i\to j})$
5: $storeDemand(f_{j\to i}, d_{i\to j})$
6: $d_{\to j} \leftarrow d_{i\to j}$
7: $E_{j-} \leftarrow G.incidentEdges_{\to}(c_j) \setminus \{e_{i\to j}\}$
8: **for each** edge $e_{j\to x}$ in E_{j-} **do**
9: $f_{j\to x} \leftarrow getFilter(e_{j\to x})$
10: $d_{x\to j} \leftarrow readStoredDemand(f_{j\to x})$
11: $d_{\to j} \leftarrow consolidate(d_{\to j}, d_{x\to j})$
12: **end for**
13: $receiveDemand(c_j, d_{\to j})$
14: **for each** vertex c_c in $G.adjacentVertices_{\leftarrow}(c_j)$ **do**
15: $d_{j\to c} \leftarrow createDemand(c_j, c_c) \mid d_{\to j}$
16: $FilterAdaptation(G, c_j, c_c, d_{j\to c})$
17: **end for**
18: **end function**

type and for this edge accordingly.

Next, the engine consolidates all (stored and latest) demands $D_{\to j}$ sent by the successor modules of c_j – we call this a *consolidation unit*, which is fully-controlled by the engine. Consolidation here is a process of finding common ground for each attribute, which meets all demands $D_{\to j}$ to c_j. The consolidation unit then modifies the triggering demand $d_{i\to j}$, resulting in a demand d_j with consolidated attributes. This demand is then passed down the subgraph recursively starting with the AI module c_j.

At a node, the corresponding AI module can (optionally) adjust its computation accordingly to meet the incoming demand. For instance, an AI module can switch its internal AI algorithm to a more advanced one, when a higher confidence is demanded, but typically for a certain cost, e.g., higher resource use or completion times. If an AI module c_j needs a different incoming data flow, it can send a modified demand $d_{j\to C_{j\to}}$ to all its children $C_{j\to}$. If the module developers/providers have not ensured that the algorithm can adapt to certain circumstances (e.g., demands for a higher confidence, higher frequency, or lower resource use), the engine just forwards d_j as $d_{j\to C_{j\to}}$.

Algorithm 3 summarizes the functioning of the filter adaptation and demand consolidation. The time complexity of the algorithm is $\mathcal{O}(|C|+|E|)$, which is again mainly characterized by the underlying depth-first search algorithm.

4.3 CONCEPT REALIZATION

This section describes the realization of the DC-PDS proof-of-concept prototype and, in particular, elaborates on the key implementation decisions. The underlying DC-PDS concept is presented in Section 4.1.2, comprising the following four components. *First*, the connector part of the prototype relied on an adapted version of the Kraken.me platform, which constitutes a multi-device user tracking suite for mobile, social, and desktop: it can automatically collect user-related data from different types of sensors and fuse them into a single data store in a prepared form [SS14a; Meu14] (the author of this thesis was part of the research team that developed this platform).

Second, the data store part was realized with Apache Cassandra[11] and GraphQL[12]. Cassandra was used as the underlying (distributed, NoSQL) database: it implemented the isolated data stores and enforces the access policies of the data stores with its internal role-based access control mechanisms (*access layer*). GraphQL served as a query language and runtime environment with a single endpoint (*query layer*); it allows the requesters to define the structure of the data required, thus replacing the need for (internal) Application Programming Interfaces (APIs).

Third, we decided to use ActiveMQ[13], an asynchronous messaging system, for the communication between the AI modules via the engine, as it already decouples the senders from specific receivers. We extended the message broker to implement the filter and demand management of openAIgraph (see Section 4.2.2.2). We used JSON as message format for the *demands* and the *service/module descriptions* (manifest file). The stores (*Service Store* and *Module Store*) used a *Neo4j*[14] database to handle these service/module descriptions in a graph-based manner and to reflect their dependencies.

Fourth, we used *Docker*[15] containers with *Alpine Linux*[16] as lightweight security-oriented Linux distribution to package and deploy AI services and modules [Meu+17f]. In this way, the prototype implementation is compatible with latest edge computing approaches and can therefore easily be built on top of them, e.g., for workload distribution. *LibreSSL*[17], which is integrated in Alpine Linux, and BouncyCastle[18] (for extended cryptographic and Java language support) were used to realize the PKI. To realize the secure execution environment, we used *Google gVisor*[19], as it provides a lightweight sandboxed container runtime to enable strong isolation to the host kernel and the underlying hardware while running in user space; it is also fully compatible with Docker.

[11] http://cassandra.apache.org (retrieved 06/30/2021)
[12] https://graphql.org (retrieved 06/30/2021)
[13] http://activemq.apache.org (retrieved 06/30/2021)
[14] https://neo4j.com/ (retrieved 06/30/2021)
[15] https://www.docker.com (retrieved 06/30/2021)
[16] https://alpinelinux.org (retrieved 06/30/2021)
[17] https://www.libressl.org (retrieved 06/30/2021)
[18] https://www.bouncycastle.org (retrieved 06/30/2021)
[19] https://gvisor.dev (retrieved 06/30/2021)

4.4 METHODOLOGY

This section presents the evaluation methodology of the proposed DC-PDS concept. The conducted experiments with the DC-PDS prototype implementation (see Section 4.3) on several constructed AI services using well-known datasets and an off-the-shelf device.

4.4.1 *Constructed Test Services*

To simulate several AI services and to avoid side effects, the computations within the AI modules and services are kept the same in the experiments (as they are identical to their monolithic counterparts); only the data traffic between the computations varies, as several computations can run in one module or distributed over several modules. The construction of test services that share AI modules is carried out from bottom to top, i.e., first the lowest AI modules in the graph are replaced with one that has the same functionality as the replaced ones – this simple 'replacement' simulates the development of AI services in reality, which takes place during their development over a much longer period of time and results in a certain graph.

4.4.2 *Experimental Apparatus*

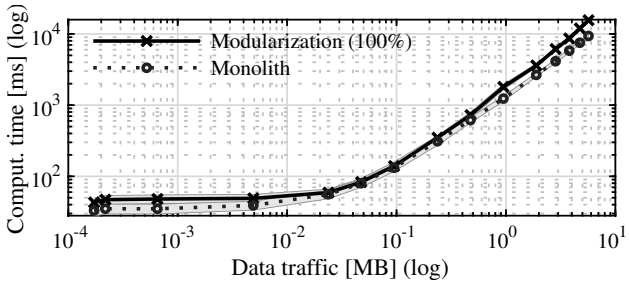
For the experiments, an Intel NUC test device hosted the user's DC-PDS, which is comparable to today's smart home servers or NAS systems; it is equipped with a 7th-generation Intel Core i5-7260U dual-core (2/4) x86-processors@2.2GHz (up to 3.4GHz), 16 GB DDR4-2400-RAM, and 500GB SSD.

4.4.3 *Performance Metrics*

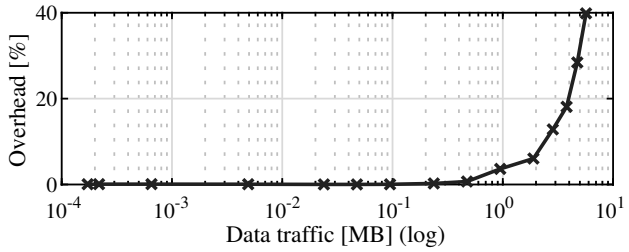
An automated test framework was developed to conduct the experiments and obtain the measurements. Each experiment was repeated 20 times; the resulting values were then averaged. In particular, two metrics were used in the following experiments: *completion times* and a newly-introduced *reusability metric*. The former metric served as a performance indicator, representing the time it takes for data to pass through the processing pipeline (or graph) and be analyzed when computing resources are fully available. As there was no suitable metric in the literature, the latter metric was self-defined as follows:

$$m_r(G) = 1 - \frac{|C \backslash C_{deg_0^-}|}{|E_{\leftarrow}|} = 1 - \frac{|C \backslash C_{deg_0^-}|}{\sum_{c \in C} deg^-(c)} \tag{4.4}$$

In addition to the already-defined graph symbol $G(C, E)$, $C_{deg_0^-}$ denotes the subset of (data flow) 'sink' modules (i.e., modules with data dependency in-degree $deg^- = 0$ – note that the data dependency direction is opposite to the data flow direction); $|E_{\leftarrow}|$ is the number of all incoming data dependency edges of the



(a) Completion time



(b) Communication overhead

Figure 4.6: Performance comparison between a fully-modularized AI service (*solid*) and its monolith (*dotted*), varying the data traffic between the computations

considered modules. The value range of this metric is $[0, 1)$ – a value of 0 means that each module is used only once (i.e., the AI services do not share any AI module) while values close to 1 indicate a very high reuse of AI modules (i.e., the AI services share many AI modules).

4.5 RESULTS

This section reports and discusses the evaluation results of a proof-of-concept (PoC) implementation of `openAIgraph`.

4.5.1 *Efficiency I: Minimal Modularization Overhead*

The first experiment examines the aspect of modularization, which is a prerequisite for the further `openAIgraph` concept. To this end, a fully-modularized AI service is compared with its monolith implementation (see Figure 4.6). It is important to note that we use exactly the same computations (running on the same system) and the same data exchanged between them (being kept in the memory in both cases). The only differences are that the computations are performed in separate containers (instead of in one container as in the monolith implementation) and the data are exchanged between these containers via the `DC-PDS` engine (instead of passing the data to the next program function within the same container). For this experiment, we used a sample AI services with a chain of three modules (place detection - place visits - routine detection), whereby the first module and the second module processed the input location values that we vary (15 location values per kB). The first module is the most compute-intensive, as we used a clustering algorithm with overall average

runtime complexity of $\mathcal{O}(n \log n)$ to detect places. In the second module, the location values are assigned to the detected places (or clusters), while the third module applies pattern recognition on the place visits for routine detection. With such an 'extreme' AI service (resource-intensive module, high number of data points), we intend to compare the behavior between the monolithic implementation and the modular approach by way of example.

In Figure 4.6a, we can see that the modularized version behaves similar to the monolith but always with a certain communication overhead. As the computations are exactly the same for both, the overhead is caused solely by the engine overhead and the required communication between the AI modules. The former represents a (nearly constant) delay caused by the message handling and management within the engine. The latter depends on the average amount of data points exchanged between the AI modules and the available 'bandwidth'.

Figure 4.6b shows the communication overhead is negligible for low data traffic (or a few data points); therefore, the total overhead in this area, which is in the two-digit millisecond range, is attributable to the engine overhead. Only when the amount of data points becomes larger, the communication overhead increases considerably in relative terms. In practice, however, the communication overhead might be lower, as in this extreme case the large number of data points (e.g., representing historical data) are typically be loaded from disk (both in the monolith implementation and by the first module).

All in all, the monolithic implementation has small performance advantages compared to the modularization approach (in this 'extreme' case). However, modularization forms the basis of `openAIgraph`; its benefits – in addition to the standard ones (e.g., improved manageability, encapsulation, better scalability[20]) – become clear in the following.

4.5.2 *Efficiency II: Reduced Local Redundancy*

This section evaluates the performance gain resulting from the redundancy reduction (only running one instance of the same AI module) of the proposed `openAIgraph` concept.

4.5.2.1 *Redundancy Reduction through Sharing of AI Modules*

This experiment investigates the performance of `openAIgraph` using various AI services that share several AI modules or, more precisely, their computation results of the single instance running per shared AI module at runtime (see Section 4.2.1.1 for the conceptual design and examples). Figure 4.7 shows the performance results. We can see that there is a performance gain when redundant computations are reduced by reusing AI modules – this is possible at all, as the computations are performed on the same data from one user only. This effect increases as more and more AI services start sharing AI modules with

[20] A complementary evaluation with focus on scalability is given in [Meu+17f] (*authorship*).

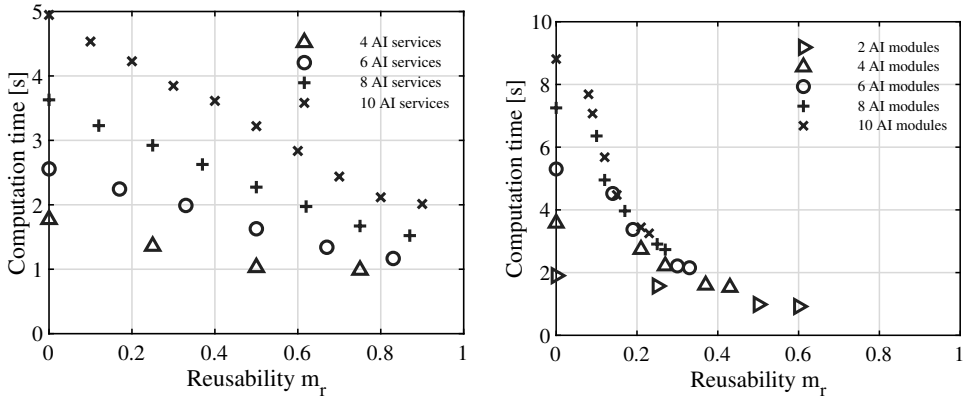


Figure 4.7: Performance evaluation of the *reusability* aspect—varying the number of AI services for a fixed average number of modules (*left*), and varying the average number of average modules for a fixed number of AI services (*right*)

other AI services, i.e., the (global) reusability factor m_r increases (see Figure 4.7, *left*). If the same number of AI services share more and more AI modules (instead of the need to run multiple identical AI module instances on the same data), this even leads to a disproportionate performance gain (see Figure 4.7, *right*)—showing the enormous improvement potential of `openAIgraph`.

All in all, `openAIgraph` can achieve a much-needed increase in the efficiency of the local system through the reusability of AI modules and the shared use of computation results. Beyond that, the communication and engine overhead caused by modularization (which is required for reusability) can be practically negligible given the performance gain enabled by reusability.

4.5.2.2 *Redundancy Reduction of Versioned AI Modules*

Finally, this experiment examines the performance of the proposed approach of backward chain of converters to ensure the backward compatibility for an AI module. To this end, the proposed approach is compared against the performance of both extremes: (1) *only the latest version of the AI module runs*—leading to the need to update each subscribing module; (2) *all versions that are subscribed run*—resulting in a high and redundant workload. In contrast to these baselines, the *proposed approach* takes the intermediate way: it only runs the latest AI modules that are still needed by other subscribing modules and 'translates' between the older outputs so that subscribers do not have to update their AI services (or more precisely, the subscribing AI module) every time.

Figure 4.8 shows the results of this experiment. We can see that the proposed approach, which also supports all versions of the sample AI module, outperforms the approach where all versions of the AI module are running. In particular, the performance curve of the former is much flatter than the latter. The proposed approach is further characterized by only a small overhead (for the conversions) compared to the approach where only the latest module version is running (requiring an update of the subscribing modules). Extrapolated to all modules

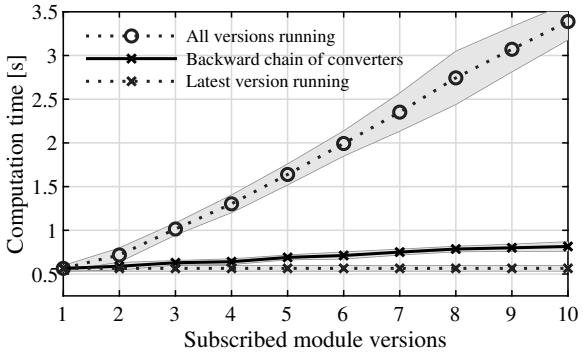


Figure 4.8: Performance evaluation of the *versioning* aspect—varying the number of subscribed versions of the same AI module

that are in such an open platform, the enormous potential for improvement in terms of efficiency becomes apparent – especially as versioning is one of the critical issues in such open platforms.

4.6 CONCLUSION

This chapter presented `DC-PDS`, a personal data store concept with a new strict data-confining policy that protects user data *at management level*. To cope with the inherent efficiency issues of locally-running data-accessing AI services, `DC-PDS` combines a sandboxed execution environment (managed by the *engine*) with provided design and runtime support (in the form of the `openAIgraph` concept) – all together, this achieves a unique tradeoff between data protection, runtime environments, and applicability.

4.6.1 *A Unique Tradeoff at Management Level*

Table 4.1 shows a summary assessment of `DC-PDS` based on the requirements established in Section 3.2.1 – supported by the evaluation in this chapter. A comprehensive comparison with the state of the art can be made using Table 3.1 (p. 38), confirming the unique tradeoff achieved by `DC-PDS`.

In terms of data protection, `DC-PDS` relies on locally-operating personal data stores with a strict data confinement policy; i.e., no (raw) user data leaves the user territory—ensuring the primary goal of data confidentiality while allowing liberal access to AI services that are forced to run the data-demanding parts locally. Through the established *open AI infrastructure* with a PKI and a global instance as trust anchor, AI services and PDSs can authenticate each other using strong certificate-based mutual (two-way) authentication. The fine-grained authorization to access the data is handled by the permission management of `openAIgraph`.

In terms of runtime environment, `DC-PDS` runs AI services in a managed sandboxing environment when they access and process user data. In addition to the central data store, AI services have their own data stores or app spaces –

Table 4.1: A summary assessment of the contributions made in this chapter based on the requirements established in Section 3.2.1 – a comparison with the state of the art protection approaches *at management level* is given by Section 3.2.2 and Table 3.1

	Data Protection				*Runtime Env.*			*Applicability*			
Contribution	Authentication	Authorization	Data Confidentiality (*observed/inferred/metadata*)	Privacy-preserving Processing	Data Store (*central/per-app*)	Low Performance Overhead/Optim.	Support of Any Data Type	Support of Any AI Algorithm	Proof-of-Concept Prototype (*implemented/evaluated*)	Low Platform-specific Dependence	Cross-site Deployment/Use (*local/nearby/cloud*)
DC-PDS	●	◐	●/●/◐	●	●/●	◐	●	●	●/●	◐	★/●/●

Note: This thesis contributes in particular DC-PDS, a representative that integrates all protection mechanisms proposed in this chapter.

Encoding: **fulfillment of requirements** (see Section 3.2.1): ●–fulfilled, ◐–partially fulfilled, ○–little or not fulfilled; **deployment**: ★–base deployment.

access is controlled by the permission management. The methods proposed by openAIgraph support in coping with the efficiency issues of locally-running AI services, eliminating computation redundancies.

In terms of applicability, like the majority of related work, DC-PDS with openAIgraph is not limited in its concept to specific AI algorithms and data types (examples for such limitations include PDV [Mun+10] and P3 [RGO13], which are limited due to their protection concept to location and photos, respectively). While developers can benefit from the reuse of shared AI module and a higher flexibility compared to monolith implementations, they can be confronted with an additional effort: for example, they require to maintain a mandatory service/module description (e.g., input and output specification) when sharing AI modules; they also have a higher effort when optionally handling *demands* in (shared) AI module to dynamically adapt it to these demands – but this entails only a low platform-specific dependency (in programming an AI service). It should also be emphasized that DC-PDS is one of the few concepts in this area for which a proof-of-concept prototype exists and has been evaluated.

4.6.2 *Integration & Outlook*

This chapter contributed protection mechanisms at management level: it decouples the data from the AI services and ensures that (i) no data inadvertently leaves the user territory and (ii) AI services and its (shared) AI modules only

have access to user data for which they have been authorized and granted (even indirectly in the latter case).

All in all, DC-PDS forms the data decentralization basis of the privacy-design platform (and the following building blocks). However, DC-PDS does not protect both user data and providers' (proprietary) AI services/ algorithms against underlying untrusted systems. This issue will be addressed next in Chapter 5.

5

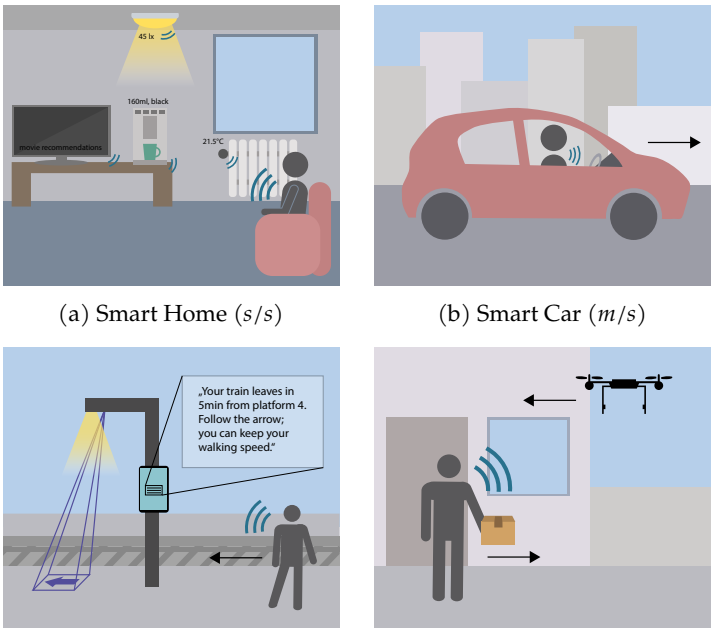
CONFIDENTIAL PROCESSING FOR PERSONALIZED AI SERVICES

The contributions from the previous chapter deal with the protection of user data against unauthorized access within the user territory, assuming that the underlying system is trusted (see Chapter 4). This chapter contributes to the protection against possibly *untrusted* systems. More specifically, this chapter focuses on the protection of user data and providers' intellectual property (IP) *at system level* – the categorization scheme is described in Section 3.1.

> *"If someone has physical access to a device, that device is no longer secure. Losing control of devices empowered with edge computing [and AI] technology can expose vastly more customer data and intellectual property than losing control of other types of devices."*
>
> Forrester Research, 2018 [Pol18]

With these words, Forrester Research describes the potential risks associated with data and service decentralization. These risks are particularly aggravated when third parties own or have access to the underlying devices, which is typically the case with edge computing infrastructures [Müh+20]. In this context, we consider AI services that need to run *confidential parts of their services* (e.g., IP-protected AI algorithms/models) locally or, in the case of distributed AI services, need to offload these confidential parts and *confidential user data* to edge devices. As defined in Section 2.1.2, a special case of distributed AI services constitutes device-bound AI services, as they additionally require nearby IoT (edge) devices with access to sensors/actuators to provide ambient support.

This (data, service) decentralization leads to the following *three-party confidentiality challenge* involving the *user* with confidential user data, the service *provider* with its (partly) confidential AI services, and the local *system*, e.g., on nearby third-party (IoT/edge) devices, running these AI services: the system must not access both the confidential user data and the confidential parts of AI services (generally the provider code); the user must not access the confidential provider code, whereas the provider must access the confidential user data but only locally, e.g., for the purposes of the personalization of their provided AI services; last, neither the user nor the provider are allowed to access the system. For better reference to these protection requirements, we have used the phrases *'user territory'* and *'provider territory'* (as first introduced in Chapter 1): user territory refers to the digital space that is fully-controlled by the user and cannot be bypassed or restricted by the provider or other parties; it is the digital space in which the confidential user data is kept – data leaving this metaphorical territory without permission of the user would be equivalent to losing control of the data; provider territory, by analogy, refers to the digital space in which the provider's confidential parts of AI services should run to ensure IP protection.



(a) Smart Home (s/s) (b) Smart Car (m/s)

(c) Public Displays (s/m) (d) Drone Interactions (m/m)

Figure 5.1: Examples of application scenarios (see Section 2.1.5) that illustrate the four possible configurations for an ad-hoc deployment of device-bound AI services. *Notation:* the bracket $(d\,/\,u)$ next to the captions encodes the IoT device d in the world reference system and the user u in the reference system of the IoT device (reflecting the contact times), respectively; possible values are s for (nearly) static and m for (highly) mobile.

In the specific context of device-bound AI services, the above-mentioned challenge is particularly pronounced in the *ad-hoc deployment* of such services. Figure 5.1 illustrates the four possible configurations for device-bound AI services (using the example of suitable application scenarios): (a) both the IoT device and the user are (nearly) static [Mar+08]; (b) the IoT device is mobile while the user is static to it [HS15]; (c) in this case, it is vice versa: the IoT device is static while the user is mobile in relation to it [DL19]; and in (d), both are mobile [Kni+18]. Depending on the configuration, the contact times between the user and the device vary greatly, resulting in additional requirements for initialization and run times – all requirements identified in this category have already been introduced in detail in Section 3.3.1.

As shown in the review of related work in Section 3.3.2, none of the considered data protection approaches can completely fulfill these category-specific requirements. In particular, the above-described three-party confidentiality challenge is an open issue in ad-hoc deployment of AI services to nearby, resource-restricted (IoT/edge) devices. For example, container-based virtualization approaches (often used in edge computing) are lightweight and scalable, but they offer less protection against the underlying system. A considerable fraction of protection approaches (e.g., [BPH14; Arn+16; Bra+18]) relies on SGX hardware support: this ensures confidentiality and integrity of code and data against the system. On the other hand, SGX is proprietary and comes with an inherent initialization

and runtime overhead caused by its necessary cryptographic operations. Only a few approaches (e.g., [Hun+16; Hun+18]) can protect the data against untrusted provider code, e.g., by sandboxing this code, which in turn limits the flexibility of AI services. Even fewer approaches (e.g., [Ohr+16; Bay+20]) take into account the nature of AI services and are optimized accordingly. All in all, there is *no* protection concept that can support an ad-hoc deployment in all four configurations (a-d) and fulfill all the category-specific requirements.

In the light of this deficit, we aim to achieve a suitable tradeoff between these competing requirements at system level. To this end, this thesis proposes a protection concept—namely `PDSProxy`—that combines sandbox and SGX-supported approaches to leverage their individual strengths. In addition, `PDSProxy` contributes (i) four different initialization modes to support the different use cases and (ii) certificate-based authentication mechanisms to stream user data between hierarchically-operating proxies, even over untrusted nodes. Specifically, the contribution of this chapter is threefold. First, this chapter presents a confidential processing environment that provides the basic protection features *within* a `DC-PDS`, protecting user data and providers' intellectual property (see Section 5.1). Second, this PDS-internal concept is then extended by a proxy concept – termed `PDSProxy` – to enable confidential processing *outside* the base PDS with the same protection features, expanding the user territory (see Section 5.2). This PDS-external concept enables a case-adapted ad-hoc deployment of device-bound AI services while coping with the three-party confidentiality challenge. Third, this chapter reports and discusses the evaluation results of a proof-of-concept implementation of `PDSProxy` on five use cases—demonstrating the unique tradeoff between the category-specific requirements (see Sections 5.3–5.5).

Contribution Statement: This chapter is based on the following (3) publications:

- Christian Meurisch, Bekir Bayrak, and Max Mühlhäuser. "EdgeBox: Confidential Ad-hoc Personalization of Nearby IoT Applications." In: *Proceedings of the 2019 IEEE Global Communications Conference*. GLOBECOM'19. IEEE, Dec. 2019. DOI: 10.1109/GLOBECOM38437.2019.9013520
- Christian Meurisch, Bekir Bayrak, Florian Giger, and Max Mühlhäuser. "PDSProxy: Trusted IoT Proxies for Confidential Personalization of AI Services." In: *Proceedings of the 29th International Conference on Computer Communications and Networks*. ICCCN'20. IEEE, Aug. 2020, pp. 1–2. DOI: 10.1109/ICCCN49398.2020.9209655
- Christian Meurisch. *The Trusted Edge*. 2021. arXiv: 2105.13601

I (*Christian Meurisch*) led the process of idea generation, conceptual design, evaluation, and writing. *Bekir Bayrak* implemented the proof-of-concept prototypes and conducted the evaluation – the student *Florian Giger* supported both. *Max Mühlhäuser* provided helpful critique and comments on the conceptual design.

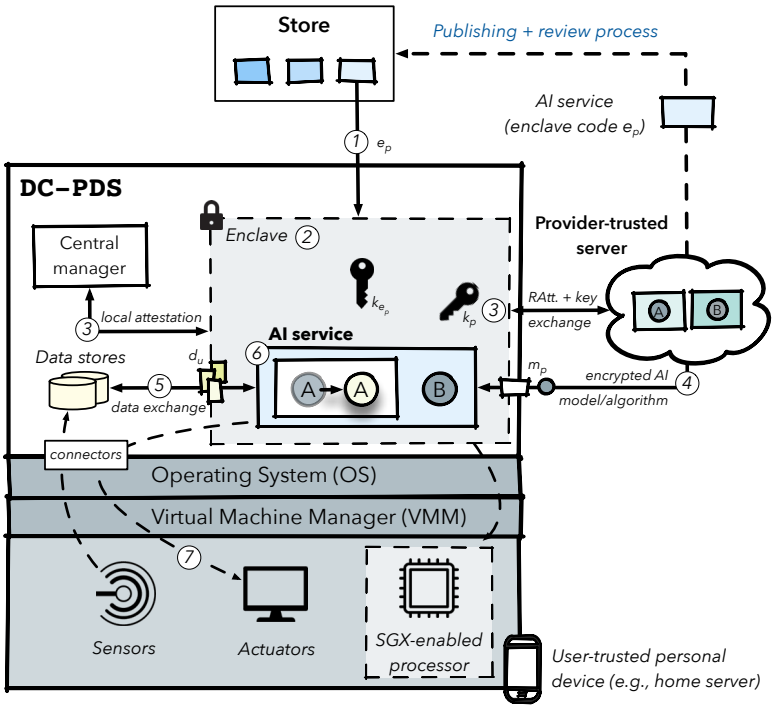


Figure 5.2: Confidential processing environment *within* the user's `DC-PDS` and its individual steps (①–⑦); *yellow* marks user data containing parts. *Abbreviations*: RAtt.–remote attestation.

5.1 PDS-INTERNAL CONFIDENTIAL PROCESSING

This section first presents the prerequisites and protection mechanisms adapted to enable confidential processing *within* the user's `DC-PDS` (see Figure 5.2). More precisely, this complements the (sandboxed) secure execution environment from Chapter 4 to provide protection even against untrusted underlying systems. We will first explain the concepts for AI services in general (running on one local device only); we will then extend them in Section 5.2 for distributed (and in particular device-bound) AI services that require to be offloaded or deployed to other nearby (third-party) edge devices.

5.1.1 *Assumptions and Prerequisites*

In this first case, we assume that `DC-PDS` (or the base PDS) runs in the user territory, e.g., on a user-trusted personal device such as a home server or NAS.

For the proposed protection mechanisms, we assume an underlying hardware-supported *processor-specific and activated Trusted Execution Environment* (*TEE*) (e.g., relying on Intel SGX) on the target device – see Section 3.3.2 (p. 41) for an introduction. Besides the basic functions, it is important that the TEE implementation used supports *remote attestation*. We also assume that an attacker cannot perform invasive hardware attacks on the device. For example, this requires that the used enclave implementation (in SGX) contains appropriate

defense mechanisms against possible side channel attacks [Arn+16; Ole+18]. The further hardening of the enclave, however, is outside the scope of this thesis – here, we refer to the active research in the S&P community.

Following current practice, providers publish their *trusted and open parts of AI services* (*blue rectangles*) on a central platform, namely a *trusted store* (e.g., similar to an app store) – this practice is fully-compatible with `OAII` introduced in Section 4.1.1 (see also Figure 4.1, p. 60). The published parts are visible to all; an optional (meanwhile common) *review process* during the publishing process can additionally contribute to trust and prevent malware or obvious data leaks. This trusted and open parts of AI services include the SGX-specific *non-confidential enclave code* e_p. The *confidential parts of the AI services* will be later downloaded from a *provider-trusted server* in encrypted form to the locally initialized *provider enclave* in step ④ (this will be explained in detail in Section 5.1.2.1).

5.1.2 *Individual Phases*

Once activated by the user, each AI service goes through two phases, namely *initialization* (①–④) and *confidential operation* (⑤–⑦) – explained in detail below.

5.1.2.1 *Initialization*

In the first step (①), the central manager downloads the newly-activated AI service, or more precisely, its trusted and public parts including the enclave code e_p from the store. The central manager running in a *sandboxed* environment then builds an enclave (referred to as *provider enclave*) from the code e_p. This thesis proposes this sandbox environment to provide (strong) isolation between the applications and the host kernel through intercepting application system calls, which come from the provider enclave, and to cope with untrusted provider code. This building process is measured by the SGX-enabled processor: it creates a cryptographic hash h_{e_p} of the initial memory content of the enclave and stores this hash securely (②). Any manipulation can be detected using this hash. Once the enclave is created, the code is immutable and completely isolated from all outside access, including access through privileged code. [CD16]

Step ③ executes the HW-based attestation: the central manager of the `DC-PDS` uses the enclave hash h_{e_p} to verify that the code has not been modified (*local attestation*) – it acts on behalf of the user. For the provider, this process is more complex: the enclave uses public-key cryptography to establish a secure channel to the provider. In particular, the enclave generates an asymmetric RSA key pair k_{e_p} with $k_{e_p}^{pr}$ as the private key and $k_{e_p}^{pu}$ as the public key and stores them inside. Next, a digital signature $\sigma(h_{e_p}, k_{e_p}^{pu})$ is generated with the remote attestation feature and the enclave key $k_{e_p}^{pr}$. The enclave then sends all three—namely h_{e_p}, $k_{e_p}^{pu}$, and $\sigma(h_{e_p}, k_{e_p}^{pu})$—to the provider. The latter uses, e.g., Intel PKI for SGX, to verify the digital signature, validating the authenticity and integrity of h_{e_p} and $k_{e_p}^{pu}$. With the hash h_{e_p}, the provider can check if the enclave code is not modified before creation, authenticating the underlying system (*remote*

attestation) – for an example of a full remote attestation flow for SGX including key exchange, we refer to the Intel Attestation Service[21]. Finally, both exchange a symmetric key k_p – this key is created and encrypted by the provider with $k_{e_p}^{pu}$, and it is (and can only be) decrypted by the enclave with $k_{e_p}^{pr}$. [Bra+18; CD16]

In the last initialization step (④), inspired by `VoiceGuard` [Bra+18] and adapted to ad-hoc deployment scenarios (in Section 5.2), the enclave can subsequently download the missing and *confidential parts m_p of the AI service* from the provider-trusted server (see Figure 5.2: AI models as *gray circles*; AI algorithms as *white rectangles*). To this end, the provider encrypts the protected parts $enc(m_p)_{k_p}$ with the shared key k_p, so that they can only be decrypted inside the enclave after download; the enclave then initializes them to complete the AI service. In this way, the provider's IP is protected against both the user and the system.

5.1.2.2 *Confidential Operation*

In the second phase, when the AI services within the provider enclave need confidential user data, e.g., for the purpose of personalization, this thesis proposes to block all outgoing data traffic from the provider enclave to ensure that the confidential user data cannot leave the user territory (the `DC-PDS` as a *sandbox* takes care of this). However, this means that AI services cannot send data outside its hosting enclave without the user's permission (if any actions are allowed by the user, they can be monitored by the `DC-PDS` and blocked if necessary) – as a limitation, this includes requests via the Internet and direct connections. For instance, if it is necessary to update the AI models/algorithms used, a reinitialization of the provider enclave must be performed, i.e., the steps from step ② onwards must be performed again – before an enclave reinitialization, intermediate results can be stored locally in encrypted form with k_p (the provider-trusted server requires to maintain this symmetric key, e.g., one per device) and reloaded in the confidential operation phase again. Only with these measures, the enclave can now request user data d_u required for the AI service from the central manager. If the required permissions exist (see Section 4.2.2.1), the data stores securely stream the user data to the enclave (⑤).

An AI service has two sub-phases (⑥): (a) the optional *training task* of the underlying AI models and (b) the *inference task*. The personalization to the user can be carried out in both sub-phases – see Section 2.1.3 (p. 14) for more details. In short, the AI models can subsequently be personalized to users using their personal data during training, while personalization in the inference task is carried out by 'interpreting' or classifying user data. A result of the AI service is then passed as a controlled 'action' to the platform-managed connectors for controlling the specific actuators (⑦), thus providing ambient user support.

[21] https://software.intel.com/content/www/us/en/develop/articles/code-sample-intel-software-guard-extensions-remote-attestation-end-to-end-example.html, Figure 3 (retrieved 06/30/2021)

5.2 PDS-EXTERNAL CONFIDENTIAL PROCESSING

The previous section contributes mechanisms to protect the confidential parts of AI services running locally in the user's `DC-PDS` on one device only. In this section, this thesis contributes a new extension to enable the same level of protection *outside* the user's `DC-PDS` on nearby (third-party) edge devices, supporting both an offloading of distributed AI services and an ad-hoc deployment of device-bound AI services. The prerequisites for this, the underlying adapted concepts, and the necessary initialization optimizations are explained in detail below.

5.2.1 *Assumptions and Prerequisites*

Besides an SGX-enabled processor (see Section 5.1.1), the target device requires a minimal set of pre-configurations. First of all, the device must be *accessible* via a use case suitable network in order to establish a secure connection to it. For instance, an Internet connection is obvious for cloud-hosted AI services. For distributed (and device-bound) AI services, in contrast, a wireless (ad-hoc) direct link (e.g., based on Apple's AWDL protocol [SKH18]) may be the choice, especially for mobile use cases like (c-d) in Figure 5.1 – for use cases similar to (a-b) in Figure 5.1, a connection via a local (home/car) network is also conceivable. In addition, the device must support the *execution of permitted platform code* in an OS-controlled user space with limited privileges or within a lightweight sandboxed container runtime (to protect the device as well). Such pre-configurations might form a new kind of future standard that devices can implement to be personalized ad hoc – the concept for this is introduced next.

5.2.2 *Trusted PDS Proxies*

The platform code (or a future OS feature) preconfigured on the target device only handles receiving and running the user-specific container—namely `PDSProxy`[22], which constitutes the main contribution of this chapter.

5.2.2.1 *Concept*

A `PDSProxy` is a lightweight and portable *functional replica* of the user's `DC-PDS`: it can be transferred to other devices, setting up the same protection mechanisms there as in the `DC-PDS` and acting as intermediary to the latter. In particular, this thesis suggests two new design choices: (1) `PDSProxy` uses separate enclaves, namely a *user enclave* and *provider enclaves*, to keep both the confidential user data and the confidential parts of the AI services protected from the underlying system, whereby the user enclave initiate the provider enclaves; inter-enclave communication can exchange data in a confidential way. (2) On the other hand, it uses a *sandbox* approach to protect the user data from being leaked by untrusted provider code – the detailed functioning will be explained in Section 5.2.3.

[22] `PDSProxy` is a further development with name adaptation of the initial `EdgeBox` concept from [MBM19a]. Concretely, the latter is incorporated into the former in this thesis.

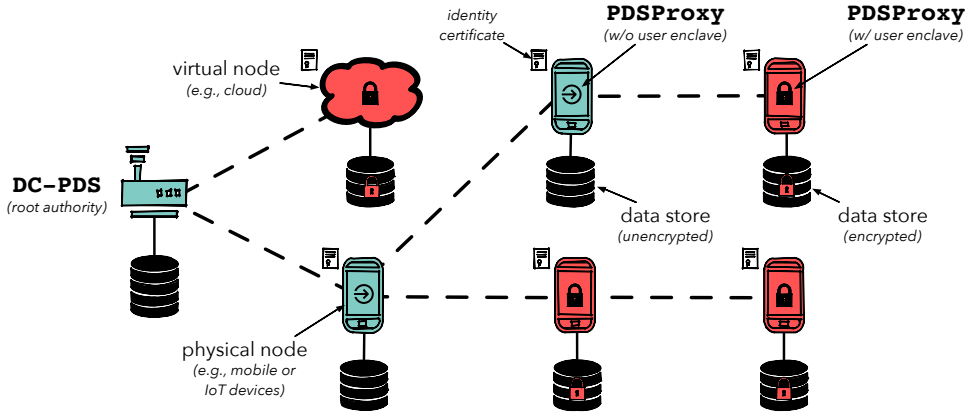


Figure 5.3: Schematic representation of hierarchically-operating PDS proxies over (virtual or physical) un-/trusted nodes (*red*/*green*)

In practice, each user runs only one such proxy instance per target device, which supports and coordinates multiple AI services – this is carried out for performance reasons and to avoid data redundancies in the user enclave. Beyond that, proxy instances can be organized hierarchically across multiple systems—running on both *virtual nodes*, e.g., in the cloud, or *physical nodes*, e.g., mobile or IoT/edge devices (see Figure 5.3). This also means that not only the base `DC-PDS` can instantiate a new `PDSProxy` on other systems, but also the proxy itself. With the additional proxy logic, `PDSProxy` can securely communicate with its parent `PDSProxy` or the user's base `DC-PDS` directly, e.g., to request and stream required confidential user data in an encrypted form into its local user enclave.

To better grasp this concept, imagine users (i) hosting their base `DC-PDS`, which contains *all* their personal digital data, on their user-trusted home server or private cloud space and (ii) carrying a `PDSProxy` (referred to as *P1* in this example) on their mobile device, which can cache few required data locally. Walking around, users can initiate and benefit from surrounding device-bound AI services (see Figure 5.1c–5.1d) as follows: *P1* transfers a 'fork' of itself (referred to as *P2* in this example) to the nearby (IoT/edge) device that fulfills the prerequisites in Section 5.2.1; *P2* acts as an autonomous `PDSProxy` on this device, with the difference that *P1* is now its direct parent node (and not the user's `DC-PDS`). As a parent node, *P1* has the right to terminate *P2*, which in turn terminates all possible child nodes of *P2*. On the other hand, *P2* can request and stream (missing) required data either from *P1* or from the `DC-PDS` directly, depending on, e.g., the connectivity, the availability of the data on the respective available node, and the latency requirements of the AI services (the decision logic will be explained in more detail in Section 5.2.3.3). In this new decentralized way, (mobile) users can expand their territory (even if only temporarily) to ad hoc initialize and use personalized AI services on nearby devices while the user data never leaves the user territory.

5.2.2.2 *Trust Model*

In general, most users trust at least some of the involved devices mentioned above. This allows one to distinguish between *trusted nodes* (T) and *untrusted nodes* (U). Typically, the former are those owned and controlled by a user, while the latter are owned and managed by third parties. Nevertheless, the trust differentiation can also be defined according to other criteria, such as user preferences, security measures in place, and the given scenarios.

To account for this, unlike related work, `PDSProxy` employs a *centralized trust model* using PKI – see Figure 5.3. The PKI consists of several hierarchically-operating CAs, of which the (base) `DC-PDS` is the Root Authority (RA)—serving as a trust anchor. Each CA manages its own PKI and issues Identity Certificates (ICs) for authentication and data authorization [Koh+17]. This means that a `PDSProxy` acting as CA can issue ICs based on its own access rights to data. For security reasons, `PDSProxy` further establishes the following two distinctions between untrusted (U) and trusted (T) nodes: (1) only trusted nodes with a CA_T can request more permissions from its `DC-PDS`, and (2) an IC_U (for untrusted nodes) should have a short-term but still case-specific expiration date. A future alternative to this concept can be based on OAuth, which is the 'industry-standard protocol for authorization'[23].

5.2.3 *Adapted Individual Phases*

Unlike `DC-PDS`, `PDSProxy` requires an additional preceding phase, namely *proxy transfer and setup* (Ⓐ-Ⓓ), in which a `PDSProxy` instance is transferred to a nearby device and set up there – these steps will be introduced next in Section 5.2.3.1. Like in `DC-PDS` (see Section 5.1.2), the user-activated AI services go through their two phases, namely the *initialization* and *confidential operation* (comprising the steps ①–⑦). However, due to the fact that it runs on another device like the one running the DC-PDS, `PDSProxy` needs small adjustments in some of these steps. In Section 5.2.3.2 and Section 5.2.3.3, we will only elaborate on these adjustments. All steps of these three `PDSProxy` phases are illustrated in Figure 5.4 as an architecture overview and in Figure 5.5 as a sequence diagram.

5.2.3.1 *Proxy Transfer & Setup*

First, the initiating (parent) node—running the `DC-PDS` or a `PDSProxy`—transmits the proxy code *pc* to the target node and initiates a new `PDSProxy` instance there. In particular, if the proxy code *pc* is confidential, this phase proceeds as follows, analogous to steps ①–④ in Section 5.1.2.1: the initiating node only transfers the enclave code e_{pc} to the target node (Ⓐ). An enclave (referred to as the *user enclave*) is then built from e_{pc} (Ⓑ): this process also generates $h_{e_{pc}}$ (a cryptographic hash of the initial memory content of the enclave) and stores it securely; the resulting enclave then generates $k_{e_{pc}}$ (an asymmetric RSA key pair). In the next step (Ⓒ), the initiating node remotely attests[21] the user enclave using the digital

[23] https://oauth.net/2/ (retrieved 06/30/2021)

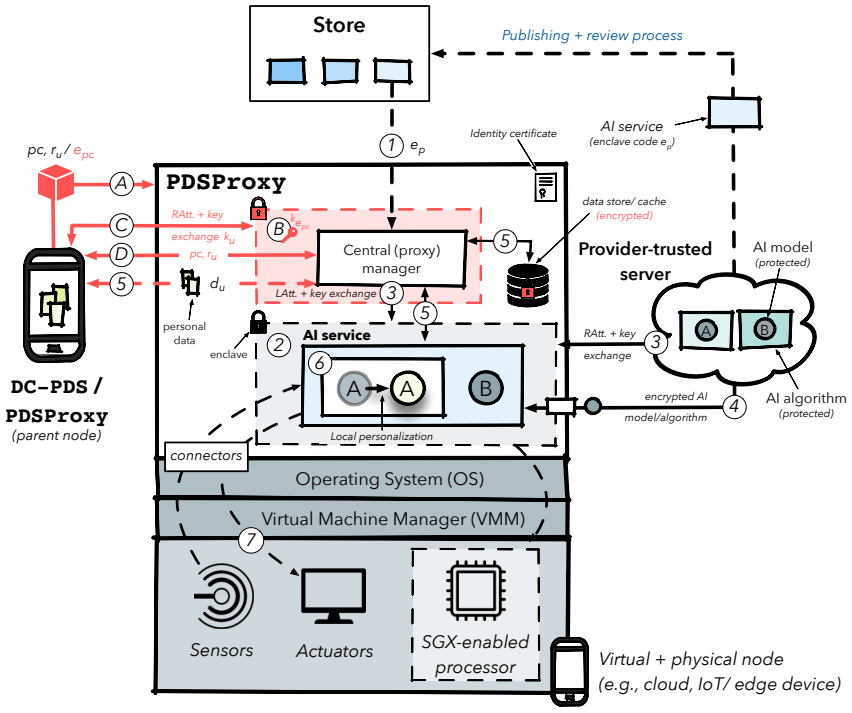


Figure 5.4: Confidential processing for distributed AI services *outside* the user's DC-PDS on un-/trusted nodes made possible through PDSProxy; differences in the proxy variants are marked in *red*. *Abbreviations*: LAtt.–local attestation; RAtt.–remote attestation.

signature $\sigma(h_{e_{pc}}, k^{pu}_{e_{pc}})$; in the same step, a symmetric key k_u (to transfer the confidential parts in encrypted form) is exchanged via the above-established secure channel [Bra+18]. In the last step (Ⓓ), the initiating node transfers the actual proxy code *pc* including the identity certificate *IC* and the user's 'request' r_u (specifying the desired AI services) to the user enclave. To this end, the initiating node encrypts the confidential parts $enc(pc)_{k_u}$ so that it can only be decrypted within the enclave using the key k_u – the proxy is finally set up.

5.2.3.2 *Initialization*

In this phase, the central manager of the PDSProxy initializes the desired AI services according to the user's request r_u. Each AI service goes through the same individual workflow steps. More precisely, these initialization steps (①–④) are identical to those with the same numbering in Section 5.1.2.1 (p. 83). Alternatively, all AI services per provider can be grouped together, sharing the same provider enclave (if memory is not an issue here). Same as in Section 5.1.2, it is important to note that the user enclave starts and also manages the provider enclaves on the system; with the sandboxing, it can control all system calls, and it blocks all incoming and outgoing traffic for the next phase.

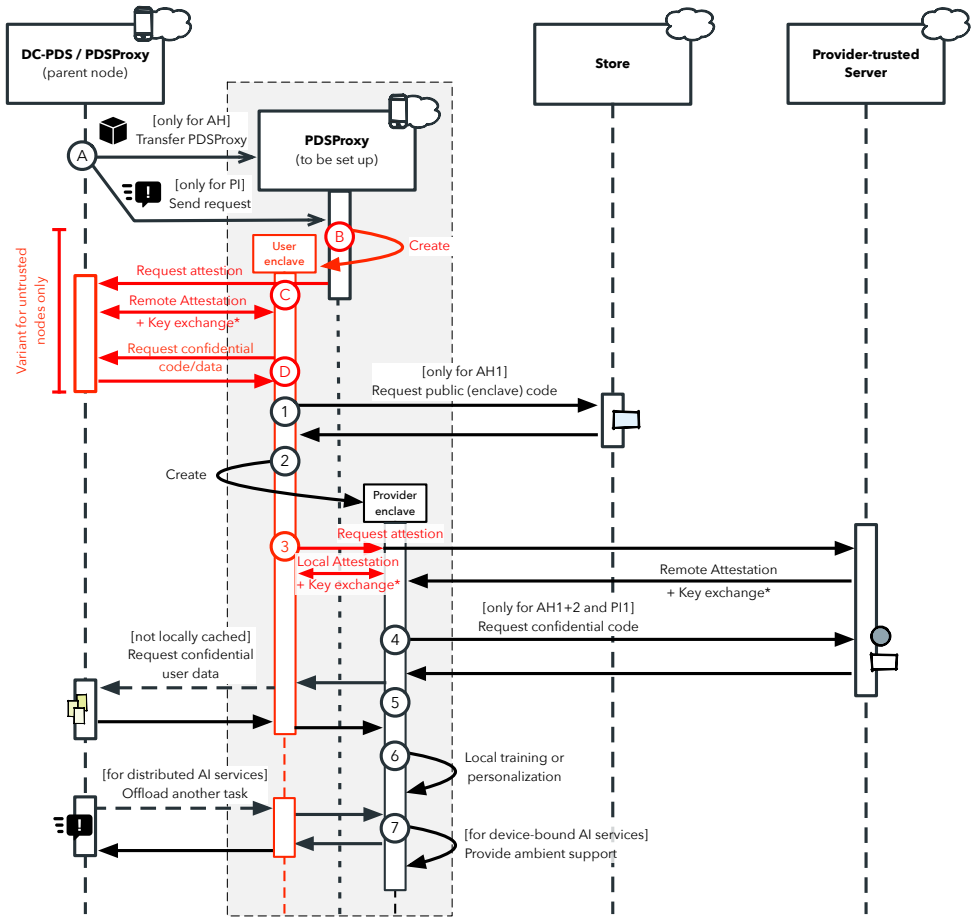


Figure 5.5: Complete sequence diagram of `PDSProxy`, showing the proxy variant for untrusted nodes (in *red*) and the different initialization modes (specified as condition). The asterisk (*) marks all 'remote attestation and key exchange' steps, whose entire interactions are simplified; for an example of a full remote attestation flow, we refer to the link[21]. The process is similar for distributed and specifically for device-bound AI services, both differing only in step ⑦. Dashed arrows indicate optional processes.

5.2.3.3 *Confidential Operation*

In this phase, AI services within the (system call restricted) provider enclaves can request required confidential user data d_u from the central manager within the user enclave, using secure inter-enclave communication (⑤). If the required permissions exist (see Section 4.2.2.1), the central manager streams the user data from its *local cache*, i.e., from its memory or from the encrypted data store on the file system. By default, user data required for the service should be prefetched in advance or injected in the proxy before transmission (Ⓐ/Ⓓ) – this is especially the case for latency-critical offloading tasks from distributed AI services.

Optionally, if the user data does not exist locally or for *long-running* AI services (e.g., device-bound AI services in smart home or car environments), which need updates from time to time, the central manager can request and stream the required user data from its parent nodes using certificate-based authentication;

it proceeds as follows (see Figure 5.3): (1) it first requests the direct parent node (if still available), as this connection is typically characterized with the lowest latency in the chain; the direct parent node checks its local cache for the data required. (2) If the direct parent node cannot provide the data, the central manager requests the data from the `DC-PDS` via the Internet. (3) If the current node has no internet connection or cannot connect to the `DC-PDS`, the algorithm can then recursively queries its parent nodes, each repeating the first two steps. The algorithm determines (i) if the requested data is found, (ii) a predefined timeout occurs, or (iii) no other connections to parent nodes exist, e.g., as the `DC-PDS` as chain root is reached. The requested data (if found) can then be streamed in encrypted form $enc(d_u)_{k_u}$ to the central manager—constituting a confidential stream (referred to as *cstream*)—and cached locally (also at the intermediate nodes) for further requests (referred to as *hierarchical caching and streaming*). Depending on the use case, this procedure can be adapted. For example, if latency is not an issue, e.g., as a fallback solution in emergency situations, `PDSProxy` can rely on delay-tolerant querying [Meu+17i] (*authorship*).

For device-bound AI services, the remaining steps (⑥+⑦) are then again identical to those with the same numbering in Section 5.1.2.2. Data newly collected (e.g., via the sensors of the IoT device) or inferred on the nodes can be synchronized upwards to the `DC-PDS` as well. For distributed AI services, only step ⑥ is identical to the known one. Step ⑦ differs in the way that the result of the offloaded part of the AI service is sent back to the parent node; `PDSProxy` can still remain initiated for new tasks to be offloaded (data offloading) – new data is transmitted encrypted to the enclaves (this is indicated in Figure 5.5, ⑦).

5.2.4 *Case-specific Optimizations*

In this section, this thesis contributes two new *optimization mechanisms* for the setup and initialization process: one establishes different *proxy variants* depending on the trustworthiness of the underlying system (see Figure 5.3); the other introduces different *initialization modes* for AI services to cope with the time constraints of different mobile scenarios (see Figure 5.1).

5.2.4.1 *Proxy Variants*

The `PDSProxy` concept is particularly designed for untrusted nodes. When operating on trusted nodes (those that users trust), however, the proxy setup process can be simplified: the proxy code *pc* must no longer necessarily run within an enclave. More precisely, the steps Ⓑ-Ⓓ are omitted, as the entire proxy code *pc* is already transmitted to the node and executed there in step Ⓐ. This also means that the data store on the file system does not need to be encrypted, further reducing the computational overhead. The remaining steps ①–⑦ remain the same; these steps (except step ⑤) concern the protection of the confidential parts of AI services and their local personalization with the streamed confidential user data (⑤) – further details were given in Section 5.2.3.

Table 5.1: Case-specific initialization modes for *external* confidential processing of distributed AI services; variant differences are marked in *gray*. The table entries indicate the initial location (*top header*) of the data. Below the top header, the connection type (via ad-hoc direct connection or the Internet) and the way in which the data is transferred to the target node is specified: `PDSProxy` denotes the initial container that is first transferred (Ⓐ); the term *'cstream'* denotes an encrypted transmission from one party directly into its enclave located on the target node. Encoding: *PtS*–provider-trusted server; *pc*–proxy code; e_p–provider enclave code; m_p–confidential (encrypted) provider code (or AI algorithms/models); r_u–user request; d_u–confidential (encrypted) user data.

			Target Node	**Parent Node**		**Store**	**PtS**
				Ad-hoc conn. (←)		Internet (↓)	
Mode	**Variant**	**ID**		`PDSProxy`	*cstream*	-	*cstream*
Ad-hoc-initialized	ad-hoc I	AH1	–	pc, r_u	d_u	e_p	m_p
	ad-hoc II	AH2	–	pc, r_u, e_p	d_u	–	m_p
Pre-initialized	pre-init I	PI1	pc, e_p	r_u	d_u	–	m_p
	pre-init II	PI2	pc, e_p, m_p	r_u	d_u	–	–

All in all, `PDSProxy` can run on both un-/trusted nodes (see Figure 5.3) but in different setup variants (see Figures 5.4+5.5, differences marked in *red*), ensuring integrity and confidentiality of AI services and user data at all times.

5.2.4.2 *Initialization Modes*

Unlike related work, the proposed `PDSProxy` concept is particularly designed for an ad-hoc setup from scratch on nearby target (IoT/edge) devices to support distributed and in particular device-bound AI services. Depending on the application scenario (see Figure 5.1, c-d), however, this is not always suitable or even feasible, e.g., due to short contact times, poor Internet connectivity, or latency-demanding AI services. To cope with these different setup constraints, `PDSProxy` further introduces two different initialization modes for distributed AI services: (1) *ad-hoc-initialized* (i.e., the proxy is transferred from the parent node to the target node and initializes itself from scratch), and (2) *pre-initialized* (i.e., at least the common non-confidential code is already initialized on the target node). Each of them has different variants for reasons of optimization, but without loss of integrity and confidentiality.

Table 5.1 shows the different case-specific initialization modes and their variants; Figures 5.5 also illustrates these different initialization modes (marked as conditions). In the *ad-hoc-initialized mode*, the proxy is transferred to the target node as usual (steps Ⓐ-Ⓓ); however, step ① is different: instead of loading the AI service, or more precisely, the provider's enclave code e_p from the store (*ad-hoc I*), it can also be injected into the proxy container (`PDSProxy`) in advance before it is transferred over the local ad-hoc connection (*ad-hoc II*). In the *pre-initialized mode*, the proxy code *pc* and the non-confidential enclave code e_p of the selected and available distributed AI services are already pre-initialized at

the target node. For the former, in the steps Ⓐ-Ⓓ, the transfer size is considerably reduced to the user request r_u (which specifies the desired AI service) and the attestation process (if necessary – see Section 5.2.4.1). For the latter, steps ①-③ can be prepared before the user even needs the AI services on the node. Here, the variants of this mode differ only in step ④: the provider's confidential parts of the AI services (including the AI models/algorithms) m_p are either loaded from the provider-trusted server in a confidential manner (*pre-init I*) or already cached on the untrusted file system of the target node (*pre-init II*). It is important to note that both variants use the encrypted form $enc(m_p)_{k_p}$ with the provider's key k_p. In all variants, the confidential user data d_u must still be transferred from the parent node to the target node in a confidential manner (*cstream*).

All in all, `PDSProxy` can be initialized on PDS-external nodes (see Figure 5.3) in two/four different initialization modes/variants (see Table 5.1), ensuring integrity and confidentiality of AI services and user data at all times.

5.3 CONCEPT REALIZATION

This section describes the realization of the `PDSProxy` proof-of-concept prototype and, in particular, elaborates on the key implementation decisions. To be compatible with the concepts proposed in Chapter 4 (and the implemented proof-of-concept prototype), we have built on the software presented in Section 4.3: *Docker*[15] containers with *Alpine Linux*[16] as lightweight security-oriented Linux distribution are used to package and deploy AI services. *LibreSSL*[17], which is integrated in Alpine Linux, and *Bouncy Castle*[18] (for extended cryptographic and Java language support) is used to realize the PKI for the hierarchically-operating PDS proxies (see Section 5.2.2.2).

To realize the `PDSProxy` concept, two implementation decisions are crucial, namely those for the *sandbox environment* (to protect user data from untrusted provider code) and the *trusted execution environment* (to protect both user data and provider code from the underlying system, and to protect the latter also from the user). For sandboxing AI services, traditional containers are not sufficient, as they can make system calls directly to the host kernel. While the kernel can restrict resources through *cgroups*, *namespaces*, and *seccomp* filters, not all resources can be controlled with these mechanisms, or users must create a predefined whitelist of system calls. Even with these restrictions, the kernel still exposes a large surface area that can be directly attacked by malicious AI services. Running containers in a separate VM provides strong isolation, which is perfect for sandboxing untrusted provider code, but it involves a large resource overhead. *Kata* containers[24] and *Google gVisor*[19] are both open-source projects that address these issues, providing a good balance between isolation and resource footprint. While Kata slims down VMs to keep the resource footprint minimal and maximize performance for container isolation, gVisor provides a sandboxed container runtime, which is compatible with Docker and Kubernetes:

[24] https://katacontainers.io (retrieved 06/30/2021)

gVisor[25] is more lightweight than a (slimmed-down) VM but has a similar level of (strong) isolation through intercepting application system calls and acting as the guest kernel. While gVisor can adapt to changing resources (unlike VMs), this flexibility comes at the cost of higher per-system call overhead [You+19]. Having carefully weighed these pros and cons, we decided to use gVisor[26] for sandboxing.

For the hardware-enforced trusted execution environment, we reviewed different approaches based on *Intel SGX*[27] for their security features and performance overhead (see Section 3.3). Particular attention is paid to *inter-enclave communication* (required for the communication between the user and provider enclaves) and *disk I/O calls* (required for the possible caching of encrypted user data or data stores and provider code including protected AI algorithms and models). The former should not use untrusted memory, otherwise the usage pattern can reveal application-specific control flow. The latter must preserve confidentiality and integrity. We decided to use *SGX-LKL*[28], as it supports these requirements for the `PDSProxy` concept by reducing the number of host call parameter and only shows a minimal overhead compared to related approaches [Pri+20]. The I/O performance depends in particular on whether in-memory stores (such as *Memcached*) are used, as only the integrity of a read-only disk must be protected here, or whether it writes to disk, and other integrity protection must be integrated. In addition to this, *Intel SGX SDK*[29][30] and the *Intel Attestation Service*[31] were used for the `PDSProxy` prototype.

Although this implementation is limited to Intel x86 architectures, the proposed concepts can be transferred to other architectures with the corresponding trusted execution environments. For example, for resource-limited ARM-based devices such as smartphones, Raspberry PI, or home routers [Kau+18] (the latter will still be relevant in Chapter 7 as hub for a possible decentralized urban infrastructure), lightweight ARM TrustZone[32] (as it offers SGX-similar features [Bay+20]) and platform-specific sandboxing techniques can be used.

5.4 METHODOLOGY

This section presents the evaluation methodology of the proposed `PDSProxy` concept. The conducted experiments with the `PDSProxy` prototype implementation

[25] https://cloud.google.com/blog/products/identity-security/open-sourcing-gvisor-a-sandboxed-container-runtime (retrieved 06/30/2021)
[26] https://github.com/google/gvisor (retrieved 06/30/2021)
[27] https://www.intel.com/content/www/us/en/architecture-and-technology/software-guard-extensions.html (retrieved 06/30/2021)
[28] https://github.com/lsds/sgx-lkl (retrieved 06/30/2021)
[29] https://github.com/intel/linux-sgx (retrieved 06/30/2021)
[30] https://software.intel.com/content/www/us/en/develop/topics/software-guard-extensions/sdk.html (retrieved 06/30/2021)
[31] https://software.intel.com/content/www/us/en/develop/topics/software-guard-extensions/attestation-services.html (retrieved 06/30/2021)
[32] https://developer.arm.com/ip-products/security-ip/trustzone (retrieved 06/30/2021)

(see Section 5.3) examined the following efficiency aspects (with corresponding hypotheses). In particular, the experiments focused on if and how ...:

1. ... use cases affect the initialization time of `PDSProxy`, and which are the *use case-specific factors* that affect the initialization time.
   - $H_1^{1.1}$ – $H_1^{1.4}$: Different sized AI services (of the use cases) require different amounts of time in each initialization step (①–④).
2. ... the *initialization mode* and the *trustworthiness* of the underlying device affect the setup time of `PDSProxy`.
   - $H_1^{2.1}$: The different initialization modes reduce the total setup time.
3. ... the *trustworthiness* of the underlying device affect the caching and streaming performance of `PDSProxy`.
   - $H_1^{3.1}$: An untrusted node causes a higher read time from its cache than its trusted equivalent.
   - $H_1^{3.2}$ – $H_1^{3.4}$: An untrusted sender (3.2)/ forwarding node (3.3)/ receiver (3.4) causes a higher transfer time than its trusted equivalent.
4. ... the data protection and IP protection mechanisms of `PDSProxy` affect the runtime performance of AI models in the training and inference phases.

The experiments were performed in the following test scenarios with well-known datasets and off-the-shelf devices.

5.4.1 *Test Scenarios*

The test scenarios were constructed with a view of covering the four ad-hoc deployment configurations introduced in the motivation of this chapter (see Figure 5.1, p. 80) and to allow a performance comparison for two different implementation languages (Java, C++) for one deployment configuration – these considerations result in the following *five test scenarios*. These scenarios cover the most widespread application classes (see Section 2.1.5 and Figure 2.1, p. 16 ff.) and the most important types of machine learning algorithms (see Section 2.1.1); the availability of both suitable community datasets and required AI algorithms or trained/ready-to-use AI models also played a key role in the selection process for the use cases (see Table 5.2 for an overview).

5.4.1.1 *UC1: A Smart Home Use Case*

This use case scenario comprises an AI service for person identification, e.g., to authenticate people unobtrusively in smart home settings, or to provide them with personalized ambient support [Sad11]. In essence, this AI service is based on *facial recognition*; the extracted test procedure is as follows: the provider trains a general model from public/volunteer data in the cloud in advance and delivers it encrypted in step ④ to the local provider enclave. The general model can extract and detect faces in a video stream. Locally, the AI service obtains the privacy-sensitive video stream of the home environment from the built-in

sensors of the target device – or from sensors of connected devices. In addition, the `PDSProxy` on the target device can access the locally stored personal data (⑤), e.g., faces of the house residents with identification information. With this data, a local matching algorithm can compare the faces recognized from the video stream by the general model to identify 'registered' persons, personalizing parts of the AI service to them (⑥). All steps have now been performed once – the test procedure ends here.

To this end, the UC1 test setup uses the *AT&T face recognition database*[33] (formerly 'The ORL Database of Faces')—comprising 40 users with ten different pictures, which vary in background, lighting conditions, or face expressions [Sam94]. The test scenario further relies on a Java-based implementation of the *Fisherface* method [JL11] as the underlying AI algorithm. [MBM20]

5.4.1.2 *UC2: A Fitness & Health Use Case*

This use case scenario comprises an AI service for activity support, e.g., to personalize the training to the individual ambitions [Sch+15b], or to intervene if the activity level is too low [Rab+15]. In essence, this AI service is based on *activity recognition*; the extracted test procedure operates as follows: analogous to UC 1, the provider trains a general model in the cloud and delivers it encrypted to the local provider enclave (④). The general model can recognize different physical activities – more details below. Locally, the general model is then personalized to the user (⑥), with the personal (labeled) data being accessed in step ⑤. All steps have now been performed once – the test procedure ends here.

To this end, the UC2 test setup uses the *WISDM human activity recognition*[34] dataset—comprising 36 users who carry a smartphone in a lab environment while performing a specific set of six simple activities: {*walking*, *jogging*, *upstairs*, *downstairs*, *sitting*, *standing*} [KWM11]. The sampling rate of the accelerometer was 20 Hz. The data is divided into 10-second segments from which the features are extracted. The resulting dataset finally contains 43 features and 5,418 instances, i.e., 150.5 ± 44.7 instances per user. The test scenario further relies on a Java-based implementation of *Patching*[35] [KF18] as the underlying AI algorithm. [MBM20]

5.4.1.3 *UC3: A Transportation Use Case*

This use case scenario comprises an AI service for multimodal trip support, e.g., to suggest routes tailored to individuals by learning about their transport preferences [Lia+07; Zhe+08b]. In essence, this AI service is based on *transportation mode recognition*; the extracted test procedure and the methods used are analogous to UC2 – only with the following difference: the underlying AI models instead recognize the transportation modes used by individuals.

[33] https://www.face-rec.org/databases (retrieved 06/30/2021)
[34] http://www.cis.fordham.edu/wisdm/dataset.php (retrieved 06/30/2021)
[35] https://github.com/Shademan/Patching (retrieved 06/30/2021)

Table 5.2: Selected use cases (UC) and the extracted test objectives – characterized by their data set, underlying AI algorithm, and programming language (PL).

UC	Test objective	Dataset	AI algorithm	PL	References
UC1	Face identification	AT&T	Fisherface	Java	[Sam94; JL11]
UC2	Activity recognition	WISDM	Patching	Java	[KWM11; KF18]
UC3	Transportation recog.	GeoLife	Patching	Java	[Zhe+09; Zhe+08a; KF18]
UC4	Place detection	GeoLife	DBSCAN	C++	[Zhe+09; ZXM10; Est+96]
UC5	Information provision	–	–	Java	[Kni+18; DL19; Müh+20]

To this end, the UC3 test setup uses the *GeoLife GPS trajectory*[36] dataset from Microsoft—comprising 26 selected users who logged their GPS trajectories and labeled enough of them with four selected transportation modes: {*walk*, *bike*, *bus*, *car*} [Zhe+09; Zhe+08a]. A sample is typically recorded every 1-5 seconds or every 5-10 meters, whichever occurred first. The dataset contains 3,960,231 location values, i.e., 152,316 $\pm$ 151,410 locations per user. The dataset transformed for UC3 finally contains 12 features and 7,511 instances, i.e., 288.9 $\pm$ 287.3 instances per user. The test scenario also relies on a Java-based implementation of *Patching*[35] [KF18] as the underlying AI algorithm. [MBM20]

5.4.1.4 *UC4: A Smart Car Use Case*

This use case scenario comprises an AI service for driver support, e.g., to proactively provide navigation or refueling recommendations based on the driver's mobility [Nak+12]. In essence, this AI service is based on *place and routine detection*; the extracted test procedure operates as follows: the provider transfers the confidential business logic and a protected clustering algorithm encrypted to the local provider enclave (④). Locally, with the above, a personal model is trained to identify significant places and intermediate routes from past trip data (⑤), learning the routines of the user (⑥). All steps have now been performed once – the test procedure ends here. Taking this use case as an example, it is important to mention that `PDSProxy` would work equally for the user's own car as well as for a newly-borrowed (foreign) car, providing an identical support, as the data is decoupled from the service and the device.

To this end, the UC4 test setup uses the same dataset[36] as that of the previous use case (UC3) – but only the location values are relevant here [Zhe+09; ZXM10]. The test scenario further relies on a C++-based implementation of *DBSCAN* [Est+96] as the underlying AI algorithm. [MBM20]

5.4.1.5 *UC5: A Signpost Use Case*

Last but not least, this use case scenario follows the example of Figure 5.1c: the service (not relying on AI models this time) just provides personalized information and orientation guidance, e.g., on public displays [DL19] or via projections [Kni+18] to find the way in airports and train stations [Müh+20].

[36] https://www.microsoft.com/en-us/download/details.aspx?id=52367 (retrieved 06/30/2021)

The extracted test procedure is as follows: the provider transfers the confidential business logic/data encrypted to the local provider enclave (④). Locally, the service requires access to user's current position(s) and travel data (⑤). Based on this data, the service 'only' looks up user-relevant information (e.g., gate, boarding and departure time), calculates the direction to the gate and the user's arrival time based on the current walking speed, to name just a few (⑥). All steps have now been performed once – the test procedure ends here.

To this end, the UC5 test setup uses a sample path of a user at Frankfurt Airport (FRA) and sample travel information of the passenger as confidential user data. The public data on the device includes a one-level map of the airport with gate information (and a sample flight schedule). Additionally, one can image other (public and confidential) data that might be of interest to passengers. The test scenario further relies on an optimized Java-based implementation to obtain only the user-relevant information from the data described above. [MBM20]

5.4.2 *Experimental Setup and Apparatus*

For the experiments, an Intel NUC test device hosts the user's base `DC-PDS`, and Udoo Ultra single-board computers (SBCs) are configured to run the `PDSProxy` instances. The former device is comparable to today's smart home servers or NAS systems; it is equipped with an SGX-enabled 7th-generation Intel Core i5-7260U dual-core (2/4) x86-processors@2.2GHz (up to 3.4GHz), 16 GB DDR4-2400-RAM, 500GB SSD, Bluetooth 4.0 (low energy), and WiFi 5 (802.11ac). The latter device is comparable to the hardware integrated into (IoT/edge) devices [Müh+20]; it comes with an SGX-enabled Intel Pentium (N3710) quad-core (4/4) x86-processors@1.6GHz (up to 2.56GHz), 8 GB DDR3L-1600-RAM, 128GB SD card, Bluetooth 4.0 (low energy), and WiFi 5 (802.11ac). All test devices have Internet and are connected to the cloud with 89.23 ± 3.81 MBit/s.

5.5 RESULTS

This section reports and discusses the evaluation results with respect to the efficiency aspects and corresponding hypotheses described in Section 5.4. The setup and initialization phases were examined first (see Section 5.5.1), followed by the confidential operation phase (see Section 5.5.2).

5.5.1 *Efficiency I: Case-specific Setup Times*

The first set of experiments analyzes the additional setup (Ⓐ–Ⓓ) and initialization (①–④) times by measuring the times for each step. Each measurement is repeated five times; the results are then averaged – which will be reported next.

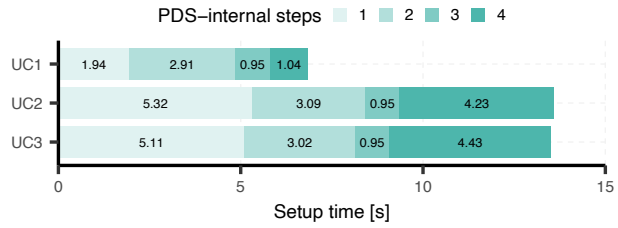


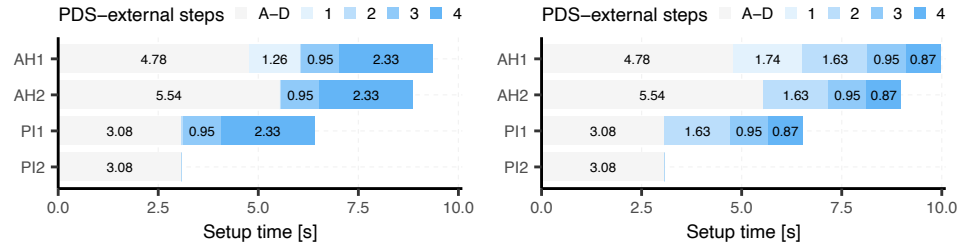
Figure 5.6: Initialization times for AI services of UC1-3 within the PDS-internal confidential processing environment – identical for both types of nodes, namely trusted and untrusted nodes

5.5.1.1 *Initialization Times for PDS-internal Processing*

In this first experiment, we examined if and how use cases affect the initialization time of `PDSProxy`, and which are the *use case-specific factors* that affect the initialization time. We also tested the hypotheses $H_1^{1.1}$-$H_1^{1.4}$ (see Section 5.4). To avoid network side effects (in the setup phase), we limited this experiment to the PDS-internal confidential processing environment, examining only the initialization times (see Section 5.1). We used the AI services of the three different use cases UC1–3 that are suitable for this *internal* environment due to their nature (see Section 5.4.1): the sizes of the public enclave code (relevant in step ①) and the confidential AI models (relevant in step ④) delivered by providers are as follows: UC1 (①: 17.3 MB, ④: 0.7 MB), UC2 (①: 58.6 MB, ④: 8.5 MB), and UC3 (①: 58.6 MB, ④: 8.3 MB) – note that UC2 and UC3 rely on the same underlying AI algorithm.

Figure 5.6 shows the initialization times for the AI services of UC1–3 in PDS-internal environments. We can see that these total initialization times vary strongly between the different use cases. To examine each setup step in more detail, we test the hypothesis associated with each step (see Section 5.4). As determined by one-way ANOVA, the analysis showed that there is *no* statistically significant difference for step ② ($F_{2,97.8} = 2.76$, $p = .068$) and step ③ ($F_{2,97.3} = 1.61$, $p = .205$) – this means that we fail to reject the corresponding null hypothesis and cannot accept the alternative hypothesis $H_1^{1.2}$ (related to step ②) and $H_1^{1.3}$ (related to step ②). On the other hand, the analysis showed a statistically significant difference for step ① ($F_{2,79.9} = 1556$, $p < .001$, accepting $H_1^{1.1}$) and step ④ ($F_{2,70.3} = 2487$, $p < .001$, accepting $H_1^{1.4}$) as determined by one-way ANOVA. Post hoc tests with a Bonferroni correction each ($\alpha = .05/3 = .0167$) revealed that the setup times of step ① and step ④ for UC1 are statistically significantly lower than those for UC2 and UC3 (both $p < .001$); there is no statistically significant difference between the setup times of step ① ($p = .415$) and step ④ ($p = .018$) for UC2 and UC3.

Based on this analysis, we can say that the steps ②+③, which concern the setup of the enclave is use case *independent* (i.e., independent of the characteristics of the AI service). Only the initialization times in steps ① and ④, which concern the download of the public enclave code from the store and the confidential AI models from the provider-trusted server, differ significantly between use cases when the respective sizes differ – this means that both steps depend on the use



(a) Initialization modes (UC4)

(b) Initialization modes (UC5)

Figure 5.7: Setup and initialization times for AI services within the PDS-external confidential processing environment (`PDSProxy`) on untrusted nodes

case or the characteristics of the AI service (i.e., sizes of enclave code and AI models). In addition, the initialization times of steps ①, ③ and ④ depends on the connectivity to the store and provider-trusted server – this offers optimization potential that is exploited by the proposed initialization modes, which we will evaluate in the next experiment (see Section 5.5.1.2).

All in all, the initialization time depends on various (partly use case dependent) factors that must be taken into account during development. In absolute terms, the initialization times of the protection mechanisms are sufficient for non-time critical use cases in which the user is (nearly) 'static' in the reference system of the device (see Figure 5.1a+b, p. 80); however, it takes too long for mobile scenarios in which the contact times are only in the low-second range.

5.5.1.2 *Setup and Initialization Times for PDS-external Processing*

In this second experiment, we examined if and how the *initialization mode* (described in Section 5.5.1.2) affect the setup and initialization times of `PDSProxy` and tested the hypothesis $H_1^{2.1}$ (see Section 5.4). As the different initialization modes are designed to deal with the different contact times in the ad-hoc deployment configurations (see Figure 5.1, p. 80), we used the AI services of the two different use cases UC4–5 that (i) cover these configurations and (ii) are suitable for this *external* environment due to their nature (see Section 5.4.1).

Figure 5.7a (UC4) and Figure 5.7b (UC5) show the setup and initialization time results for the different *initialization modes*. There is a statistically significant difference between the different initialization modes across both use cases as determined by non-parametric Friedman test [Demo06], $\chi^2(3) = 138$, $p < .001$. This means that we can reject the corresponding null hypothesis and accept the hypothesis $H_1^{2.1}$. Wilcoxon signed-rank post hoc tests with a Bonferroni correction ($\alpha = .05/4 = .0125$) confirmed this result, revealing statistically significant differences in *all* pairwise comparisons ($p < .001$), as shown below. More precisely, using the example of UC4, the ad-hoc mode AH2 ($\mu = 8.861s$, $\sigma = 0.712s$) statistically significantly reduces the total setup and initialization time compared to AH1 ($\mu = 9.355s$, $\sigma = 0.709s$). While both *ad-hoc modes* do not require any prior setup of the target system, offering a high flexibility, they take too long for use case scenarios with one-time contacts of a few seconds (see

Figures 5.1c+d). To further optimize this setup in terms of time, a certain level of pre-configuration is required, which would in turn restrict the flexibility of `PDSProxy` (similar to other/traditional edge computing approaches but not with advanced data and especially IP protection mechanisms) – in `PDSProxy`, this is realized with two *pre-init modes* (see Table 5.1). Both *pre-init modes* statistically significantly reduce the total setup and initialization time compared to the *ad-hoc modes*. Specifically, the pre-init mode PI2 ($\mu = 3.086s$, $\sigma = 0.673s$) has a statistically significantly lower value than PI1 ($\mu = 6.406s$, $\sigma = 0.685s$). The initialization modes in UC5 behave analogously.

In terms of *trustworthiness* of the underlying devices, `PDSProxy` offers two different *proxy variants*: one for trusted devices (only requiring one setup step, namely step Ⓐ), and one for untrusted devices (requiring all four setup steps Ⓐ–Ⓓ). The difference between both proxy variants lies in the setup of a user enclave (Java implementation), which took on average $0.99s$ ($\sigma = 0.03s$) in the experiment. There is no statistically significant difference between the setup times of the user enclaves for all use cases considered in the given experimental setup as determined by Friedman test, $\chi^2(1) = 0.532$, $p = .466$.

For setting up the provider enclave (②), which is use case independent (i.e., independent of the characteristics of the AI service, as shown above), the crucial parameters are the network conditions (e.g., bandwidth), computing power, and the implementation language used. For example, the C++-based implementation used in UC4 can set up the provider enclave fast ($\mu = 0.042s$, $\sigma = 0.002s$), much faster than the Java-based implementation used in UC5 (($\mu = 1.629s$, $\sigma = 0.014s$)) – C++ should therefore be the preferred language in use cases in which the initialization time is critical.

Nonetheless, establishing an ad-hoc connection to a nearby device ($\mu = 1.850s$, $\sigma = 0.646s$; included in step Ⓐ) remains the limiting factor. For example, for the initialization variant PI2, this step took the most time (88.5% and 60.1% of the whole setup time for trusted and untrusted nodes, respectively) – the same issue also arises with (other) edge computing approaches. For this reason, and since further improvements are beyond the scope of this thesis, we refer to current network research: Apple's AWDL ad-hoc protocol – reverse engineered and made available as open source by Stute and others [SKH18] – in particular represents a promising starting point for future work.

All in all, the proposed setup optimizations in form of the proxy variants and the initialization modes can be successfully adapted to the (trusted or untrusted) systems involved and the requirements of the respective use case – we showed a significant reduction in the setup and initialization times of `PDSProxy` without compromising the data and IP protection mechanisms.

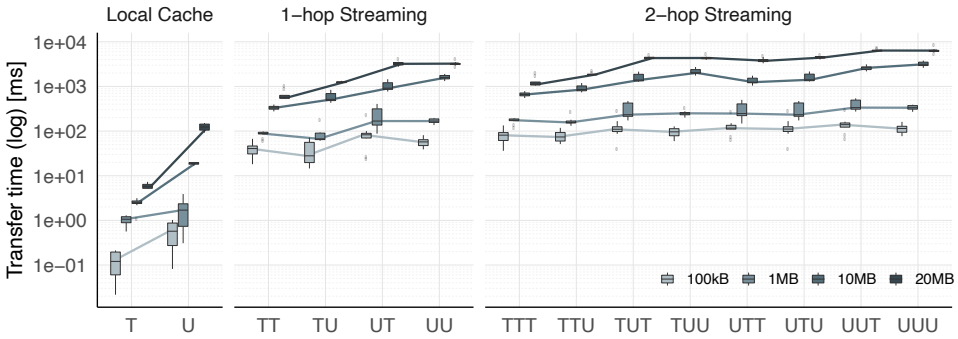


Figure 5.8: Streaming performance overhead (⑤) of `PDSProxy`: a chain is encoded with U and T for every un-/trusted node respectively; the streaming direction is from left to right, i.e., the first (or the leftmost) node in the chain is the sender, which loads the data from its local cache/store, and sends it to the last (or the rightmost) node in the chain via all intermediate nodes (if any).

5.5.2 *Efficiency II: Runtime Performance*

The second set of experiments evaluates the runtime performance in the operation phase (⑤–⑥). Again, each measurement is repeated five times; the results are then averaged – which will be reported next.

5.5.2.1 *Caching and Streaming Performance on Un-/Trusted Nodes*

In this experiment, we examined if and how the *trustworthiness* of the underlying device affect the caching and streaming performance of `PDSProxy` and tested the four hypotheses $H_1^{3.1}$–$H_1^{3.4}$ (see Section 5.4).

Figure 5.8 shows the caching and streaming performance (⑤) across several constructed chains of un-/trusted nodes: for simplicity, we encode a trusted node with T and an untrusted node with U; for example, a chain of three nodes (trusted node → untrusted node → untrusted node) is then encoded as TUU, whereby the leftmost node is the sender that loads the data from its local cache/store and sends it to the rightmost node in the chain via all intermediate nodes (if any). Nodes running `PDSProxy` can connect ad hoc over WiFi; the first hop in these constructed chains with more than two nodes goes over the Internet from the first node, which is always the base `DC-PDS` (leftmost node).

With this setup, we can first of all see a low performance overhead for *local caching*, lying in the low millisecond range; only for larger data sizes (10MB+) on an untrusted node, the read time increases more strongly due to the encryption required. As determined by Friedman test, $\chi^2(1) = 25.6$, $p < .001$, the read times on untrusted nodes are statistically significantly higher than those on trusted nodes (cf. $\underline{U}$ vs. $\underline{T}$) – this means that we can accept the hypothesis $H_1^{3.1}$.

We can also see that the requesting and streaming of data from another node further increase the transfer time obviously due to the additional network

operations. Here, the number of nodes involved is not as crucial as the *trustworthiness* and the *role* of the node involved. As determined by Friedman test, $\chi^2(1) = 64.8$, $p < .001$ (accepting $H_1^{3.2}$), an untrusted sender causes a higher transfer time than its trusted equivalent (cf. the relevant chains: $\underline{T}T$ vs. $\underline{U}T$, and $\underline{T}U$ vs. $\underline{U}U$). The same is true for forwarding nodes: as determined by Friedman test, $\chi^2(1) = 46.2$, $p < .001$ (accepting $H_1^{3.3}$), an untrusted forwarding node causes a higher transfer time than its trusted equivalent (cf. the relevant chains: $T\underline{T}T$ vs. $T\underline{U}T$, $U\underline{T}U$ vs. $U\underline{U}U$, $T\underline{T}U$ vs. $T\underline{U}U$, and $U\underline{T}T$ vs. $U\underline{U}T$). For the receiver role, we fail to reject the corresponding null hypothesis and cannot accept the alternative hypothesis $H_1^{3.4}$: as determined by Friedman test, $\chi^2(1) = .2$, $p = .655$, there is no statistically significant difference between untrusted and trusted receivers (cf. the relevant chains: $T\underline{T}$ vs. $T\underline{U}$, and $U\underline{T}$ vs. $U\underline{U}$).

For example, in a two-node chain (e.g., personal mobile device → edge device), if the sender is a trusted node, there is no significant streaming performance overhead, regardless of the trustworthiness of the receiver. In a three-node chain (e.g., home server/private cloud → personal mobile device → edge device), there is only a significant performance overhead if the first two nodes in the chain are not trusted. In practice, however, the sender is typically a trusted node fully controlled by the user. In such a case, the streaming of user data with a size up to 10MB took place in the millisecond range regardless of the trustworthiness of the other nodes in the two- or three-node chain within this experiment – even time-critical use cases in the low-second range can be handled in this setting.

All in all, depending on the setting, i.e., only if the sender and forwarding nodes are untrusted, `PDSProxy` can cause performance overhead for both local caching and data streaming. In all cases, however, user data is *strongly* protected against the providers and, on untrusted devices, against them as well.

5.5.2.2 *Local Operation Performance with Maximal Data and IP Protection*

In this experiment, we examined if and how the data protection and IP protection mechanisms of `PDSProxy` affect the runtime performance of AI models in the training and inference phases. To this end, we consider all use cases (see Section 5.4.1), as all corresponding AI services go through these steps.

Table 5.3 shows the operation performance of AI services (⑥) within `PDSProxy`. This experiment distinguishes between (a) the local *training task*—i.e., the local personalization step—and (b) the *inference task*, both of which are typical processes for AI models (see Section 2.1.2) [MBM20]. It is important to note that our goal in this experiment is not to improve any AI algorithms/models – quite the contrary, these are only considered as black boxes; the effects of the proposed protection mechanisms are the focus of this analysis.

Compared to the baseline, where user data and provider code are unprotected, they are *strongly* protected from other parties in the confidential processing environments of `PDSProxy`. This comes at the expense of AI services taking

Table 5.3: Operation performance overhead (⑥) of PDSProxy compared to the non-confidential case (*baseline*) on exemplary use cases (UC) – see Table 5.2.

	(a) Training task ('local share')			(b) Inference task		
UC	*Baseline* $[s]$	*PDSProxy* $[s]$	*Overh.*	*Baseline* $[s]$	*PDSProxy* $[s]$	*Overh.*
UC1	3.905 ± 0.138	15.073 ± 2.162	2.86	0.002 ± 0.001	0.007 ± 0.003	2.50
UC2	0.770 ± 0.004	2.913 ± 0.140	2.78	0.040 ± 0.005	0.196 ± 0.019	3.90
UC3	0.190 ± 0.007	0.690 ± 0.039	2.63	0.004 ± 0.000	0.009 ± 0.003	1.25
UC4	0.670 ± 0.000	0.990 ± 0.000	0.47	0.090 ± 0.000	0.140 ± 0.010	0.55
UC5	–	–	–	0.003 ± 0.000	0.008 ± 0.000	1.67

longer in such an environment: their overhead ranges between 2.63 and 2.86 for the training tasks and between 1.25 and 3.90 for the inference tasks in Java-based implementations; in the C++-implementation (see UC4), the overhead is only about 0.5 for both tasks. The overhead is generally caused by the additional cryptographic operations to encrypt the memory for both the user and provider enclaves. As this memory per enclave is very limited (concretely, maximal 96MB for the SGX implementation used), *swapping processes* are the main reason for the overhead. In the use cases, these occur mainly due to the memory-intensive programming language used (e.g., Java in UC1-3), a high computational complexity of AI algorithms (see UC2+3) and the size of user data (e.g., images in UC1). Although UC4 uses an algorithm with a high computational complexity $O(n^2)$ (in the worst case) and larger input data sets, the overhead of the C++-implementation is comparatively low, i.e., a memory-intensive programming language such as Java (and its runtime environment) contributes the most to the overhead.

Although PDSProxy entailed an overhead in all use cases, the execution times for the inference tasks are in absolute terms in the low one- to three-digit milliseconds range ($< 200ms$). In this respect, these low execution times still make PDSProxy suitable for most scenarios including mobile scenarios, despite the overhead. However, depending on scenario (e.g., ad-hoc deployment scenario with short contact times in the low second range – see Figure 5.1c+d), the training overhead is too high. In such scenarios, PDSProxy requires runtime optimizations; the following three ways can be considered for this. (1) Developers can optimize their AI services to avoid swapping processes. For example, developers can only execute necessary and worth protecting parts of the AI service in the enclave. They can also use a lightweight programming language and seek to reduce the computational complexity and the size of the data required for their services. (2) Resource-intensive training tasks can also be distributed across multiple enclaves, as proposed in [EN20] – this can accelerate the training up to 4.7x (but additional setup effort must then be considered). In Chapter 8, we will propose another way of proactive deployment to conceal the inherent setup and initialization overhead (including the training or personalization of AI models). Apart from that, this thesis refers to current security research: (3) we strongly assume that the available memory capacity of

Table 5.4: A summary assessment of the contributions made in this chapter based on the requirements established in Section 3.3.1 – a comparison with the state of the art protection approaches *at system level* is given by Section 3.3.2 and Table 3.2

	Data/Code Protection			*Performance*			*Applicability*			
Contribution	Integrity (*code/data*)	Confidentiality (*code/data*)	Protection Against Untrusted Code	Low Performance Overhead	Initialization Optimization	Runtime Optimization	Data Streaming (*local/1-/n-hop*)	Support of Any AI Algorithm	Support of Any Use Case	Support for Ad-hoc Deployment
`PDSProxy`†‡⅄⊀	●/●	●/●	●	◐	●	○	●/●/●	●	●	●

Encoding: **fulfillment of requirements** (see Section 3.3.1): ●–fulfilled, ◐–partially fulfilled, ○–little or not fulfilled; **HW/SW prerequisites**: ⅄–TEE (e.g., SGX, TrustZone), ⊀–sandbox, ‡–PKI; **primary development context**: †–specially designed for AI algorithms.

an enclave will continue to increase in future—leading to reduced swapping processes and to a better support of resource-intensive AI services.

All in all, while user data and the provider code are always *strongly* protected against the other parties involved, this is at the expense of an overhead for both training and inference tasks in AI algorithms. While this is not an issue for non-time-critical use cases (e.g., smart home use cases), as the execution times of the user-experience-critical inference tasks are in the low millisecond range, in other cases (e.g., in mobile scenarios) a resource-intensive training still needs to be optimized or distributed over several enclaves.

5.6 CONCLUSION

This chapter presents `PDSProxy`, a confidential distributed processing environment that protects both user data and providers' intellectual property *at system level*. To this end, `PDSProxy` combines *sandbox* and TEE approaches (based on *enclaves*) together with case-specific initialization optimizations, achieving a unique tradeoff between data/code protection, performance, and applicability.

5.6.1 *A Unique Tradeoff at System Level*

Table 5.4 shows a summary assessment of `PDSProxy` based on the requirements established in Section 3.3.1 – supported by the evaluation provided in this chapter. A comprehensive comparison with the state of the art can be made

using Table 3.2 (p. 44), confirming the unique tradeoff achieved by PDSProxy.

In terms of data and code protection, related work can ensure the integrity and confidentiality of both using a single enclave, but then the provider code has to take care of the data retrieval (e.g., streaming user data from other devices), which makes it inflexible; otherwise the retrieval part is unprotected against the system (see Section 3.3). In addition, only a few approaches integrate mechanisms to protect user data from untrusted provider code. PDSProxy contributed a concept to address this *three-party confidentiality challenge* by establishing two separate enclaves, namely a user enclave (to protect the user data against the system) and a provider enclave (to protect the provider code against the system and the user) instantiated by the user enclave. Inter-enclave communication allows the confidential exchange of user data. Sandboxing of the provider enclave provides protection against untrusted provider code and the control of outgoing system calls (e.g., to prevent leakage of confidential user data). In this way, PDSProxy locally establishes a confidential processing environment that are attested by both parties, temporarily expanding the user territory and the provider territory, even to untrusted (third-party) devices. Data and code are transferred there only in encrypted form – this ensures integrity and confidentiality of data and code at system level.

In terms of performance, similar to related work, PDSProxy entails an overhead that is caused by the protection mechanisms – this overhead is only low for related works that are specialized (e.g., protecting passwords only [Kra+18]) and do not cope with the three-party confidentiality challenge. Unlike related work, PDSProxy contributed two initialization optimizations to speed up the initialization process without compromising protection, which is particularly relevant for ad-hoc deployment of distributed/device-bound AI services on nearby devices. *Proxy variants* as the first optimization slim down the setup process depending on the trustworthiness of the underlying system. Different *initialization modes* as the second optimization provide either flexibility or a fast setup process depending on the use case requirements. The evaluation showed a significant reduction in setup and initialization times across the use cases.

In terms of applicability, PDSProxy is particularly designed to support all phases of AI services, but it is generic in a way that it can also be applied to other applications. Similar to most of the related work, PDSProxy is not restricted to a specific AI algorithm or use case. To apply PDSProxy, it also requires a minimum of prerequisites on the target devices, namely TEE hardware support and compatibility with a sandbox solution (see Section 5.2.1). Compared to related work (see Section 3.3.2), PDSProxy is the first approach that enables an ad-hoc deployment of these strong protection mechanisms on nearby devices (e.g., in mobile scenarios) and supports hierarchical data caching and streaming across multiple untrusted and trusted nodes in a fully confidential way.

5.6.2 *Integration & Outlook*

The protection mechanism presented in this chapter constitutes a technical measure against untrusted underlying systems – this is not only limited to the user's local system running the personal data store (see `DC-PDS` introduced in Chapter 4), but also as a new achievement *ad hoc* on other third-party devices (e.g., IoT/edge devices) without compromising the protection offered. Specifically, `PDSProxy` expands the territories of both the user and the provider to systems untrusted for them to ensure the protection of user data and provider code, addressing the three-party confidentiality challenge.

All in all, `PDSProxy` contributes confidential processing environments and mechanisms to ad hoc deploy them, addressing the second basic requirement (`N2`) of distributed and in particular device-bound AI services. However, these protection mechanisms do *not* allow (authorized but untrusted) providers to improve their shared AI models by this user data in order to provide more efficient AI services for new users. This issue will be addressed next in Chapter 6.

6

PRIVACY-ENHANCING PERSONALIZATION OF AI SERVICES

The contributions from the previous two chapters dealt with the protection of personal data at *management level* (see Chapter 4) and at *system level* (see Chapter 5). This chapter contributes to the protection of personal data at *AI level* – the categorization scheme is described in Section 3.1 (see also Figure 3.2, p. 33).

> *"Yet advancing AI by collecting huge personal profiles is laziness, not efficiency.* [..] *For artificial intelligence to be truly smart, it must respect human values, including privacy. If we get this wrong, the dangers are profound.* [..] *We can achieve both great artificial intelligence and great privacy standards. It's not only a possibility, it is a responsibility. In the pursuit of artificial intelligence, we should not sacrifice the humanity, creativity, and ingenuity that define our human intelligence."*
>
> Tim Cook, CEO, Apple, 2018 [She18]

With these words, Tim Cook once again characterizes the ongoing and indispensable need for data protection mechanisms in AI services. He particularly highlights the challenge of providing effective and efficient AI services while protecting user privacy – competing requirements that have been identified and elaborated in Section 3.4.1. For example, *personalization* is a common and important way to improve the performance of AI models for individuals (see Section 2.1.3), but it may lead to efficiency issues, if it is performed locally on the user's device, and to privacy issues, if it is performed elsewhere [Ser+18].

As shown in the review of related work in Section 3.4.2, none of the latest data protection approaches can fulfill all these category-specific requirements to a high extent at the same time. For instance, *data modifying* approaches (e.g., based on differential privacy [Dwo06; Aba+16]) have a low overhead (efficiency), but their inherent conflict between the usefulness of personal data and user privacy remains unsolved. While *data-encrypting* approaches (e.g., based on HE [Gen09; Dow+16] and SMC [Pin02; MZ17]) are perfectly suited to ensure confidentiality and integrity of data, low computational efficiency for a limited set of operations limits their applicability. *Data-minimizing* approaches, on the other hand, are more diverse: some (e.g., CSN [Lan+11]) can achieve high effectiveness while minimizing the amount of personal data required in the cloud-based training – but user privacy cannot be guaranteed, as (raw) personal data leaves the user territory; others (e.g., Google's Federated Learning [McM+17]) only require to share model parameters/ weights of locally trained models – but the effectiveness of the aggregated model is comparable to that of a general model, i.e., relatively low for some users compared to the effectiveness of their personal models. Last but not least, *data-confining* approaches (e.g., local training [Ser+18]) are best suited to achieve high (personalization) accuracy while preserving user privacy: they can give AI

services full access to local data that never leaves the user territory – but low efficiency (e.g., labeling effort, local resource use) remains an open issue. [MM21]

In this context, we focus on AI services relying on *supervised learning* algorithms (see Section 2.1.1) and, in particular, address the challenge of achieving high efficiency in local *classification* tasks by reducing both the labeling effort for users and local resource use. The contribution of this chapter is twofold. First, the chapter presents the `{P}Net` concept, which (i) enables personalized AI services while protecting user privacy through local data confinement, and (ii) addresses the cold-start problem for new users in classification tasks by anonymously sharing model updates based on inter-user similarity (see Section 6.1). Second, the chapter reports and discusses the evaluation results of `{P}Net` on two publicly available sensing datasets (see Section 6.2 - 6.4). The results demonstrate four aspects of `{P}Net`: its feasibility, efficiency (in terms of labeling effort and local resource use), effectiveness (in terms of accuracy of the model and usefulness of personal data), and robustness.

Contribution Statement: This chapter is based on the following publication:

– Christian Meurisch, Sebastian Kauschke, Tim Grube, Bekir Bayrak, and Max Mühlhäuser. "{P}Net: Privacy-Preserving Personalization of AI-Based Models by Anonymous Inter-Person Similarity Networks." In: *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. MobiQuitous'19. ACM, Nov. 2019, pp. 60–69. DOI: 10.1145/3360774.3360819

I (*Christian Meurisch*) led the process of idea generation, conceptual design, implementation, evaluation, and writing. The student *Bekir Bayrak* implemented parts of the proof-of-concept prototype and supported the evaluation. *Sebastian Wagner né Kauschke*, *Tim Grube*, and *Max Mühlhäuser* provided helpful critique and comments on the conceptual design.

6.1 THE {P}NET CONCEPT

This section presents the `{P}Net` concept, a data protection approach at *AI level*, which is particularly designed for classification tasks in supervised learning. Figure 6.1 gives an overview of the `{P}Net` architecture. `{P}Net` first assumes a general model (GM), which can be learned in different ways, e.g., 'classically' from voluntary data or in a more privacy-preserving way through differential privacy or federated learning approaches (see Section 3.4.2). Such a general model does not require personal data of the target users; it can work well for many target users but typically not for all (see Table 6.1, p. 124).

Starting from a general model, `{P}Net` comprises two buildings blocks, namely ❶ local personalization (LP) and ❷ community-based personalization (CP), to address this issue in a privacy-enhancing way by exploiting their individual strengths. Both building blocks are now explained in detail below.

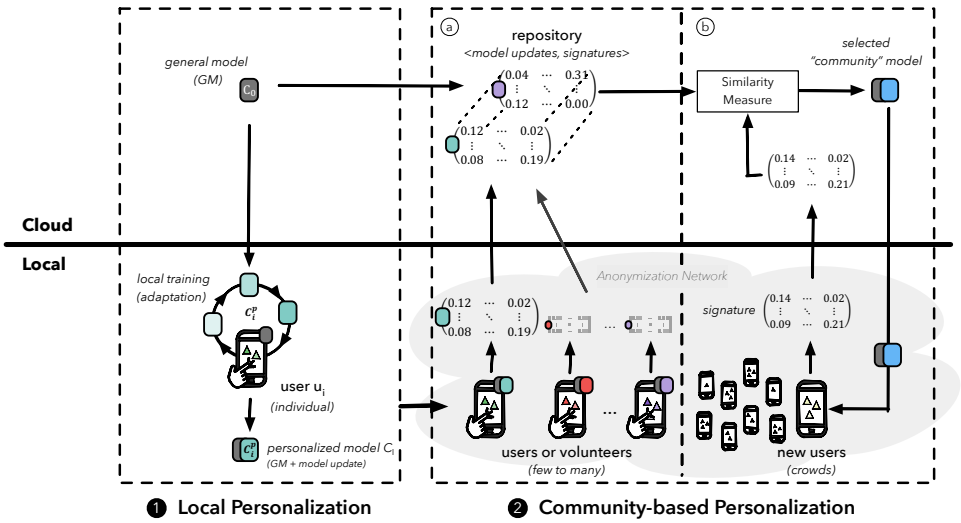


Figure 6.1: Overview of the {P}Net architecture. {P}Net comprises two buildings blocks: ❶ local personalization (LP), and ❷ community-based personalization (CP). The hand symbol indicates a required labeling effort; triangles represent personal data; and rectangles depict AI models (the gray rectangle represents the general model; all others colored rectangles are user-specific model updates).

6.1.1 *Local Personalization (LP)*

This first building block (❶) takes advantage of a 'divisible' AI algorithm to decouple the *personalization* task from the *general model training* task (Fig. 6.1, *left*). Specifically, this building block assumes a personalization algorithm that can *locally adapt* a general model to the respective users based on their personal data (Fig. 6.1, *bottom left*): the adaptation of (general) classification models for individuals can significantly improve performance for them and avoids learning from scratch (i.e., it requires less personal data than training a fully personal model for the same performance), as various research has shown (e.g., [Ser+18]); the local execution ensures the user's privacy, as only the subsequent personalization step requires personal data locally, i.e., no personal data needs to be transferred (see *data-confining* approaches in Section 3.4.2). We refer to this building block as 'local personalization' or abbreviated as LP.

Formally, LP starts with the general model C_0, which is assumed as immutable – this is a prerequisite for the second building block, to which we will come back in the next section. C_0 can classify a general instance space D of instances x with labels $l(x)$ quite well with a high probability $Pr[C_0(x) = l(x)]$. Locally, for a user u_i ($i > 0$, marking user-specific variables below) with another labeling function $l_i(x)$ and underlying (local, user-specific) instance space D_i, C_0 typically makes less accurate predictions:

$$Pr[C_0(x) = l_i(x)] \leq Pr[C_0(x) = l(x)], x \in D_i \tag{6.1}$$

To overcome the worse performance of C_0 for this user, a personalization algorithm with the following characteristics is applied: such an algorithm can train a new model C_i^p on D_i and data from predictions outputted by C_0 so that the resulting ensemble C_i benefits greatly from a small $|D_i|$ and improves the performance for this user as follows:

$$Pr[C_i(x) = l_i(x)] \geq Pr[C_0(x) = l_i(x)], x \in D_i \tag{6.2}$$

In this way, C_0 is adapted to the user with less personal data than training a completely personal model that is only trained on D_i from scratch and achieves the same performance [Ser+18] – this results in a lower labeling effort. Given that C_i^p updates C_0, we refer to it as a *model update*.

We will now exemplify how LP can be performed by using `Patching`[35] [KF18], which represents one potential algorithm of this category of algorithms that fulfill the above conditions. Given C_0, *Patching* locally trains a binary classifier E_i on D_i that can identify specific error regions $R_{i,j}$ $(j > 0)$ in the instance space, in which C_0 incorrectly classifies $x \in D_i$. For these error regions, a new classifier (or internally an ensemble of classifiers) $C_i^p = f(R_{i,j}|C_0)$ is trained. The resulting overall classifier C_i is then composed as follows:

$$C_i(x) = \begin{cases} C_i^p, & E_i(x) = 1 \wedge x \in R_{i,j}(x) \\ C_0, & E_i(x) = 0 \end{cases} \tag{6.3}$$

For a new instance x', C_i decides based on the location of x' in the instance space either to use the model update C_i^p or 'fall back' to the general model C_0. For more technical details on `Patching` beyond the understanding required for this thesis, the reader is referred to [KF18].

All in all, this first building block is characterized by the usual tradeoff of *data-confining* approaches (see Section 3.4): LP provides high performance through a personalized model (in relation to the general model) with full privacy protection (as personal data is only used locally) and a moderate local resource use (as only the lightweight personalization task has to be performed locally). On the other hand, LP still has a *cold-start problem* for new users, as it needs labeled data for the personalization tasks – in case of explicit feedback, users also have a labeling effort that should not be underestimated, as it may lead to a bad user experience.

6.1.2 *Community-based Personalization (CP)*

The second building block (❷) builds on the first one and is the actual contribution of this chapter: it addresses the cold-start problem (concerning the issue of requiring labeled data in supervised learning) and efficiency issues with (local) personalization of classification models for new users. Intuitively, the underlying assumption of this building block is that a personalized model (or a model

update) of one user may also work well for other users who have a similar data distribution as the former – this assumption has to hold for the personalization algorithm used for LP (see Section 6.1.1). By implication, finding a user with similar data characteristics to those of a new user can determine which (shared) personalized model (or model update) can work for this new user, too. We refer to this building block as 'community-based personalization' or abbreviated as CP.

CP comprises two stages (Fig. 6.1, *right*): (a) a few to many users or volunteers anonymously share model updates originating from the LP step to populate a central repository; (b) the mass of *new* users starts directly to request appropriate model updates from this repository to overcome the cold-start problem as mentioned above. In this way, the cold-start problem that a new user may have with data labeling becomes another kind of cold-start problem for the provider in terms of initially populating a repository with model updates (similar to the challenge of training an initial general model). We will now describe the two stages in more detail and discuss possible practices for use (see also Section 6.4).

6.1.2.1 *Populating the Central Repository With Model Updates*

In this first stage, the central repository is populated with model updates and corresponding user-specific *signatures*, which are a kind of characteristics representation of the user data on which the model updates were trained and that still allow similarity comparisons with other users (see Figure 6.1, ❷-a) – these signatures will be used in the next stage to find appropriate model updates for a new user (see Section 6.1.2.2). In the following, we will first explain the generation of such signatures and the anonymous sharing process of model updates to the central repository.

Generation of a User-specific Signature

To decouple each model update C_i^p trained in the LP step from the (personal) training data of the creating user u_i, which would be required for similarity comparisons with other users [Lan+11], we introduce so-called *signatures*. In particular, these signatures should not contain any (raw) user data or it should not be possible to exactly reconstruct personal data from these signatures. At the same time, however, these signatures should still retain inter-user similarity characteristics. To meet these requirements, CP uses Location-sensitive Hashing (LSH)[37] to build inter-user similarity-preserving histogram representations $T(u_i)$ [Ji+12].

Figure 6.2 illustrates the two-step generation process of such signatures. In the first step, LSH is used as a probabilistic dimension reduction method that maps the d-dimensional instance space $\mathbb{R}^d$ to a lower d'-dimensional space $\{0,1\}^{d'}$ $(1 < d' < d)$. In contrast to cryptographic hash functions that seek to avoid hash collisions, LSH relies on a hash function family $\mathcal{H}$ that can preserve the local relations or similarity $sim(x_a, x_b)$ between the data $x_a, x_b \in D_i$ by seeking to

[37] We use *Super-bit LSH*, a LSH implementation that features better similarity-preserving properties by orthogonalizing the projection vectors – compared to other LSH implementations [Ji+12]

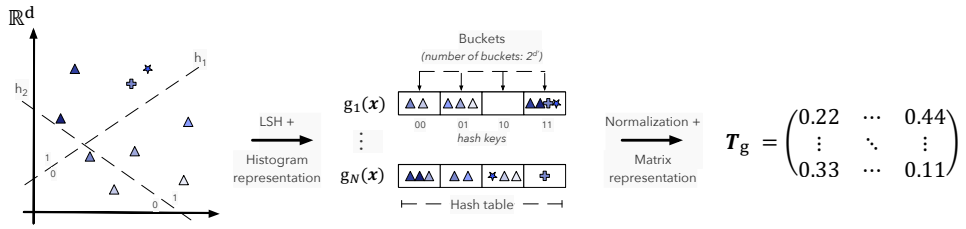


Figure 6.2: Schematic representation of the *signature* (T_g) generation. In this example, the data space (*left*) has the dimension d=2; only the hash function $g_1(x) = [h_1(x), h_2(x)]$ is shown for the sake of clarity. Each data point (colored *triangles*; the *star* and *plus* symbols should help to a better comprehend the mapping) is mapped into a single bucket in each of the N hash tables (*middle*) using the respective hash functions $g_j(x)$ $(1 \leq j \leq N)$. In the last step, each hash table (or histogram) is normalized, and all hash tables are represented as a matrix (*right*).

maximize hash collisions for similar data samples. This means that similar data points have the same hash value $h(x)$ with a high probability [Lan+11]:

$$Pr_{h \in \mathcal{H}}[h(x_a) = h(x_b)] = sim_{\mathcal{H}}(x_a, x_b), \{x_a, x_b\} \in D_i \tag{6.4}$$

To get a similarity-preserving histogram representation, CP randomly samples L independent $\{0,1\}^{d'}$-valued hash functions $h(x)$ from $\mathcal{H}$, defining a concatenated function $g(x) = [h_1(x), h_2(x), ..., h_L(x)]$. In this way, the probability of *false positives* (distant data points are hashed together in the same bucket) decreases exponentially, but the probability of *false negatives* (close data points are hashed apart) increases [Tei+13]. To overcome this issue, multiple functions $g_j \in \mathcal{G}$ $(1 \leq j \leq N)$ are used to build N independent hash tables (*middle* of the Figure 6.2). The number of hash tables ($N = |\mathcal{G}|$) determines the tradeoff between efficiency and accuracy [Lan+11] – this tradeoff will be further analyzed in Section 6.2.3. If only one data point would be considered (i.e., no hash collisions exist in the hash tables) and if a large number of LSH keys would be observed, this single data point could be approximately localized but 'only' in a sector of a hyperplane within the instance space [Pat12]. By applying the hash functions $g_j \in \mathcal{G}$ to the whole dataset D_i, hash collisions (i.e., different data points with the same hash key are mapped together in the same bucket of a hash table) prevent such localization of single data points.

In the second step, the histogram vectors H_{g_j} resulting from each hash table $g_j \in \mathcal{G}$ are normalized (i.e., information about the absolute number of underlying data $|D_i|$ is lost) and then transferred into a matrix as follows:

$$T_g(u_i) = |D_i|^{-1}[H_{g_1}, .., H_{g_N}]^\top \tag{6.5}$$

Here, $\{x_s, s = 1..|D_i|\}$ is the considered data of a user u_i for training the model update C_i^p. Each element of the resulting (normalized) histogram representations H_{g_j} can be regarded as a bucket representing the frequency of the user data

mapped. It is important to note that these mapping functions $g_j(x) \in \mathcal{G}$ are known to all participating users in the system in order to maintain comparability. `{P}Net` uses this matrix representation $T_g(u_i) \in [0,1]^{N \times 2^{a'}}$ (referred to as *signature* in the following) as an irreversible representative [Hoy14] for the underlying training data characteristics of a user u_i.

Anonymous Sharing of Model Updates

To populate the central repository, users or volunteers u_i can share their model updates C_i^p together with the corresponding signatures $T_g(u_i)$ (see Figure 6.1, ❷-a). As shared model updates in particular may negatively affect the user's privacy depending on the classification algorithm used (e.g., through membership inference attacks [Sho+17]), the following two complementary approaches are conceivable: the initial population can be made by a few (to many) paid users or volunteers, also to solve the cold-start problem before the service is released; after the release of the service, voluntary users (e.g., due to offered incentives by the provider) may also share model updates anonymously through an anonymization network, which is used to ensure *sender anonymity* – this prevents the linkability (from communication metadata) between the shared model updates with signatures $\{C_i^p, T_g(u_i)\}$ and the sharing users u_i.

We will now exemplify how CP can be performed by using `Crowds` [RR98], which represents one potential approach for such an anonymization network that fulfills the requirement of sender anonymity. `Crowds` relies on probabilistic/randomized forwarding that hides the communication relationships by using a distributed network of participating service users (the broker network). More precisely, users can communicate with each other *independently* from the AI service provider (e.g., via an underlying peer-to-peer network); they work together as brokers to deliver the message (model update) to its recipient (AI service provider) by probabilistically relaying the message: the message circulates among the users for an unknown and undefined amount of time until a random broker (user) delivers the message to its recipient. We assume that the AI service providers have *no* knowledge of the global message flow, i.e., they do *not* cooperate with Internet service providers. In this way, all participating users U are equally likely to be the creator of a message. Thus, the resulting linking probability is $1/|U|$ [RR98] – the greater the number of participating users, the less likely that the AI service provider is able to link model updates to their creators. Even if the AI service provider would be a forwarding member with multiple user identities (i.e., provider-controlled nodes) $U_c \in U$ in the network, anonymity guarantees can be determined by the certainty of the provider to identify its predecessor as the creator of the message. Achieving a rather conservative level of anonymity, e.g., $p \leq 0.5$ as the *probable innocence*, the number of all participating users $|U|$ must be large enough to satisfy the inequation $|U| \geq \frac{p_f}{p_f - 0.5} \cdot (|U_c| + 1)$ with a forwarding probability $p_f > 0.5$ [RR98].

Other potential approaches that can also ensure sender anonymity are from the group of DCnets (dining-cryptographer networks) such as `ADCnets` [GDM18]

and `Dissent` [CF10] – a comprehensive survey of potential approaches and their characteristics is given by [Shi+18].

6.1.2.2 *Requesting Community-based Model Updates*

Assuming a populated repository $\mathcal{R}$, the mass of *new* service users u_k can then request an appropriate community model C_m from this repository only using their signature $T_g(u_k)$ (see Figure 6.1, ❷-b). The procedure is as follows: a user u_k creates a signature $T_g(u_k)$ (as described in the last section) with only those personal data D_k that are needed for the service (or in other words, which would serve in the LP step for training a personalized model). This signature $T_g(u_k)$ can then be used to request a model update at the central repository by searching for *similar* signatures and associated model updates $\{T_g(u_j), C_j^p\} \in \mathcal{R}$ – as it is expected that such model updates C_j^p from other users u_j will improve the general model for this new user u_k as well.

To determine the similarity between two signatures $T(u_k)$ and $T(u_j)$, a (dis)similarity measure is applied, which compares the histograms per row of the matrices and averages over the results as follows:

$$dissim[T(u_k), T(u_j)] = \frac{1}{N} \sum_{r=1}^{N} dist[T_r(u_k), T_r(u_j)] \tag{6.6}$$

A *dissim*-value of (/near) zero means an identical (/very high) similarity between both signatures. In other words, two users u_k and u_j are 'similar', if they have many (frequency) values in common in the corresponding bins of their histograms. We now exemplify a possible bin-to-bin distance and decided to use the non-linear Chi-Squared (χ^2) distance, as it takes the difference between small bins (in relation to the difference between large bins) more into account than other distances – this leads to a finer distinctions between the users. The χ^2-distance is defined as follows (in simplified notation) [PW10]:

$$dist[t_1, t_2] = \frac{1}{2} \sum_{r} \frac{(t_{1,r} - t_{2,r})^2}{t_{1,r} + t_{2,r}} \tag{6.7}$$

Based on the (dis)similarity measure, `{P}Net` builds a similarity network with edge weights $(w_{k,j})_{j=1..|\mathcal{R}|} = dissim[T_g(u_k), T_g(u_j)]$ around a user's request. The resulting community-based model $C_m(x)$ is composed of (i) the model update C_j^p, which is associated with the signature $T_g(u_j)$ having the lowest edge weighting $w_{k,j}$, and (ii) the general model C_0 [KFF19]; the community-based model is then delivered to the requesting user u_k, who can use it directly.

Despite the unique benefits of CP (e.g., obtaining an initial personalized model without labeling effort), users still have to accept a compromise in terms of their privacy (though smaller than comparable community-based approaches in the literature; see Section 3.4). This is due to the inherent possibility of similarity

comparisons with the shared signature: although {P}Net never shares highly privacy-critical (raw) personal data and the signature of the user is irreversible, meaning raw personal data cannot be reconstructed exactly [Hoy14], the provider can assign the user to a group of 'similar' users (i.e., users with a similar relative frequency distribution of their service-relevant data).

To address this issue, there are two conceivable ways that lead to even better tradeoffs: (1) New users can also send their request to the provider via an anonymization network, as described in the last section and illustrated in Figure 6.1 (❷-b); this prevents linkability (from communication metadata) between the shared signatures and the requesting users. (2) If enough local resources are available, which may be the case for non-mobile users, the central repository (if not too large, which depends on the AI service) can be downloaded in encrypted form; together with the data protection approaches from Chapter 5, this enables to execute the similarity comparisons locally. This second (alternative) path involves a high local resource use, but the user does not have to compromise on privacy. We will discuss the newly achieved tradeoffs again in Section 6.4 after the evaluation of {P}Net, which follows next.

6.2 EXPERIMENTAL SETUP

This section describes the methodology used to conduct experiments with the {P}Net prototype. Specifically, {P}Net is evaluated on two different datasets and compared against currently-practiced, representative concepts.

6.2.1 *Datasets*

The two datasets used are publicly available and well-known in the community. Both will be described in detail below.

The first one is the *WISDM human activity recognition*[38] dataset (abbreviated as D^{act})—comprising 36 users carrying an off-the-shelf smartphone in a lab environment while performing a specific set of six simple activities: {*walking*, *jogging*, *upstairs*, *downstairs*, *sitting*, *standing*}. The sampling rate of the accelerometer was 20 Hz. [MBM20] The collected data is then divided into 10-second segments from which 43 summary features (variants of six basic features) are extracted. Due to the high dimensionality of the feature space and the small number of available data samples, we apply a feature selection method (namely information gain ranking filter [Yil15]) to avoid the curse of dimensionality [CRM09], resulting in 10 features. In total, the resulting dataset contains 5,418 instances, i.e., 150.5 ± 44.7 (labeled) instances per user. [MBM20]

The second one is the *GeoLife GPS trajectory*[39] dataset (abbreviated as D^{trans})—comprising 26 selected users who logged their GPS trajectories and

[38] http://www.cis.fordham.edu/wisdm/dataset.php (retrieved 06/30/2021)
[39] https://www.microsoft.com/download/details.aspx?id=52367 (retrieved 06/30/2021)

labeled enough of them with four selected transportation modes: {*walk*, *bike*, *bus*, *car*} [ZXM10]. A sample was recorded every 1-5 seconds or every 5-10 meters – whichever occurred first. In total, the resulting transformed dataset contains 12 features and 7,511 instances, i.e., 288.9 ± 287.3 (labeled) instances per user. [MBM20]

6.2.2 *Reference Baselines*

A comparison against (representative) reference baselines from Section 3.4.2 is used to classify the performance of `{P}Net` in the comparative Table 3.3. As concluded in Section 3.4.3, *data-modifying* and *data-encrypting* approaches are currently not well suited for personalization, as they suffer from either low effectiveness or limited applicability. `{P}Net` focuses on the more practical and promising *data-minimizing* and *data-confining* approaches – of which the reference baselines are selected below.

According to their re-implementability and source openness, three representative baselines were selected and categorically named as follows: *crowd-sourced general model* (GM), *local-retrained personal model* (PM-LR), and *local-incremental personal model* (PM-LI) – see also Section 3.4.2 for more details. The former is a currently-practiced data-minimizing approach, while the latter two are used in a data-confining setting. It is important to note that all approaches (GM, PM-LR, PM-LI, LP, CP) examined in Section 6.3 use the same features and feature extraction process. However, they differ (i) in the possible underlying AI algorithms—only state-of-the-art algorithms such as in [Ser+18] are used—and (ii) how and where they approach the AI model training and access user data (local vs. remote vs. 'local/remote'-hybrid).

6.2.3 *Experimental Parameters*

This section explains the used experimental parameters—namely the *LSH parameters* and the *partitioning of the dataset*—in the following experiments.

6.2.3.1 *LSH Parameters*

`{P}Net` relies on LSH in the CP step for similarity comparisons. LSH has two key parameters, namely the number of hash functions L per hash table, and the number of independent hash tables N (see Section 6.1.2 and Figure 6.2). Both parameters control the tradeoff between accuracy and efficiency: the larger the value of the parameters, the more often close data points are hashed together (true positives) [Tei+13] – but the more hashes ($L \times N$) must be computed, which decreases the efficiency [Lan+11].

We will now analyze and experimentally determine this tradeoff. Figure 6.3 shows the accuracy as a function of both parameters. Note that the plot on the *left* shows on the x-axis the ratio between the dimension d of the origin data space ($\mathbb{R}^d$) and the dimension d' of the target data space ($\{0,1\}^{d'}$) to

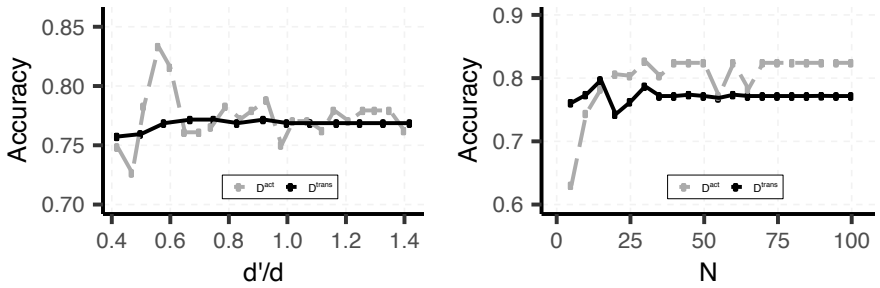


Figure 6.3: Analysis of the tradeoff between accuracy (*y-axis*) and efficiency (*x-axis*) for both LSH parameters L (the number of hash functions per table, see *left* plot) and N (the number of hash tables, see *right* plot). To make statements independent of the data set, the plot on the left shows on the x-axis the ratio between the dimension d of the origin space and the dimension d' of the target space (the value of the latter is identical to L). The higher the value on the x-axis, the more hashes must be computed and the lower the efficiency.

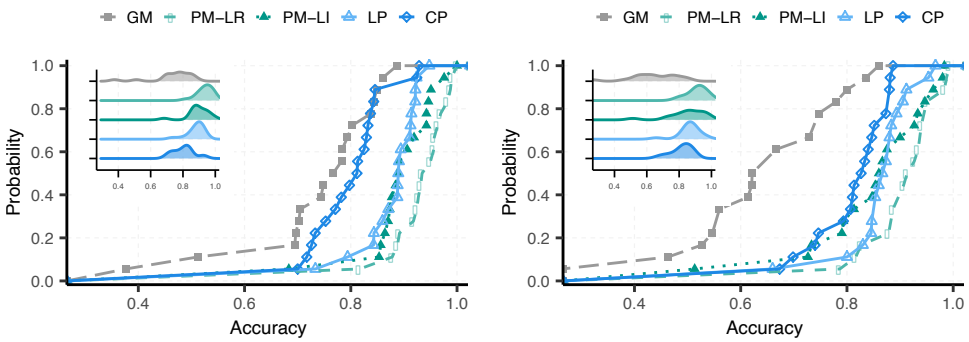
make data set independent statements – the value of the latter is identical to L. The best tradeoff for the respective parameters results from values that are as far as possible to the upper left. In addition, a requirement declared in Section 6.1.2 is that $d'/d < 1$ for the purpose of dimensionality reduction.

Taking into account the above requirement, the values are experimentally determined as follows [Tei+13]: the ratio d'/d is set to 0.6, as the accuracy reaches a maximum for both datasets; this results in $L_{D^{act}} = 6$ and $L_{D^{trans}} = 7$ (see Section 6.2.1 for dataset characterization). For the number of hash tables, the best tradeoff is $N = 30$ for both datasets; larger values would also be possible as the accuracy remains (nearly) constant but would be less efficient. We will use these values to conduct further experiments in Section 6.3.

6.2.3.2 *Dataset Partitioning*

To simulate the entire workflow of `{P}Net` (see Figure 6.1), each dataset is first split into two sets: *crowd-sourced data* (D_c) and *local/personal data* (D_p). The former (D_c) is only used to train the general model (GM), building the initial part of `{P}Net`. The latter (D_p) is used to (i) train model updates to initially populate the repository and (ii) test the community-based model for a new user. Specifically, we split both datasets D^{act} and D^{trans} so that the resulting subset D_p for each dataset contains the same number of users (U_p) for comparability and at least half of the users to have enough users to train the model updates.

As model validation method, a *k-fold cross-validation* is used: the set D_p is randomly partitioned into k equal sized folds; the models are then trained on $k - 1$ folds ($\rightarrow D_l$) and tested on the remaining fold ($\rightarrow D_t$); this method is repeated for all k folds, so that each fold serves once as a test set; the final result is averaged across all iterations. It is important to note that this method is applied per user $u_i \in U_p$ to assess the model performance for each user individually. For instance, to simulate CP, the training sets of all users $D_l(u_j) \mid u_j \in U_p \backslash \{u_t\}$ except that of the current test user $u_t \in U_p$ are used. Due to the small data sizes of the



(a) Dataset D^{act} (b) Dataset D^{trans}

Figure 6.4: Classification accuracy (defined as the ratio of number of correct predictions to the total number of predictions made) as CDF and PDF (*subplot*)

different user data sets $u_i \in U_p$ (see Section 6.2.1) and to ensure that each fold contains sufficient variation so that the underlined distribution is represented, we chose $k = 5$: this is a value that has been empirically shown, particularly for small dataset sizes, to yield test error rate estimates that suffer neither from either excessively high bias nor from very high variance [Jam+13]. The reference baselines introduced in Section 6.2.2 are evaluated analogously.

6.3 RESULTS

This section examines the personalization performance of `{P}Net` in terms of effectiveness and efficiency.

6.3.1 *Effectiveness: Robust Classification*

Figure 6.4 shows the Cumulative Distribution Functions (CDFs) of accuracy per user—better performance is indicated by curves closer to the right. The subplots further show the Probability Density Functions (PDFs). As not all data is normally distributed[40], non-parametric Friedman tests are applied in the following to support the visual impression statistically [Dem06].

According to the results for D^{act}, $\chi^2(4) = 55.9, p < .001$, there exists a statistically significant difference between the techniques considered (see Figure 6.4a). Median (and Interquartile Range (IQR)) perceived effort levels for GM, PM-LR, PM-LI, LP, and CP are 0.775 (0.703 to 0.829), 0.942 (0.920 to 0.968), 0.890 (0.871 to 0.942), 0.890 (0.862 to 0.914), and 0.812 (0.757 to 0.836), respectively.

Post hoc analysis with Wilcoxon signed-rank tests was conducted with a Bonferroni correction applied to avoid type I errors—resulting in a new significance level set at $\alpha = .05/5 = .01$. Figure 6.5 visually summarizes the results of the pairwise comparisons. It is important to note that the performance of two approaches is significantly different, if the corresponding average ranks

[40] Shapiro-Wilk tests: $p_{act} < .05$ for GM, PM-LI, and LP; $p_{trans} < .05$ for PM-LI, LP, and CP

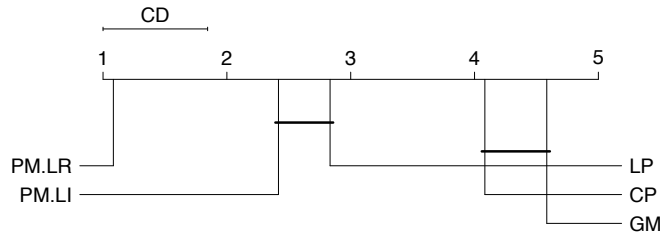


Figure 6.5: Statistical comparison of all approaches for the dataset D^{act} using Wilcoxon signed-rank tests (with Bonferroni correction). Approaches that are *not* significantly different ($p > .01$) or whose corresponding average ranks differ less than the *critical difference* (CD) are connected.

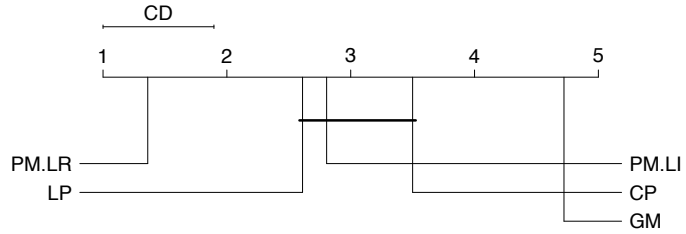


Figure 6.6: Statistical comparison of all approaches for the dataset D^{trans} using Wilcoxon signed-rank tests (with Bonferroni correction). Approaches that are *not* significantly different ($p > .01$) or whose corresponding average ranks differ less than the *critical difference* (CD) are connected.

differ by at least the so-called *critical difference* (CD) [Dem06]. These pairwise comparisons reveal that LP ($\mu = 0.880$, $\sigma = 0.052$) is significantly better ($p < .001$) than GM ($\mu = 0.744$, $\sigma = 0.126$) and consistently achieves high accuracy, demonstrating *effectiveness* of `{P}Net`; it further achieves comparable results ($p = .086$) as PM-LI ($\mu = 0.896$, $\sigma = 0.068$) that incrementally considers only personal data of a user. Not surprisingly, PM-LR ($\mu = 0.937$, $\sigma = 0.047$) performs best across all users, as it is always retrained locally on all available labeled user data; it is significantly better than LP ($p < .001$) and CP ($p < .001$). However, PM-LR involves a high labeling effort for users and a high local resource use. LP is significantly better ($p = .004$) than CP. For this dataset, there exists no significant difference ($p = .069$) between GM and CP ($\mu = 0.804$, $\sigma = 0.064$). However, CP has a higher average accuracy ($\Delta_\mu = 0.060$) and a lower variance ($\Delta_\sigma = 0.062$) compared to GM. Especially, CP ($acc_{min}^{CP} = 0.700$) overcomes the issue that GM can be very bad for particular users ($acc_{min}^{GM} = 0.376$). This means CP is *more robust* than GM among all users. [MBM20]

According to the results for D^{trans}, $\chi^2(4) = 44.3, p < .001$, there also exists a statistically significant difference between the techniques considered (see Figure 6.4b). Median (and IQR) perceived effort levels for GM, PM-LR, PM-LI, LP, and CP are 0.644 (0.560 to 0.747), 0.920 (0.887 to 0.947), 0.863 (0.798 to 0.932), 0.870 (0.848 to 0.891), and 0.830 (0.797 to 0.868), respectively.

Again, post hoc analysis with Wilcoxon signed-rank tests was conducted with a Bonferroni correction applied to avoid type I errors—resulting in a new

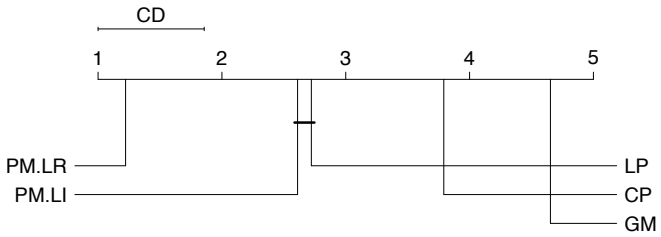


Figure 6.7: Statistical comparison of all approaches *across all datasets* against each other using Wilcoxon signed-rank tests (with Bonferroni correction). Approaches that are *not* significantly different ($p > .01$) or whose corresponding average ranks differ less than the *critical difference* (CD) are connected.

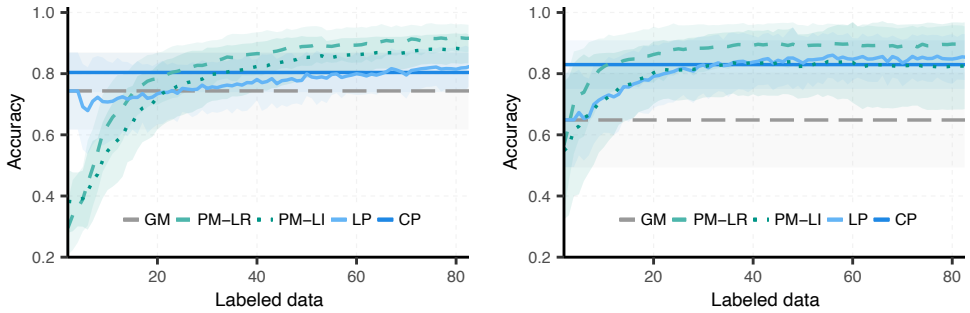
significance level set at $\alpha = .05/5 = .01$. Figure 6.6 visually summarizes the results of the pairwise comparisons. These pairwise comparisons reveal that *all* techniques are significantly better ($p < .001$ for LP, PM-LI, and PM-LR; $p = .004$ for CP) than GM ($\mu = 0.645$, $\sigma = 0.148$). PM-LR ($\mu = 0.912$, $\sigma = 0.058$) again performs best, followed by LP ($\mu = 0.865$, $\sigma = 0.065$), PM-LI ($\mu = 0.850$, $\sigma = 0.115$), and CP ($\mu = 0.816$, $\sigma = 0.064$) – there are significant differences to the last three ($p < .01$), which do not differ significantly from each other. In other words, CP achieves comparable classification results for this dataset as the incremental approaches PM-LI and LP.

All in all, according to the results, $\chi^2(4) = 97.2, p < .001$, there again exists a statistically significant difference between the techniques considered *over all datasets* (D^{act}, D^{trans}). Figure 6.7 visually summarizes the results of the pairwise comparisons. We can see that PM-LR, as usual, performs best, followed by PM-LI and LP – there is no significant difference between the latter two. As intended, CP outperforms GM over both datasets, demonstrating its *effectiveness*. Although LP and all PM baselines perform quite well, they burden the users and their devices in terms of labeling effort and local resource usage, respectively (*low efficiency*) – this becomes clear in the following experiments.

6.3.2 *Efficiency: Low User Burden*

This section analyzes the classification performance achieved by the techniques under different numbers of labeled data available per user, as a means for investigating the effects of different user efforts for labeling. To this end, the performance of the respective models is evaluated on the basis of different training subsets with a size of $\{1..|D_l|\}$. This also simulates real-world situations where more and more new labeled data gradually becomes available.

Figure 6.8 shows the accuracy averaged over all users as a function of available labeled data. We can see that both data protection techniques of `{P}Net` reduce the burden on users caused by the labeling of training data. CP in particular achieves the highest classification performance when *no* or only a few labeled data is available – this is true for both datasets (see Figures 6.8a+6.8b). LP starts from the general model (GM) or from the CP model and further adapts to



(a) Dataset D^{act} (b) Dataset D^{trans}

Figure 6.8: Averaged classification accuracy (defined as the ratio of number of correct predictions to the total number of predictions made, averaged over all users) as a function of the available personal data labeled by the respective user

the user with each additional labeled instance (see Figure 6.8b for the former case). Compared to PM-LI, LP achieves even better results with the same amount of labeled data; it is also more efficient (w.r.t. labeled data and local resources) than PM-LR, as the latter retrains the entire model for each new instance.

To emphasize the advantages of CP in such cases and to statistically confirm the performance gain compared to GM again, a non-parametric Wilcoxon signed-rank test is applied to compare both [Demo6]. According to the results, $Z = 75.0$, $p < 0.001$, there exists a statistically significant difference between the two approaches (CP and GM), which are independent of the number of labeled data. Using Cohen's d, the effect size is -0.742, which indicates a medium ($|d| > 0.5$), almost a large ($|d| > 0.8$), magnitude of the phenomenon [Coh88].

All in all, we can say that CP performs as intended: when *no* or only a few labeled data are initially available, in cases in which personal models have a cold-start problem, CP performs best, significantly outperforming the general model.

6.4 DATA PROTECTION ACHIEVED AND APPLICATION VARIANTS

This section discusses the data protection achieved by the approach and its different application variants for CP. First, `{P}Net` assumes a *general model*, which should be learned before the service is released, as it does *not* require any personal data of the new (target) user. The general model can be learned in different ways, e.g., 'classically' directly from voluntary data or in a more privacy-preserving way through approaches relying on differential privacy [Aba+16] or federated learning with secure aggregation [Bon+17] (see Section 3.4.2).

In the LP building block (❶), `{P}Net` can adapt an underlying shared model to get a *personalized model* for a user. Specifically, LP is executed completely locally, i.e., *no* data leaves the user territory (see data-confining approaches in Section 3.4.2). LP can be used in two ways: once as preparation for the CP building block (see Figure 6.1, ❶→❷-a, p. 109), once as follow-up to the CP building block. In the former way, the underlying model is the general

model; in the latter way, it can also be an 'evolved' model [KLF19] such as the community-based model, which is the same procedure for the target user as described in Section 6.1.1.

In the CP building block, the first stage (❷-a) is to populate the central repository. For this, both (i) model updates learnt in the LP step and (ii) LSH-based histogram representations need to be shared. Regarding (i), machine learning models can generally be subject to so-called membership inference attacks, although only black-box access exists. In this way, it could be determined if a data point was in the training dataset of the model [Sho+17], which may *negatively affect* the user's privacy depending on the classification algorithm used. Regarding (ii), considering only a single LSH hash of a data point, various works showed that this hash is *irreversible* [Hoy14] and does *not leak* information about the original data point [FKM21]. In practice, however, approaches may require to release the choice of hash functions (or hyperplanes) $g_j \in \mathcal{G}$, e.g., to achieve comparability. This could lead to potential linkage or intersection attacks by an attacker armed with auxiliary information [FKM21], revealing the probable d-dimensional subspace in which the original data point could be located. In `{P}Net`, a normalized histogram representation is built based on multiple colliding hashes (see Figure 6.2, p. 112). This procedure results in a loss of information, meaning that the number of data points per d-dimensional subspace can no longer be determined in addition to having data points that cannot be reconstructed exactly (see above). However, the relative frequency distribution over specific d-dimensional subspaces can be derived using the hash functions released, which may *negatively affect* the user's privacy. On the one hand, as intended, the provider uses such a frequency distribution of a user in the second stage of CP (❷-b) to determine suitable model updates from other users with 'similar' frequency distributions; on the other hand, the provider can also use them to obtain probable information about the user or to assign this user to a specific group through similarity comparisons with 'reference' users.

In `{P}Net`, populating the repository can be carried out in two complementary ways: (1) similar to the general model training, the initial population can be made by a few (to many) paid users or volunteers, also to solve the cold-start problem *before* the service is released. For example, parts of the voluntary data can be used for training model updates. (2) To further grow the repository, *after* the release of the service, voluntary users (e.g., due to offered incentives by the provider) may also share model updates *anonymously* through an anonymization network (as illustrated in Figure 6.1, ❷-a, p. 109), which is used to ensure *sender anonymity* – this prevents the linkability (from communication metadata) between the shared model updates with signatures $\{C_i^p, T_g(u_i)\}$ and the sharing users u_i. Exemplary state-of-the-art approaches fulfilling this requirement and being compatible with `{P}Net` are described in Section 6.1.2.1; a comprehensive survey for further approaches is given by [Shi+18]. To further enhance the privacy for these voluntary users, future work can investigate a new setting with federated learning: it can be used to train different community models that *only* consider 'similar' users in each case and thus

become better for those users and in particular similar new users than the general model – the similarity comparisons can be performed as proposed by {P}Net.

The second stage (❷-b) of the CP building block concerns the vast majority of actual users of the system, whose privacy must be specially protected. To address the cold-start problem for new users u_j, they only require their signatures $T_g(u_j)$ for finding an appropriate model update C_i^p from other but 'similar' users u_i. However, as discussed above, sharing such a signature with the provider without additional protection measures may *negatively affect* the user's privacy.

To cope with this issue, {P}Net can work in two conceivable application variants that lead to different tradeoffs between privacy and efficiency (in terms of local resource use and initialization time), which a new user must accept: The *first* variant (abbreviated as *CP1*) is particularly suitable for users for whom *resource efficiency* and a *fast initialization time* of the service are crucial: these users can send their request to the provider via an anonymization network (as described above and illustrated in Figure 6.1, ❷-b, p. 109); this prevents linkability (from communication metadata) between the shared signatures and the requesting users. The *second* variant (abbreviated as *CP2*) is particularly designed for users for whom privacy and data (including metadata) must be *fully protected* (which only works at the expense of resource efficiency): in addition to the general model, the central repository or a diversified part of it (if not too large, which depends on the AI service) can be downloaded in encrypted form. To give the reader an impression for the data sizes, they are exemplified by the datasets used: the general model (based on random forest) has a size in the high single-digit to mid double-digit MB range; it highly depends on the size of the underlying dataset and the algorithm used. A single {signature, model update}-pair has a size of less than 1MB, whereby the signature with its size in the low one- to two-digit kB range (which highly depends on the chosen LSH parameters L and N) only accounts for a fraction. Without optimizations, the resulting repositories are in the low one- to two-digit MB range for the user base of the experiments conducted (see Section 6.2), which is in the same order of magnitude as the general model. For a much larger user base and repository, only a diversified subset should be provided, as a performance gain has already been demonstrated in a relatively small number of 'communities' (see Section 6.3). Together with the data protection approaches proposed in Chapter 5, providers can then perform the similarity comparisons (which only require decrypting and processing the small-size signatures) *locally* within a confidential processing environment – this would neither leak the user's signature nor model updates from the repository that are not relevant to this user.

6.5 CONCLUSION

This chapter presented the {P}Net concept to protect user data at *AI level*, which is particularly designed for AI services relying on supervised learning algorithms. {P}Net combines *data-confining* and *data-minimizing* approaches (namely LP and

Table 6.1: A summary assessment of the contributions made in this chapter based on the requirements established in Section 3.4.1 – a comparison with the state-of-the-art protection approaches *at AI level* is given by Section 3.4.2 and Table 3.3

	Data Protection		Personalization					*Applicability*				
Contribution	Integrity (*data*)	Data Confidentiality (*observed/inferred/metadata*)	Performance (*w.r.t. accuracy*)	Personalization Capabilities	Low Personal Data Involvement	Low Labeling Effort	Low Local Resource Usage	Support of Any Data Type	Support of Multiple AI Algorithms	Low Algorithm-spec. Dependence	Easy deployment and Integration	GM Learning/ Improv. Capabilities
`{P}Net`	●	●/●/◐*	◐-●	●	●	●	●*	●	◐	●	◐	◐

Encoding: **fulfillment of requirements** (see Section 3.4.1): ●–fulfilled, ◐–partially fulfilled, ○–little or not fulfilled; *the fulfillment of these two requirements can be interchanged depending on the used application variant CP1 or CP2.

CP) to exploit the strengths of both groups of approaches, achieving a unique tradeoff between personalization and data protection.

6.5.1 *A Unique Tradeoff at AI Level*

Table 6.1 shows a summary assessment of `{P}Net` compared to other representative baseline approaches based on the requirements established in Section 3.4.1 – supported by the evaluation in this chapter. A comprehensive comparison with the state of the art can be made using Table 3.3 (p. 51), confirming the unique tradeoff achieved by `{P}Net`.

In terms of data protection, AI services in `{P}Net` only have access to user data *locally* (see LP, ❶), and the signatures are also computed *locally* (see CP, ❷-b). Both ensure the integrity and confidentiality of raw user data at AI level; this kind of data never leaves the user territory. In CP, however, a new user (❷-b) has to make a tradeoff more in favor of data protection and in favor of resource efficiency: in the setting of CP1, users require to *anonymously* share metadata, namely their signatures (i.e., the LSH-based frequency distribution of the service-relevant data), with the provider to get a suitable model update and cope with the cold-start problem of personalized services; the linkability (from the communication metadata) between the user and the shared signature (which may reveal the user's belonging to certain 'reference' groups) is prevented – this privacy tradeoff is still better than comparable works that address the cold-start problem, such as other community-based approaches like CSN [Lan+11] (see Section 3.4). In the setting of CP2, all user data – even the signatures – are

analyzed *locally* by the provider; however, this is at the expense of resource efficiency and initialization time. This data protection achieved for new users requires a populated repository, which can be provided by voluntary data or by anonymously-contributed model updates from other (voluntary) users (❷-a).

In terms of personalization, {P}Net copes with the *cold-start problem* of personalization in AI services for new users. More precisely, in case in which no or only a few labeled data is available, in which local training and personalization algorithms suffer from a cold-start problem, the evaluation showed that CP performed best and significantly outperformed the general model. For this, CP does not require any labeled data. If labeled data is available, LP can personalize the general model to users, achieving comparable high performance as personal models. In contrast to the general model, the evaluation also showed that LP and CP perform consistently well across all users.

In terms of applicability, {P}Net is not restricted to a specific data type but to supervised learning algorithms (and classification tasks). CP also works only for the subset of personalization algorithms that can reflect the differences between different users in their model updates – this, in turn, means that a model update of one user will also work well for a 'similar' user, whereby similarity here refers to sensor data similarity. The used patching framework for LP fulfills these requirements for the considered datasets and classification tasks. Patching can significantly improve the performance of the general (or the base) model for a few algorithms including random forest and some kind of deep neural networks in specific settings, as shown in [KF18; KLF19; KFF19]. Theoretical-conceptually, the underlying model can be a black-box classification model, the proof of generalizability is still to be provided.

6.5.2 *Integration & Outlook*

The concept presented in this chapter constitutes a technical protection measure against another use of user data (than initially intended) by authorized but untrusted service providers at AI level – especially when it comes to the personalization of AI services relying on supervised learning algorithms. Under the strict data confinement policy introduced in Chapter 4, {P}Net addresses the inherent efficiency issues (in terms of high labeling effort and local resource use) in local personalization; it copes in particular with the cold-start problem for new users. Due to the split between resource-intensive general model training and lighter local operations that require user data (e.g., local personalization in LP or signature calculations in CP), {P}Net keeps the inherent performance overhead caused by the IP protection mechanisms for locally-running AI services – introduced in Chapter 5 – to a minimum.

All in all, {P}Net completes this part (Part III) of the thesis: all contributions of this part – integrated in PrivAI (see Chapter 4) – enable protection of user data and provider's IP rights at three different levels. At the same time, these

integrated protection mechanisms still satisfy the basic requirement (N1) of emerging AI services—namely the indispensable need for an extensive, constant stream of personal data (for more details on this requirement, see Section 1.1.2).

As AI services and corresponding protection mechanisms based on data decentralization require local resources, the next part (Part IV) of this thesis proposes two appropriate open urban (edge computing) infrastructure concepts and mechanisms to reduce this burden on personal (often resource-limited) devices. In particular, the concepts proposed in Part IV address the second basic requirement (N2) of emerging AI services—namely the need for running them on nearby (IoT/edge) devices to make human environments more responsive and supportive (for more details on this requirement, see Section 1.1.2).

Part IV

DATA DECENTRALIZATION SUPPORTIVE CONCEPTS

7

PROPOSED OPEN INFRASTRUCTURES FOR DISTRIBUTED AI SERVICES

The previous part (Part III) contributed protection mechanisms for personalized AI services based on the data decentralization paradigm. This part (Part IV) adds concepts for *distributed* (personalized) AI services to support their operation—together with the proposed protection mechanisms—*on surrounding* (*IoT/edge*) *devices*, especially when the user is on the move. Distributed AI services can benefit from edge resources in particular to relieve the load on the user's mobile device thanks to short-delay, reliably-connected computing but also to prosper with richer data and new application scenarios (see Figure 5.1, p. 80, as an example) – we use the term *device-bound* AI services (as introduced in Section 2.1.2) to refer to the subset of distributed AI services that additionally require access to sensors and/or actuators on specific (IoT/edge) devices in the physical proximity of the user in order to provide ambient support. On the other hand, AI services can be considered as a driver and as one of the most important application domains for edge computing infrastructures – commonly coined as *edge AI*. [Zho+19]

This chapter first deals with the classic (application–infrastructure) bootstrapping problem that *edge AI* still faces: without unique use cases and corresponding AI services that leverage nearby edge devices, there is no incentive to provide appropriate infrastructures; on the other hand, without a large-enough infrastructure that provides high service coverage for mobile users, there is little incentive for developers to create those new applications.

> *"This state of affairs is similar to that at the dawn of the Internet in the late 1970s to early 1980s. An open ecosystem attracted investment in infrastructure and applications, without any single entity bearing large risk or dominating the market. Over time, this lead to the emergence of a critical mass of Internet infrastructure and applications (such as email) that could uniquely benefit from that infrastructure. By the time the World Wide Web emerged as a "killer app" in the early 1990s, sufficient Internet infrastructure had been deployed for growth to explode."*
>
> Mahadev Satyanarayanan, *The Emergence of Edge Computing*, 2017 [Sat17]

With these words, Mahadev Satyanarayanan describes a way to break this deadlock, namely with an open ecosystem: Chapter 4 presented a possible concept at application level; in this chapter, we will look particularly at the infrastructure level and propose appropriate concepts to achieve the critical mass there.

Reaching mobile users everywhere to enable distributed AI services to benefit from edge resources requires full (spatial) coverage, i.e., a *large-scale deployment* of edge resources is required. To achieve low latency and enable ambient support, physical proximity to the user (typically in the one to two-digit meter range) plays a crucial role (see Chapter 5), i.e., the edge resources must also be

densely deployed. Both requirements result in a correspondingly high number of (IoT/edge) devices that would have to be deployed – this leads to the key challenge of balancing the expected benefits of deployed edge resources with economic aspects. The conditions for this are particularly good in *urban environments* due to the high density of both potential users and existing infrastructures that can be economically exploited. After reaching the critical mass there, expansion to rural areas (where a good cost-benefit tradeoff is more difficult to achieve) can follow, possibly with complementary (wireless) technologies, e.g., to address the greater distances – for example, the expansion of wireless communication technologies (e.g., from 4G to 6G) is done in a similar manner [Hon+21].

Exploiting existing infrastructures can reduce the hardware *deployment or upgrading costs*, especially if the infrastructure is already *electrically operated*: a connection to the power grid, for example, can ensure constant operation; in the event of a power blackout, energy harvesting technologies or batteries can also allow for emergency operation. The latter operation becomes particularly relevant when a dependency on distributed AI services gradually evolves and such an edge computing infrastructure becomes one of the critical infrastructures. Besides the number of (IoT/edge) devices deployed, their *hardware capabilities* also play an important role for the following two aspects: (1) certain prerequisites must be fulfilled for the proposed protection mechanisms of the user data and the provider's intellectual property (see Section 5.2.1), which is often neglected in edge computing [Meu21a]; (2) a good cost-benefit tradeoff requires the weighing of these capabilities per device. For example, each device in the infrastructure should be at least equipped with an appropriate computing platform (at best with TEE support) and network capabilities to enable distributed (yet confidential) AI services; additional hardware such as sensors and actuators are then required for device-bound AI service in particular. However, the hardware should be placed in the environment in a way that a suitable tradeoff between performance and economic aspects can be achieved, e.g., by strategically placing more powerful though more expensive hardware only at a few specific nodes in the network. Last but not least, it is also necessary that the (IoT/edge) devices are not only *available* in the proximity of the users (e.g., to offload resource-intensive AI services, see distributed AI services) but also 'spatially' *reachable* for all users (e.g., to receive ambient support and interact with these devices, see device-bound AI service). The requirements discussed in the last two paragraphs will be further elaborated in Section 7.1, and the related works will be first reviewed based on these established requirements.

In the face of this (cost-benefit) challenge, this chapter examines existing urban infrastructures that can potentially be utilized as an open platform for distributed (and also specifically device-bound) AI services while keeping the necessary deployment or upgrading investments low. Specifically, the contribution of this chapter is twofold. First, this chapter proposes an infrastructure concept based on street lamps (see Section 7.2), referred to as *Street Lamps as a Platform* (`SLaaP`): an initial case study shows its potential; proposed upgrading opportunities show its possible (cost-effective) realization; the identification of newly-enabled

AI services that can benefit from this infrastructure shows the value added for end-users; a discussion from the stakeholder perspective highlights its benefits and the remaining challenges. Second, in an analogous manner, this chapter proposes another infrastructure concept based on wireless home routers (see Section 7.3), referred to as *Routers as a Platform* (RaaP). This infrastructure can be made available at low cost via a sharing concept, complementing the *SLaaP* concept. It is important to note that the two proposed infrastructure concepts are applicable not only to distributed or device-bound AI services covered by this thesis but also to other application domains of edge computing.

Contribution Statement: This chapter is based on the following (4) publications:

- Max Mühlhäuser[a], Christian Meurisch[a], Michael Stein, Jörg Daubert, Julius von Willich, Jan Riemann, and Lin Wang. "Street Lamps as a Platform." In: *Communications of the ACM* 63.6 (2020). **JCR-IF (2019): 6.988**, pp. 56–64. DOI: 10.1145/3376900.

- Christian Meurisch, Alexander Seeliger, Benedikt Schmidt, Immanuel Schweizer, Fabian Kaup, and Max Mühlhäuser. "Upgrading Wireless Home Routers for Enabling Large-scale Deployment of Cloudlets." In: *Proceedings of the 7th International Conference on Mobile Computing, Applications, and Services*. MobiCASE'15. (**Best Paper Candidate**). Springer, Nov. 2015, pp. 12–29. DOI: 10.1007/978-3-319-29003-4_2

- Christian Meurisch, The An Binh Nguyen, Stefan Wullkotte, Stefan Niemczyk, Florian Kohnhäuser, and Max Mühlhäuser. "NICER911: Ad-hoc Communication and Emergency Services Using Networking Smartphones and Wireless Home Routers." In: *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. MobiHoc'17. (**Best Poster Award**). ACM, July 2017, 33:1–33:2. DOI: 10.1145/3084041.3084075

- Christian Meurisch, Julien Gedeon, Artur Gogel, The An Binh Nguyen, Fabian Kaup, Florian Kohnhäuser, Lars Baumgärtner, Milan Schmittner, and Max Mühlhäuser. "Temporal Coverage Analysis of Router-based Cloudlets Using Human Mobility Patterns." In: *Proceedings of the 2017 IEEE Global Communications Conference*. GLOBECOM'17. IEEE, Dec. 2017, pp. 1–6. DOI: 10.1109/GLOCOM.2017.8255035

I (*Christian Meurisch*) led the process of idea generation, conceptual design, implementation, evaluation, and writing. *Alexander Seeliger* and the students *Stefan Wullkotte* and *Artur Gogel* implemented the proof-of-concept prototypes and conducted parts of the evaluation. *Michael Stein*, *Jörg Daubert*, *Julius Von Willich*, *Jan Riemann*, *The An Binh Nguyen*, *Stefan Niemczyk*, *Florian Kohnhäuser*, *Lars Baumgärtner*, *Julien Gedeon*, *Fabian Kaup*, *Benedikt Schmidt*, *Immanuel Schweizer*, *Lin Wang*, and *Max Mühlhäuser* provided helpful critique and comments on the conceptual design; the latter (*Max Mühlhäuser*) has also contributed to the writing process and the generation of new ideas for the CACM article (co-first authorship).

[a]Co-first authors: these authors have contributed equally to this work.

7.1 RELATED WORK

This section first discusses the related work on infrastructure concepts in the context of data decentralization and *distributed* AI services 'at the edge'.

7.1.1 *Specific Requirements*

This section presents the set of identified requirements that are specific to open infrastructure concepts (see the motivation of this chapter). The requirements are divided into two groups, namely *G: global characteristics* and *L: local characteristics*. For the purpose of overview, we also indicate properties that relate to *O: ownership*, as this has a decisive influence on the economic aspects. The group concerning the global characteristics of the infrastructure, which have a direct impact on the service coverage, contains the following essential (mostly spatial) requirements:

- (*G.1*) *Large-scale Deployment:* Mobile users require edge resources not only at a specific location but along their entire path. To support such mobile scenarios, the deployment of (IoT/edge) devices should *cover* a large area.
- (*G.2*) *Dense Deployment:* Users require edge resources in their *physical proximity* in order to benefit from low latency and ambient support. Therefore, the (IoT/edge) devices should be densely deployed to ensure that at least one is always close to a user.
- (*G.3*) *Non-isolated Infrastructure* (*availability/spatial reachability*)*:* All (IoT/edge) devices of the infrastructure should be *available* (in terms of access rights) for all users to benefit from these edge resources (see distributed AI services). For device-bound AI services in particular, the (IoT/edge) devices should be *'spatially' reachable* for all users to benefit from ambient support (e.g., by interacting with them from the HCI perspective).

The individual devices of the infrastructure must be suitable to enable a platform for distributed (and device-bound) AI services at all; the group concerning these essential local characteristics of the infrastructure, which have a direct impact on the performance and economic aspects, contains the following requirements:

- (*L.1*) *Electrical Operation* (*power grid/battery*)*:* To deploy (IoT/edge) devices or augment existing 'things' with hardware or IoT components (see next requirement), they must be connected to the *power grid* or at least run on *battery*. The former enables continuous operation while the latter may enable mobility and make devices more resistant to power failures (blackouts) [Müh+20]. Therefore, both power supplies should ideally be available – energy harvesting technologies can complement them.
- (*L.2*) *Required HW/IoT Capabilities:* To run device-bound AI services, the devices require certain capabilities—including the following (partly essential) components [AIM10]: a *computing platform*/CPU (constituting the indispensable core), a *graphic card*/GPU (to accelerate AI algorithms), *storage* (to cache user data and provider's code/AI algorithms – see Section 5.2),

networking capabilities (to interact with other edge and user devices), accessible *sensors* (to capture the user's context), and *actuators* (to provide direct ambient support). The latter two allow user interaction as well.

– (*L.3*) *Low Deployment/Upgrading Costs:* Stakeholders must continually balance economic viability with technical and application necessities. One possible adjusting screw is to keep the acquisition (and ongoing) costs for the infrastructure low.

The infrastructure or, more precisely, the individual devices of the infrastructure are owned by one or more stakeholders whose interests can be decisive (e.g., for the establishment of an open ecosystem). As mentioned above, this group regarding the ownership only consists of the following properties referred to in the review of the related works below:

– (*O.1*) *Owner type:* Different types of owners pursue different interests, which bring certain advantages and disadvantages to such infrastructure. For instance, companies work in a customer-oriented manner and want to contribute to the success and growth of the 'product', such as in this case an edge infrastructure; on the other hand, companies pursue *commercial* goals that may, for example, infringe user privacy depending on the business model. Non-commercial *private* owners or households tend to pursue only their personal interests, e.g., sharing own resources only in return for the use of resources from others. A sharing economy can keep the costs of such an infrastructure low or distribute them among the participants. *Public* owners (e.g., the city) have a duty to the citizens; such an ownership is ideal for assuring a 'true' (open) infrastructure and establishing regulations, but this is always subject to financial viability; past experience has also shown that such public projects tend to be slow and less customer-oriented. *Mixed* ownerships are also conceivable. For example, a public owner could auction off or lease existing infrastructures with the requirement of open interfaces or other regulations; companies can play to their strengths in terms of speed and growth ambition.

– (*O.2*) *Number of Owners:* Many owners can share the costs but have to compromise on strategic decisions. On the other hand, few owners have a high cost burden but are more independent in their decisions.

7.1.2 *Infrastructure Concepts*

There is a large body of related works on infrastructure approaches that support offloading resource-intensive tasks (or in this context distributed and in particular device-bound AI services). In particular, the infrastructure approaches related to edge computing mostly suggest to exploit existing infrastructures to address the bootstrapping problem. We group these approaches, which have the same or very similar fulfillment of the requirements. We will now discuss these infrastructure concept groups with reference to representative examples.

"The *cloud* is highly elastic and can easily scale resources out", offering virtually-unlimited resources with regard to (*L.2*) computing power and storage capacity [Yee17; VB18]. These resources can be (*G.3*) accessed by all mobile users (*G.1*) almost everywhere over the Internet. However, the cloud or its geographically distributed data centers are neither (*G.2*) densely deployed nor close to the edge, meaning that higher latencies and varying bandwidths are to be expected; the cloud is therefore also not (*G.3*) spatially accessible for users to receive ambient support, especially as it does not include (*L.2*) sensors and actuators. Infrastructure approaches based on cloud computing alone are inherently unsuitable for distributed AI services (when low latencies and high bandwidth are required) and specifically for device-bound AI services.

Privately owned infrastructures constitute, for example, *home computing* (e.g., PCs, NAS, home servers [Cun+09]) and *smart home devices* (e.g., IoT devices like smart speakers [LS16]). Both infrastructures are (*G.2*) densely deployed and (*G.1*) deployed at a large scale, achieving a high coverage in urban environments. Home computing infrastructures typically already have (*L.2*) compute resources, graphics card access, sufficient storage, and the necessary networking capabilities; these characteristics make them highly suitable for distributed AI services. Smart home devices (e.g., smart speakers), on the other hand, are typically (*L.2*) less performant but are equipped with sensors and actuators [Sar17; Cai+19], which makes them suitable for device-bound AI services. However, both infrastructures are (*G.3*) isolated due to the deployment in the homes of device owners, often available only to those owners and also spatially reachable only by them.

Cell towers operated by (*O.1+2*) a few companies have an inherently (*G.1*) high coverage in urban environments, and they are (*G.3*) available for all users. Cell towers are inherently equipped with the necessary networking capabilities (e.g., 3G/4G); an upgrade with (*L.2*) appropriate edge resources for many simultaneous users is (*L.3*) cost-intensive. [Ged+18a] This infrastructure can be exploited as a possible edge computing infrastructure (see mobile edge computing [Abb+17; MB17]) and for distributed AI services. For lower latencies and higher bandwidths, other communication technologies (e.g., mmWave [Hon+21]) with higher frequencies and thus shorter ranges must be used – however, cell towers are (*G.2*) not densely deployed for such ranges and (*G.3*) not in the close proximity to the user. Additionally, cell towers cannot be equipped with (*L.2*) sensors and actuators due to their location at height; therefore, this infrastructure is not suitable for device-bound AI services.

Wireless Metropolitan Area Networks (WMANs) consisting of many wireless (typically WiFi) access points (APs) provide Internet coverage for mobile users in urban areas; they are often owned and operated by (*O.1+2*) local authorities as public infrastructures [Com02]. WMANs are (*G.2*) densely-deployed but often (*G.1*) only in a limited area (e.g., in inner cities or pedestrian zones) – similar to company-operated (wireless) LANs in shopping malls, airports or train stations. WMANs are (*G.3*) available and spatially reachable for all users. [JCL15; Xu+15;

Jia+16] However, using them for distributed or device-bound AI services involves (*L.3*) high upgrading costs, as (*L.2*) computing and storage resources as well as sensors and actuators need to be deployed.

Mobile Ad-hoc Networks (MANETs) are continuously self-organizing, infrastructure-less networks of mobile devices (e.g., smartphones, cars) [HV19]. For instance, Vehicular Ad-hoc Networks (VANETs)—a subfield of MANETs—rely on vehicles and roadside devices for inter-car communication [AF18]. Such networks can achieve (*G.1*) a high but not constant coverage of urban areas. The (*L.3*) upgrading costs to exploit these devices as edge resources are low; however, it is (*O.2*) difficult to encourage users to participate (e.g., with their personal, battery-powered devices) and (*G.3*) make their resources available for other users, which considered individually are (*L.2*) quite limited. In the event of a blackout (*L.1*), however, MANETs could still provide edge resources in addition to emergency communication.

Edge computing infrastructures located in (*coffee*) *shops* or (*public*) *buildings*, as the first concept of cloudlets proposed [Sat+09], can be (*G.2*) densely-deployed in a limited areas (e.g., pedestrian zone) in urban environments; it is (*G.3*) available and spatially reachable for all users. However, (*G.1*) a large-scale deployment of (*L.2*) appropriate edge resources is associated with (*L.3*) considerable costs.

7.1.3 *Implications for this Thesis*

Table 7.1 summarizes relevant infrastructure concepts, showing an overall comparison of them based on the requirements identified in Section 7.1.1. We can see that there is *no* infrastructure concept that completely fulfills all requirements for a (city-wide, performant, cost-efficient) operation of distributed and in particular device-bound AI services. Specifically, existing cloud infrastructures achieve high coverage and are characterized by high elasticity; they are well-suited for distributed (resource-intensive) AI services in cases where latency and the missing physical proximity to users (e.g., to benefit from sensors or actuators) are not an issue. In all other cases, infrastructures close to the edge are required, as outlined below.

Cell towers usually achieve lower latencies than cloud infrastructures and also have a high coverage in urban environments [Ged+18a]; however, their upgrade with appropriate edge resources for potential users within their range is cost-intensive. Private infrastructures (e.g., based on home devices), on the other hand, are already densely deployed with appropriate hardware, achieving low latency and high coverage in urban environments. However, despite their proximity to users, the devices of such infrastructures are isolated, typically only available to their owners. In the light of this deficit, this thesis contributes `RaaP`, an infrastructure concept based on *shared* privately-owned home routers to make these home resources available to other mobile users at low (upgrading) cost.

Table 7.1: Summary of infrastructure concepts for edge computing discussed in the context of distributed AI services

	Global Characteristics			*Local Characteristics*			*Ownership*	
Infrastructures	(G.1) Large-scale Deployment	(G.2) Dense Deployment	(G.3) Non-isolated Infrastructure (*availability*/'*spatial*' *reachability*)	(L.1) Electrical Operation (*power grid*/*battery*)	(L.2) Required HW/IoT Capabilities (*Comp.*/*Graphic Card*/*Storage*/ *Networking*/*Sensors*/*Actuators*)	(L.3) Low Deployment/Upgrading Costs	(O.1) Owner type	(O.2) Number of Owners
Cloud	●/○	●/○	●/○	●/◐	●/●/●/●/○/○	●	co	1+
Home Computing	●/○	●/○	○/○	●/◐	●/◐/●/●/○/○	●	pr	1,000+
Smart Home (IoT)	●/○	●/○	○/○	●/◐	●/○/◐/●/●/●	●	pr	1,000+
Cell towers	●/○	○/○	●/○	●/◐	●/●/●/●/○/○	○	co	1+
WMAN	◐/○	◐/○	●/●	●/●	◐/○/◐/●/◐/◐	◐	co/pu	1+
MANET	◐/○	○/○	◐/○	○/●	◐/○/○/●/◐/○	●	*all*	1000+
Shops/(*public*) *buildings*	○/○	◐/○	●/◐	●/◐	●/●/●/●/◐/◐	○	co/pu	10+

Encoding: **fulfillment of requirements** (see Section 7.1.1): ●–fulfilled, ◐–partially fulfilled, ○–little or not fulfilled; **color coding**: ■–already-existing, ■–potentially-possible and -justifiable (but associated with corresponding costs/efforts), ○–missing for the provision of device-bound AI services; **owner type**: *pr*–private (i.e., non-commercial owners), *co*–commercial (i.e., companies), *pu*–public (e.g., cities)

MANETs can also be used this way as an edge computing infrastructure at low cost. However, the devices considered individually are also resource-limited and do not provide a constant and stable infrastructure. To establish a *potentially* '*true*' infrastructure, it should be operated by local authorities or companies. This purpose can be served, for example, by upgrading WMANs or deploying edge resources in (coffee) shops or (public) buildings – both infrastructures are spatially reachable by users and already densely deployed but only in a limited urban area. Therefore, it is difficult to achieve high coverage with these infrastructures at reasonable costs. In the light of this deficit, this thesis contributes `SLaaP`, an infrastructure concept based on publicly-owned street lamps to achieve a high coverage and dense deployment of edge resources (in close proximity to users); we also point out a once-in-history opportunity to realize it at reasonable costs.

In the following, we will present two complementary infrastructure concepts—namely `SLaaP` (see Section 7.2) and `RaaP` (see Section 7.3)—for edge computing

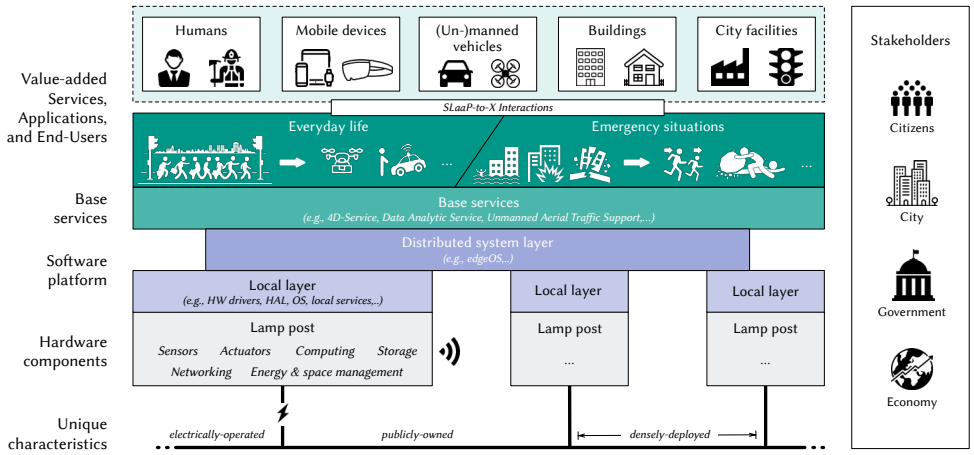


Figure 7.1: An overview of the proposed city-wide infrastructure based on augmented street lamps, its potential end-users/'things' and the stakeholders

and specifically, in the context of this thesis, for distributed AI services in order to address the inherent bootstrapping problem (see the motivation of this chapter).

7.2 PUBLICLY-OWNED: STREET LAMPS AS A PLATFORM

This section presents an infrastructure concept for realizing a cost-efficient, city-wide platform for running distributed AI services, which is based on publicly-owned street lamps – we refer to this concept as *Street Lamps as a Platform* (SLaaP). Specifically, SLaaP fills the gap of a potential 'true' public (edge computing) infrastructure. In the context of the thesis, SLaaP can run distributed AI services and even enable device-bound AI services to provide ambient support to users.

Historical Background

Recent years have seen the emergence of smart street lamps, with very different meanings of 'smart'—sometimes related to the original purpose as with usage-dependent lighting but mostly alluding to add-on capabilities like urban sensing, monitoring, digital signage, WiFi access, or e-vehicle charging[41][42][43]. Research about their use in settings for edge computing [Meu+17b; Jia+18; Ged+18a; Ged+18b] (*authorship* in the former; *co-authorship* in the latter two) or car-to-X support (e.g., hazard detection or autonomous driving) [CM16] hints at their great potential as computing resources. However, to the best of our knowledge, the author of this thesis and co-authors proposed the first comprehensive infrastructure concept based on street lamps, highlighting their innovation hotspots (including newly-enabled distributed and device-bound AI services) for smart cities.

[41] https://smight.com (retrieved 06/30/2021)
[42] https://www.schreder.com (retrieved 06/30/2021)
[43] https://www.stengg.com (retrieved 06/30/2021)

Structure

Figure 7.1 presents an overview of this new infrastructure and the remainder of this section, which is structured (*from the bottom to top*) as follows: after presenting the identified unique characteristics of street lamps (see Section 7.2.1) and a brief case study (Section 7.2.2), this section details cost-effective upgrade options to `SLaaP` (Section 7.2.3). Next, the section presents newly-enabled AI services that will be shown to benefit from such an infrastructure (Section 7.2.4). A discussion from the stakeholder perspective that highlights the benefits and the remaining challenges closes this section (Section 7.2.5).

7.2.1 *Uniquely-qualifying Characteristics*

Street lamps are a basic and important facility of cities, illuminating roads and sidewalks in order to increase the safety of road users and pedestrians' sense of security. This leads to the following characteristics that make them highly attractive from an Information and Communications Technology (ICT) perspective and for distributed AI services or smart city concepts in general.

City-wide and densely deployed: Due to their *dense deployment* along roads and sidewalks, street lamps are already ubiquitous in our everyday urban life. In view of the use as digital infrastructure, this dense deployment makes them *accessible* anywhere in the city and provides high *scalability* due to tens and hundreds of thousands of instances per city – the case study conducted in Section 7.2.2 will quantify and analyze these aspects in more details.

Electrically-operated: In most cities, street lamps are connected to subterranean power grids, which are usually separated from the main power grid. Most of them are still only powered from dusk to dawn since the lamps lack operable switches (power-on means light-on). Nevertheless, the existing power lines (and the avoidance of a cost-intensive new deployment) definitely predestine street lamps as a digital infrastructure, augmented with IoT components [AIM10] – Section 7.2.3 will discuss appropriate upgrading opportunities in more details.

Publicly-owned: Public ownership is ideal for assuring a *true* infrastructure in the strict sense like road, water, energy or telecommunication infrastructures, which constitute sovereign duties like assuring no access (threat protection) or use-case discrimination, professional maintenance, emergency operation, etc. When regulations are established and enforced, privatization is possible, but the inverse i.e. turning private goods into a veritable infrastructure is socially unacceptable. This rules out the consideration of electrically operated and densely deployed devices under private ownership like wireless routers, which could, in principle, be qualified as ICT infrastructure – with `RaaP`, we will propose such a privately-owned infrastructure in Section 7.3.

Cost-efficiently realizable with the 'LED dividend': The widespread lack of such considerations in cities is even more dramatic since a once-in-history opportunity

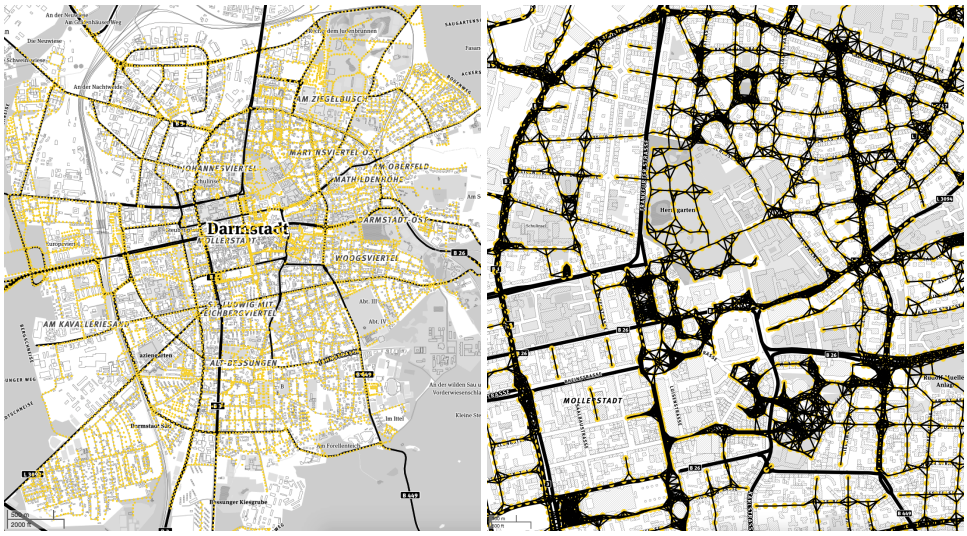


Figure 7.2: Geographic density of public street lamps in urban areas (*left*), and resulting wireless mesh networks (d = 50m) of interconnected inner-city street lamps (*right*), presented by the example of Darmstadt (Germany)

opens up with the changeover to energy efficient LED lighting, expected to save large cities millions in terms of energy cost, called 'LED dividend' below. Given their notoriously-tight budgets, cities urgently need to dedicate these savings if they want to 'own' and control an infrastructure, which, once built, can foster innovation and assure royalties and new business as sources of city and citizen prosperity – Section 7.2.5 will discuss these aspects from the perspective of the different stakeholders in more details.

The four characteristics discussed above uniquely qualify street lamps as *the* scalable foundation for a novel urban digital infrastructure: besides the enrichment with novel (distributed) AI services – as described below – it has the potential to even bootstrap smart cities in a large context.

7.2.2 *Case Study*

To substantiate the currently-unique potential of street lamps as decentralized (urban) infrastructure for edge computing and distributed AI services, an initial case study was conducted in Darmstadt (Germany)[44].

7.2.2.1 *Street Lamps Dataset*

A dataset obtained from an official source of the city government forms the basis of this study, containing all 14,331 publicly-owned street lamps in Darmstadt with a detailed specification (e.g., type, location). More precisely, the data comes directly from the database of the street lamp operator and is therefore carefully maintained; it was provided to us upon request for research purposes only.

[44]Darmstadt has won the prestigious national competition "Digital City" in 2017, aiming to become the digital model city for Germany and even for Europe: https://digitalstadt-darmstadt.de

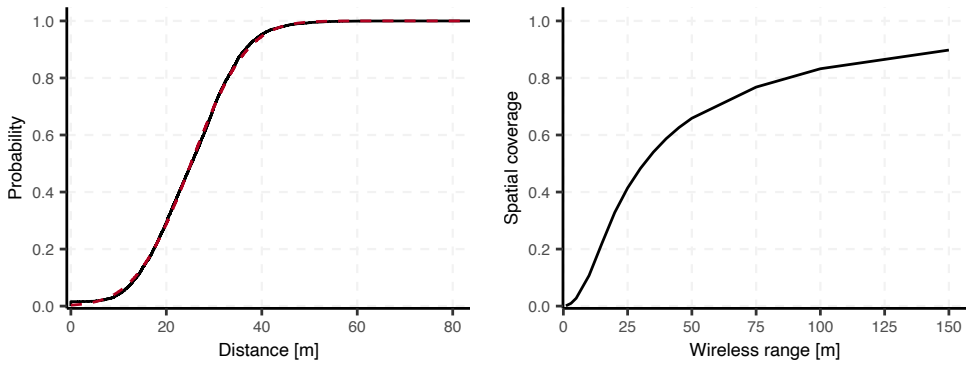


Figure 7.3: Case study in the city of Darmstadt (Germany): distance distribution of public street lamps to their nearest neighbor (*left*), and spatial coverage of street lamps as a function of wireless ranges (*right*)

7.2.2.2 *Understanding the Spatial Coverage of Street-Lamp-based Infrastructures*

Based on this dataset, this section examines the *spatial coverage* that can be achieved with a street-lamp-based infrastructure. Figure 7.2 (*left*) shows the real-world geographic distribution of these street lamps (*yellow dots*) in Darmstadt. We see that the distribution of public street lamps is particularly dense along streets and sidewalks in public or populated areas (e.g., downtown, parks, residential areas); sparse in industrial or privately-owned areas, where property owners are responsible for the lighting. This distribution reflects a representative street lamp deployment pattern in urban environments; cities only vary in the regulated distance between two neighboring street lamps [Set+14; Kap12; HLH15]. In the example of Darmstadt, the distance is about $25.2 \pm 9.2\,m$ (see Figure 7.3, *left*); the underlying values are normally distributed – this is also visually supported (*red dashed line*). These short distances between neighboring street lamps show their dense deployment; it also allows interconnecting them via high-bandwidth wired (e.g., optical fiber, Powerline) or wireless technologies (e.g., WiFi, mmWave) to mesh networks across the city (see Figure 7.2, *right*) – this can additionally enable both the realization of distributed AI services accompanying the mobile user and the workload distribution among each other.

A street lamp of this city-wide mesh network can further act as a nearby access point of distributed AI services (*accessibility*). For instance, assuming a realistic outdoor wireless range of 50m and if only 30% (70%) of all 5,608 street lamps in the inner city ($14.57 km^2$) are upgraded, i.e., 1,682 (3,926) street lamps or $115.4/km^2$ ($269.5/km^2$), nearly half (two thirds) of the inner city can already be covered spatially (see Figure 7.3, *right*), where (mobile) end-users can access services with low latency and high bandwidth – a more detailed analysis is given in [Ged+18a] (*co-authored* by the author of this thesis).

7.2.2.3 *Key Findings and Implications*

This initial case study gives a first impression of the potential of public street lamps as a decentralized and stationary infrastructure for distributed AI ser-

vices or more generally urban services. In the following, the key findings are summarized:

- Street lamps are *densely-deployed* (approx. every 25*m* in Darmstadt) and distributed at *large scale* in the city, especially in public and residential areas.
- With a relatively small fraction (e.g., 30%) of the street lamps, a moderate but comparatively-high spatial coverage (> 50% in Darmstadt, assuming a 50*m* wireless range or user interaction range) can already be achieved.
- Even if all street lamps are taken into account, and a realistic action range is assumed (e.g., 50*m* as above), a complete spatial coverage cannot be achieved but a high and unprecedented one in this context (up to 70% in Darmstadt). Complementary infrastructures are required to (i) approach a complete spatial coverage and (ii) cover more privately-owned areas (e.g., backyards, residential complexes, industrial areas). Regarding (i), if most of the infrastructure is already in place, the (cost-efficient) option will arise of only having to deploy missing devices in the few areas not covered. Regarding (ii), we will propose an infrastructure concept (namely `RaaP`) in Section 7.3, which can provide edge resources in these areas.
- Due to their dense deployment and their physical proximity to the users, street lamps can be equipped with wireless technologies for short distances (e.g., WiFi, mmWave), allowing low latencies and high bandwidths during offloading from mobile devices (see *distributed AI services*).
- The already dense deployment of street lamps along streets and sidewalks predestines them as infrastructure that can effectively provide ambient support to mobile users (see *device-bound AI services*).

7.2.2.4 *Study Limitations*

While this case study provides some interesting findings, it has the following limitations. *Firstly*, the dataset is complete for publicly-owned street lamps (as it comes from an official source), but it does not include street lamps on private property (e.g., parts of the university campus, industrial areas, backyards, or residential complexes). *Secondly*, this initial case study highlights the unique characteristics of street lamps and focuses on their spatial coverage only – a more detailed analysis of the latter is given in [Ged+18a; Ged+18b] (*co-authored* by the author of this thesis).

7.2.3 *Upgrading Opportunity*

In the last two sections, we have seen that street lamps are predestinated as a platform for distributed and even device-bound AI services due to their inherent characteristics. This section describes (cost-effective) upgrade options to exploit them for this purpose – we refer to this concept as `SLaaP` below.

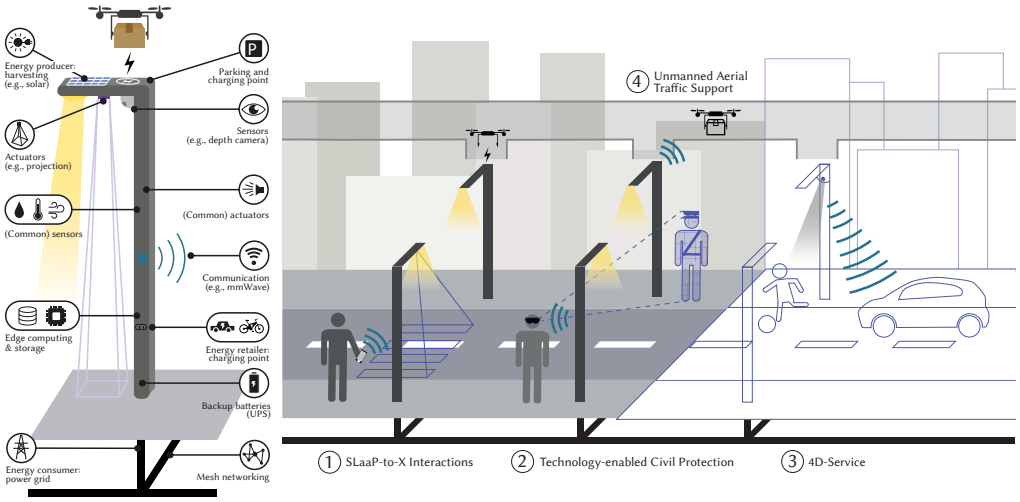


Figure 7.4: A sample view of augmented street lamps (*left*), which act as a citywide infrastructure in smart cities for its new kinds of applications and end-users/smart 'things' (*right*).

As mentioned at the beginning of the chapter, over time, a dependence of people on such AI services, or urban services in general, may gradually evolve to the point where these services become indispensable to people's lives. At this point, such (edge computing) infrastructures as discussed here can become one of the critical ones. Therefore, these infrastructures should be designed and constructed in a resilient manner in order to remain operational in emergency or crisis situations. We will also discuss this requirement of a resilient infrastructure in detail below at the respective crucial components of SLaaP.

7.2.3.1 *Hardware Components*

Conventional lamp posts first need to be upgraded or replaced by augmented street lamps (termed *ASL* – see Figure 7.4, *left*), pursuing a modular and exchangeable design that is appropriate to the desired services. It is important to understand that not all of the following elaborated (exemplary) components must be part of each lamp post but can be strategically distributed across the city to well-selected lamp posts, depending on the service demands and budget.

Computing and Storage

First, distributed AI services and the protection mechanisms proposed in this thesis require computational resources on the edge devices to run there at all. Specifically, a computing platform forms the indispensable core of any ASL. Storage capacities complement this computing platform, e.g., to cache encrypted AI models or system states – but also to enable novel AI services (more in Section 7.3.4).

In concrete terms, this thesis proposes to start upgrading street lamps with single-board computers (SBC)—having a fair tradeoff between good performance, low energy consumption, compact size, and low prize [Kau+18] (*co-authored by the*

author of this thesis). For instance, low-energy ARM-based Raspberry PI ($40), Odroid ($60), or Nvidia Jetson Nano ($109) would be performant enough for a minimalist setup to basically run lightweight AI services. Coping with more resource-intensive AI services, more performant but still economic hardware like 64-bit x86 quad-core Udoo Ultra ($300) or 64-bit six-core Nvidia Jetson Xavier NX with GPU support ($429) is required for real-time capabilities – to still have a good tradeoff between performance and economic aspects, such hardware can be strategically placed in a lower density in the infrastructure. It is further important that the processors used support trusted computing features (e.g., ARM TrustZone, Intel SGX) to allow the protection of user data and providers' intellectual property on (untrusted) devices – see the proposed concepts in Chapter 5.

In the large smart city context, integrating computing power and storage capacities into street lamps is a completely undervalued aspect: street lamps with their unique characteristics profile (see Section 7.3.1) can actually enable an economic large-scale deployment of edge resources (despite their range restrictions), making the breakthrough of practicable edge computing at long last.

Sensors and Actuators

Device-bound AI services require sensors and actuators on the edge device to be able to capture the context and provide ambient support to nearby (mobile) users at all – both allow user interaction as well. In practice, today's lighting industry starts equipping street lamps with first sensors and actuators, e.g., for a smart lighting control [Gas18]. An EU-wide research initiative has proposed *The Humble Lamppost*, a project that aims to deploy 10 million 'smart' lamp posts across EU cities, addressing first use cases, such as traffic monitoring or environmental data acquisition [Nah+16]. Besides the currently considered sensors (e.g., air quality, noise, temperature) and actuators (e.g., displays, speakers), new promising hardware components should be investigated to enable innovative applications. As novel actuators, parallel reality displays [DL19], short-throw or laser-based projections can be used to provide situation-aware, personalized information to mobile users.

Networking and Communication

Connectivity to the Internet, end-user/'things' or other street lamps is an essential ability of the `SLaaP` concept: resource-demanding AI services that overstrain batteries of mobile platforms such as smartphones or AR-Head-mounted Displays (HMDs) can be offloaded to the edge device; and, among others, AI algorithms/models can be subsequently downloaded from the respective provider's cloud to protect their IPs (see the proposed protection mechanisms in Chapter 5). The former requires an ad-hoc (wireless) direct link; the latter requires Internet access. Unfortunately, there is no economic one-size-fits-all solution to act as high-performance backhaul technologies for decentralized street-lamp networks.

For *latency-critical* applications, either fiber or 5G connectivity would be a viable option, but at a high cost [Sim+16]. Also, a *subset* of street lamps could be connected to a fiber network and acts as gateway nodes for the street lamp network. The fiber connected lamps can wirelessly relay traffic to other lamps in proximity by either WiFi or mmWave communications (street lamps are typically in line of sight with each other – see Figure 7.3). Depending on the technologies used (and their inherited *wireless range*), strategic placement on street lamps is crucial; combining with mobile base stations, a nearly city-wide *spatial coverage* can be reached, and many street lamps can be left out – see the initial case study in Section 7.2.2. Depending on the depth of relaying, support of low latency applications might be limited. Since mmWave offers multi G throughput [Zhe+15], *bandwidth* should not be an issue. In a second wave (after the first wave of 5G mobile network roll-outs from 2020 onward), street lamps can be appropriate base stations for applying such mm-wave frequencies (as increasingly dense base stations are required) – but SLaaP can already benefit from its high density of ASLs: the short distances between street lamps and (mobile) users allow high bandwidths to be achieved with relatively low energy consumption (and low radiation load); the short ranges required can also lead to a high frequency reuse potential and thus to a higher spatial multiplexing output [Gen11]. These are invaluable advantages that can enable (1) short-delay, reliably-connected edge computing and (2) low-energy (or energy-efficient) communication, which is particularly relevant in emergency situations (e.g., when the the infrastructure is powered by batteries – see next section for more details). For both, for example, UWB (ultra-wideband) or similar wireless technologies could be used [Cap+19].

For various *best-effort* services, a WiFi-mesh could act as *backhaul technology* between street lamps (see Figure 7.2, p. 139, *right*); this would not allow offering hundreds of M to/from individual lamp posts. However, such a decentralized mesh network would allow vital *emergency communications* in an accelerated way, avoiding dysfunctions and supporting the ICT resilience of smart cities.

Energy and Space Management

Finally, an ASL can play three different roles namely *consumer*, *producer* and *retailer* at the same time. As a consumer, street lamps need power to primarily operate their lighting but also the additional ICT components introduced above.

For this, an ASL relies on a balanced combination of power from the subterranean power grid (see Section 7.2.1), integrated backup batteries (*UPS*), and energy harvesting (e.g., solar)—acting as a producer. The latter two are optional but can transform an ASL into a largely energy self-sufficient infrastructure when the regular power source fails, e.g., in emergency situations like blackouts. Due to this potential for self-sufficient energy supply together with its highly decentralized nature, SLaaP is well-suited to serve as a *resilient ICT infrastructure*. This means that SLaaP can still provide important ICT functionalities (e.g., emergency communications, edge computing – see previous sections) and even operate distributed AI services in emergency situations. For

Numerical example illustrating the feasibility of a resilient infrastructure:

- *Energy harvesting.* Theoretically, a solar cell generates about 100 mW/cm^2 during the daytime, which is however highly influenced by many factors (e.g., time of day, seasonal weather,...) [Ku+16]. Assuming a realistic conversion efficiency rate of 10-20% on the Earth's surface and a typical LED lamp head of 75 x 40 cm=0.3 m^2, on which a solar panel can be mounted 'invisibly', we get an average energy harvesting level of 100 mW/cm^2 * 0.3 m^2 * 0.1-0.2 = 30-60 W. Often used but 'visible' solar panels have an area of 1 m^2 ($100) and would generate about the triple: 100-200 W.
- *Energy consumption.* Despite new LED technology, the lightning still consumes 15-100 W depending on the model and luminance. A proposed lightweight upgrade, building on a Raspberry PI3 would consume less than 5 W in total [Kau+18]. A more powerful yet compact upgrade, building on Udoo Ultra/Bolt (the latter with GPU support), would require around 12/25 W. In addition, there is the energy consumption of the individual sensors and actuators.

example, in blackout situations, the harvested energy can be sufficient to cover the ASL's power consumption under best conditions (see the *numerical example* in the box, p. 145). However, in practice, the harvesting level varies widely and is not reliable; therefore, an ASL must manage the available energy and balance the load by limiting non-critical ICT functionalities or dimming the light situation-consciously.

As a retailer, an ASL can provide charging points for electric vehicles (cars, drones, ...)—requiring permanent current from the fixed power grid. In UAV research, in particular, challenges such as 'endless' flying can now be practically addressed, e.g., by recharging or automatically replacing the battery [FHR13] at the designated parking place on top of ASLs.

7.2.3.2 *Software Platform*

To exploit the full potential of this decentralized infrastructure of (potentially interconnected) ASLs, software supporting the proposed protection mechanisms must be integrated locally, and distributed concepts/methods must be applied and refined to its characteristics. To this end, the `SLaaP` concept uses a two-layered software platform—consisting of the *local layers* and a *distributed system layer* (see Figure 7.1, p. 137). We also consider an important aspect of such a software platform on an open infrastructure, namely *security and privacy*.

Local layer

This layer running on a single street lamp should consist of three sublayers: (1) a modular *hardware abstraction layer*—to communicate with the various lower level

components while preventing direct access to the hardware; (2) an appropriate Operating System (OS)—to manage the underlying hardware resources, enable controlled software executions (e.g, permission model, process isolation, low-level power management), and support container-based virtualization, e.g., HypriotOS or balenaOS for low-energy ARM-based Single-board computers (SBCs) like Raspberry PI or Odroid, Alpine or CoreOS for x86-SBCs like Udoo, and Nvidia EGX Edge AI Platform[45] for Nvidia SBCs; (3) a *local service layer*—providing software-defined basic edge computing functionalities (for non-confidential data/code), storage, energy, and user/'things' management locally. For ensuring confidential data/code, the latter layer also contains the platform code to run a user's `PDSProxy` (see *Prerequisites* in Section 5.2.1).

Distributed system layer

For distributed operation, we introduce a *3n-tier architecture*, which extends the traditional "device-cloud" paradigm to a three-tier "device-edge-cloud" setting (*vertical*), where *n* interconnected edge nodes (i.e., augmented street lamps) participate in the middle tier (*horizontal*). The `SLaaP` concept envision a distributed OS (coined *edgeOS*) for the middle tier. *edgeOS* acts as an overlay over the local layers of ASLs and the interconnecting mesh networks; it is responsible for the *routing* and *distributed processing and storage* between ASLs, from/to the mobile devices, and from/to the cloud. *edgeOS* can also take care of the control intelligence in the 3n-tier architecture, optimizing the resource allocation to improve the overall system performance – a related survey is given by [HV19].

Security and Privacy

There are still various open security and privacy issues in `SLaaP` (analogous to those in edge computing in general), most of which arise from its nature, including the following examples: `SLaaP` or edge infrastructures are characterized by many access points near to the user, which offers a higher attack surface, not least due to the weaker perimeter security of the edge devices compared to cloud datacenters; as edge devices (e.g., ASLs) are often spatially reachable to anyone, attackers can gain more easily physical access to them. Important paths to contribute to alleviating these issues include "the development of tamper-resistant and tamper-evident enclosures, remote surveillance, and trusted platform module–based attestation". [Sat17] However, as edge devices are resource-limited and such infrastructures need to be responsive, the additional overhead for security measures can have a crucial negative impact on performance; a careful tradeoff must be found between these two aspects – a comprehensive survey of vulnerabilities and open security/privacy issues in edge computing as well as first common countermeasures is given by [RJL21].

We will now highlight two open issues concerning the protection of user-related data and providers' IPs [Meu21a], which are particularly relevant in the context of the thesis. `SLaaP` – possibly equipped with a variety of sensors – and its services

[45] https://www.nvidia.com/en-us/data-center/products/egx/ (retrieved 06/30/2021)

can collect and process user-related data in the public space (without the knowing of these users). Such data should only be processed locally for privacy reasons; in addition, a permission management must be established to determine which third-party services are allowed to access which data. Even more critical is the situation when users offload their own (sensitive) data to benefit from edge resources; the same applies to locally running AI services, which could infringe the IP rights of providers. As there are still no comprehensive protection concepts for `SLaaP` and in general edge computing infrastructures, which are operated by third parties (see Section 3.3), this thesis contributes in particular the following three building blocks that allow a responsive and more confidential operation of distributed (single-user) AI services on such open infrastructure (consisting of possibly untrusted third-party devices):

1. `openAIGraph` proposed in Chapter 4 enables design and runtime support for efficiently operating modularized (distributed) AI services;
2. `PDSProxy` proposed in Chapter 5 can setup confidential processing environments on untrusted edge devices, which can also act as intermediary to allow confidential multi-hop data streaming;
3. `PDSProxy++` that will be introduced in the next chapter (Chapter 8) enables a proactive deployment of AI services based on the user's mobility to conceal the initialization overhead of the enhanced protection mechanisms.

The *first* building block is compatible with microservices (applications are developed as a set of small individual functions that are instantiated and executed on demand [Bou+18]); this allows a more efficient distribution of AI service modules in the given architecture. The *second* building block is required to make this decentralized infrastructure (to which infrastructure provider and even attackers can have physical access) secure and confidential for both users and service providers. The *third* building block is compatible with the above mesh networks, coping with high user mobility and strict responsiveness requirements. It is important to note that these concepts should be considered complementary to the general security and privacy mechanisms of such edge platforms.

7.2.4 *Newly-enabled AI Services*

This section examines newly-enabled services that uniquely benefit from the resulting `SLaaP` infrastructure. As `SLaaP` is not only intended to serve as a platform for special types of applications (e.g., distributed AI services, as discussed in this thesis) but as a 'true' urban infrastructure, we will also discuss newly-enabled urban services in general. Figure 7.4 (*right*, p. 142) shows different kinds of newly-enabled applications and 'end-users' (including smart 'things').

As discussed, such an infrastructure has extensible lamp post hardware and a general-purpose computing platform (a distributed edge cloud) as its basic constituents (see Figure 7.4, *left*, p. 142). However, the success of this infrastructure comes from enablements and services, such that the selection of provided

hardware add-ons and base services as well as a proper bootstrapping of them will be crucial – these base services are discussed next.

7.2.4.1 *SLaaP Base Services*

This type of services constitutes the (typically general- or multi-purpose) core services whose functionalities can be used by other *base services* or by so-called *value-added services* for specific use cases (see Figure 7.4, p. 142) – latter type of services includes distributed and device-bound AI services (see Section 7.2.4.2). Base services are specially designed for the installed hardware to make them available in prepared form for many other services; they are therefore also only available if the required hardware is available on the respective ASL. Value-added services, in turn, are geared to provide a high value to the end users.

To illustrate this type of services to the reader, we exemplify below three of several novel base services that are only made possible to this extent by `SLaaP`. We made the selection depending on the degree of innovation of the base service and the installed hardware; it was important that we cover many end-users and use cases or areas, both from everyday life and in emergency situations, in which these base services can be used by value-added services.

4D-Service for Urban Scene Modeling

The first base service is particularly demanding and innovative, coined as *4D-Service*[46] below. The 4D-service can drive innovation in virtually every application domain of smart cities. It is related to the fact that augmented and virtual reality (AR/VR) are so eagerly pursued in industry that press tends to speak of a *race* between major companies[47]. In contrast to this race for 3D models and corresponding AR/VR apps in many domains, the smart city domain is still one where software is for the most part bound to 2D maps: navigation and routes, location information (w.r.t. businesses, friends, etc.), situational awareness, city planning and operations software, etc. The 4D-service shall enable smart cities to move from 2D map based apps to 3D AR/VR based ones.

To this end, street lamps equipped with 3D capturing hardware (LiDAR and/or cameras) shall provide a constant stream of visual "city situation" information, distinguishing the immobile city scenery (buildings, roads, trees, ...) from mobile elements (pedestrians, vehicles, ...). The captured scenes are automatically processed, populating a semantic-annotated 3D model of the city. Since time-series of 3D models are captured, a fourth dimension comes into play (hence the term *4D*). For public and standard app use, the mobile elements are replaced by semantically equivalent models, such that individual persons, cars, etc. cannot be detected. Access to privacy-sensitive information is tightly controlled, especially to the encrypted true representation of mobile entities stored at the edge and kept for a determined time span (rolling overwrite). The resulting 4D model enables

[46] *4D* refers to the three spatial dimensions (including depth information) and the time dimension.
[47] see, e.g., https://www.cnet.com/news/apple-augmented-reality-advantages-wwdc-2017 (retrieved 06/30/2021)

or supports various urban use cases over multiple timescales (see Figure 7.4, ③, p. 142), e.g.:

- highly demanding real-time services, especially for 3D-AR apps serving mobile users (pedestrian, public/individual transport, handicapped) with smartphones or head-mounted/-up displays
- realistic simulations for urban development/planning, people flow or impact analyses
- reliable, multi-perspective model of the "street situation", enhancing the onboard model of autonomous vehicles for making faster, safer decisions (e.g., [Qiu+18]) or enabling mobility control and traffic support (e.g., parking search) without specialized hardware
- emergency response and management support, e.g., in detecting buried people, assessing their health states (aka triage), selectively searching missing people through face recognition
- immersive visualization and interaction in VR, e.g., for planning and performing evacuations, time travel, city walks, tourism, gaming, sales, and marketing, etc.

Unmanned Aerial Traffic Support

The next base service may support Unmanned Aerial Vehicles (UAVs). For instance, latest industry research of logistic companies (e.g., *DHL Parcelcopter*, *Amazon Prime Air*)[48] uses UAVs for an innovative parcel delivery service. However, a limited flying range and unresolved safety issues arising from the necessity of flying over residential areas technically block the breakthrough in all these cases [Men+17]. ASLs can notably accelerate the everyday suitability of UAVs in urban areas by playing a key role in overcoming both issues: (1) street lamps provide recharging stations mounted on top of them, extending the UAV's flying range (see *Energy and Space Management* in Section 7.2.3); (2) street lamps acting as *waypoints*[49] span a virtual air corridor along them, i.e., a citywide network of air routes, limiting the possible crash sites. Flying along these defined and controlled flight paths in combination with the 4D-service for tracking and positioning can enable fully automated "aerial highways" (see Figure 7.4, ④). For safety and security—the main reasons for blocking their breakthrough—UAVs may have an integrated self-destruction mechanism or parachute/airbag system; street lamps can operate a catching and safety system; a "traffic police" service [Men+17] can intervene by controlling a prescribed, independently working *UAVsecurity chip* that overwrites the general software and enforces controlled landing. Using these air routes, UAV job services such as an address-independent people-to-people packet transport, wedding filming, etc. can be established. In emergencies, these

[48] see, e.g., https://interestingengineering.com/drones-for-search-and-rescue-delivery-services-take-off (retrieved 06/30/2021)

[49] A waypoint is similar to the reference point of landing within the radio-navigation system (called *ILS*) for aircraft, which provides aircraft with horizontal and vertical guidance.

UAVs can also be used elsewhere for supporting disaster management by flying rescue missions [Erd+17], providing projected evacuation advice [Kni+18], etc.

Technology-enabled Civil Protection

Last but not least, ASLs acting as a citywide surveillance infrastructure may support counter-terrorism and crime fighting. For instance, a combination of sensors (acoustic, optical, ...) mounted at the street lamp could detect gunshots or explosions, allowing fast response and the identification of perpetrators. Assessing situations with regard to risks (e.g., nightly way homes) and avoiding these risky areas, a remote or virtual companion can escort and guide a user to his destination (see Figure 7.4, ②). Preventing or detecting a serious crime (e.g., harassment, rape), police forces are directly in attendance by combining so-called *holoportation* [Ort+16], which can rely on the proposed 4D-service and thus become ubiquitous, with 3D projection techniques to produce a realistic *hologram* of a police officer. The holographic officer, who can be seen, heard and interacted almost as if he is co-present, serves as a deterrent and victim support until the real police officers or paramedics arrive there. Analogously, an injured/buried person gets holographic victim support in disasters until the rescuers arrive physically. However, systematic further development of such promising techniques is necessary to make the holographic scene more realistic and immersive – and to make it applicable in the SLaaP concept.

7.2.4.2 *SLaaP Value-Added Services, Applications, and End-Users*

Above the base services, SLaaP facilitates novel value-added services and applications for both humans and 'things', covering or digitally developing several application domains: tourism, urban mobility, marketing, sales, gaming, city planning, logistics, environmental affairs, sustainable cities, to name a few [KZ16]. These new (qualities of) applications enabled by SLaaP lead to a novel kind of interactions, labeled as *SLaaP-to-X interactions* (see Figure 7.4, ①); *X* represents the possible "end-users": (1) humans and mobile devices, (2) vehicles of all kind, (3) city inventory (e.g., buildings, places, hotspots,...) and facilities. For instance, in the context of this thesis, SLaaP can host device-bound AI services within a PDSProxy (proposed in Section 5.2) that 'moves' along user's path and personalizes his environment in a confidential way for user data and providers' code – application examples include the displaying of personal navigation/evacuation advice through laser-based projection (for more examples, see Figure 5.1, p. 80); humans can further interact with ASLs in novel ways (e.g., through vision-based gesture interactions), arising HCI challenges like obtrusiveness, occlusion, and privacy in the public multi-user environment. As a static *supportive infrastructure*, SLaaP can complement autonomous driving/flying and tackle safety and security issues by reliably detecting obstacles (e.g., pedestrians, accidents, roadworks) or the road surface state using the proposed 4D-service (see Figure 7.4, ③). In the larger context, SLaaP can be the *entry point into the smart city* for smart buildings/homes, exchanging data for intelligent lighting and energy management, etc. As the city's *'nervous system'*, SLaaP can detect environmental changes that

impact the city and autonomously adapt facilities to respond to such events (e.g., traffic regulation).

7.2.5 *Discussion: A Stakeholder Perspective*

All in all, SLaaP promises an outstanding urban infrastructure for distributed AI services and, more generally, an outstanding technological progress for smart city concepts, clearly offering many benefits for its "end-users" – however, interdisciplinary efforts by stakeholders, detailed below, will be required in the future to actually realize such a citywide 'true' infrastructure. Besides *research*, we identify four major stakeholders: *citizens*, *city*, *government*, and *economy*.

From the Citizens' Perspective

As we have learned from [Gas18], a smart city has to involve its most important part, its citizens, not only as recipients of its services but also includes them as a partner (*civic participation*), "deciding the type of city they want to live in". One crucial challenge lies in designing the ASL in ways that citizens socially and ethically accept – or merely tolerate – the new technology, considering a socio-technical design. At the same time, in addition to the proposed data/code protection mechanisms for individuals and providers, *common* security and privacy concepts for SLaaP are indispensable (see Section 7.2.3.2), especially if the infrastructure or one of its base services collects and processes user-related data (without their knowing) [Zha+17; Meu21a]. Otherwise, the consequences may be disastrous if the infrastructure is hacked or misused. Therefore, a *secure data management* [..] and *privacy-preserving data analysis techniques* (e.g., replacing privacy-critical 'objects' in the 4D-service by means of semantic modeling directly 'at the source') are desirable while ensuring a *two-way data tracking* (e.g., based on blockchains [Hal18]): (1) access and usage control—ensuring the traceability of the data use (e.g., for the compliance with the *GDPR*) and enforcing contracts or any data usage fees; (2) non-repudiation of data authorship through smart contracts—ensuring the undeniable proof of the guaranteed data confidence.

From the City Perspective

Approximately, 60-90 million street lamps exist across Europe; 75 % are older than 25 years[50]. It is therefore not uncommon that street lamps consume up to 50 % of the energy budget of a city. Large-scale retrofitting/renewal programs (e.g., in LA or NYC[51]) to highly efficient LED lamps quickly amortize and can save up to 70 % of the energy costs, representing a once-in-history opportunity (introduced as the 'LED dividend') for economically upgrading to the proposed ASLs. On the other hand, there are challenges in (1) energy-efficiently designing SLaaP, and (2) finding a good cost-benefit tradeoff of the deployed ICT components (see Section 7.2.3) that SLaaP does not lead to disproportionately high

[50] http://eu-smartcities.eu/initiatives/78/description (retrieved 06/30/2021)
[51] https://singularityhub.com/2013/11/10/new-york-city-to-replace-250000-street-lights-with-leds-by-2017 (retrieved 06/30/2021)

costs. For instance, major cities can team up with industry and further push a standard (e.g., *ImHLa* [Heu+17]) to (1) open up market competition in this segment, and (2) ensure compatibility and interoperability between different manufacturers [Bre+14]. Beyond that, a city can offer economic incentives to companies and rent out infrastructure resources to (partly) cover the costs. Other important challenges include a strategic rethinking of urban development and planning—concerning the new augmented infrastructure and air corridors—and a reliable integration of city facilities (power plants, traffic facilities, ...) and rescue organizations (fire/ambulance service, police, ...).

From the Government Perspective

The government's role is to (1) timely adopt laws that allow SLaaP with all its benefits, but at the same time, (2) review legal issues w.r.t. privacy, ethics, applicability, liability, etc., coming with the new (technological) opportunities and innovative (device-bound) AI services. For instance, one could get the idea to use the 4D-service for solving the question of fault in road accidents or other incidents (similar to dash cams). For promises of safety, many citizens are likely to accept sacrificing – at least some – privacy. A governed disclosure strategy can be realized in serious threats only (e.g., terrorist attacks, natural and human/technology-caused disasters): it would allow authorities (e.g., requiring N parties) and data owners (e.g., for alibi reasons) to carefully and selectively unhide private information to enable fast and focused emergency responses or counter-terrorism. To establish citywide air corridors for UAVs, corresponding regulations must be adjusted so that they can fly near people and safely coexist in everyday urban life [Men+17]. The prevailing legal situation will surely have the decisive influence on how UAVs can use and benefit from the new infrastructure.

From the Economy Perspective

Finally, SLaaP should be open for third parties, targeting a comprehensive involvement of industry. Specifically, the challenges lie in (1) providing *economic incentives* for companies to support the new infrastructure; (2) establishing a *open ecosystem* (including the proposed OAII concept in Section 4.1.1) with standardized interfaces [San+17], allowing companies to offer their novel services and business models for different smart cities without redeveloping.

7.3 PRIVATELY-OWNED: ROUTERS AS A PLATFORM

The previous section presented SLaaP, an infrastructure concept based on publicly-owned street lamps to (i) fill the gap of a potential 'true' public (edge computing) infrastructure and (ii) realize a relatively cost-efficient, city-wide platform for running distributed AI services in particular – SLaaP is also spatially reachable to allow device-bound AI services. While SLaaP is densely deployed and achieves a high coverage of low-latency edge resources along streets and in public areas, the coverage in privately owned areas is not sufficient.

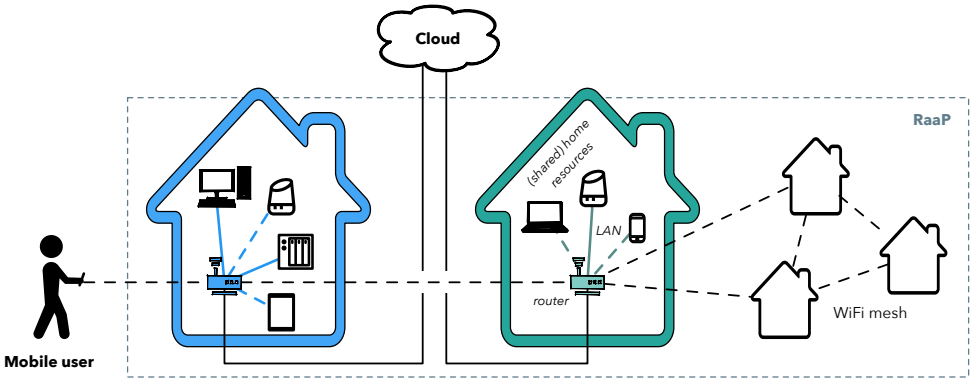


Figure 7.5: An overview of the proposed city-wide infrastructure based on upgraded and shared wireless home routers; routers can act as a hub for shared home resources. We use the following coding: different colors indicate different LANs within the respective homes/households; dashed lines depict wireless connections, and solid lines represent wired connections.

This section presents RaaP, a complementary infrastructure concept based on privately-owned wireless home routers to achieve high coverage of low-latency edge resources at low cost, especially in these private areas and indoors in which the coverage of SLaaP may be limited – it also represents a way to tackle the bootstrapping problem (see the motivation of this chapter). Specifically, RaaP proposes a sharing concept to (i) share its own resources but also (ii) act as a hub for LAN-connected shared home resources (see Figure 7.5). In this way, RaaP makes these densely-deployed, city-wide home resources (see Section 7.1) available to other mobile users at low (upgrading) cost; these edge devices are still not spatially reachable for these users (i.e., they are not suitable for device-bound AI services), but users can use them to offload their resource-demanding AI services.

Historical Background

In historical terms, such wireless home routers were – in addition to their actual purpose – first investigated by researchers as decentralized emergency communication infrastructure [Pan+12]. To the best of our knowledge, the author of this thesis was part of a research team that first proposes upgrading them as cloudlets [Meu+15b; Meu+15c; Meu+17i; Meu+17g] (*authorship*). Based on this concept, further work was carried out in cooperation with other researchers [Ngu+17b; Ngu+17c; Ngu+17a; Ged+17; Bau+17; Ngu+18; Ged+18a; Ged+18b] (*co-authored* by the author of this thesis).

Structure

The remainder of this section is structured as follows: after highlighting the identified unique characteristics of home routers (see Section 7.3.1) and presenting an initial case study (Section 7.3.2), this section discusses cost-effective upgrade options to RaaP (Section 7.3.3). Next, the section presents identified newly-enabled AI services that uniquely benefit from such an infrastructure

(Section 7.3.4). A discussion from the stakeholder perspective that highlights the benefits and the remaining challenges closes this section (Section 7.3.5).

7.3.1 *Uniquely-qualifying Characteristics*

Wireless routers are a basic networking device in almost every household in developed country today, providing access to the Internet for (W)LAN-connected devices. This leads to following characteristics that make them highly attractive for (urban) AI services.

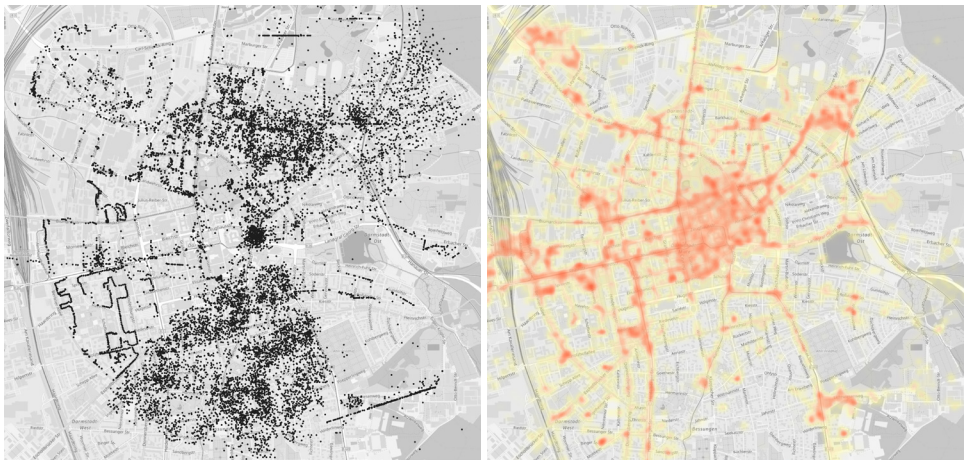
City-wide and densely deployed: Due to the dense settlement of the cities, which results in their city-wide and dense deployment, wireless home routers are already ubiquitous in our everyday urban life – the case study in Section 7.3.2 will quantify and analyze these aspects in more details.

Electrically-operated and hardware-ready: All routers are typically supplied with permanent power. As network devices, they are also equipped with the minimum hardware (namely computing and networking capabilities) required to run (offloaded) AI services or algorithms – Section 7.3.3 will discuss appropriate upgrading opportunities in more details.

Privately-owned: Private (citizen) ownership with tens and hundreds of thousands owners is ideal to share the (running) costs and establish a community-based (sharing) infrastructure, where the end users also act as infrastructure providers. On the other hand, a critical mass of participating users must first be reached, and the resulting infrastructure is dynamic in terms of its participating devices.

Cost-effectively realizable with a software update: Since appropriate hardware already exists, a simple software-based update is sufficient to exploit such routers and LAN-connected home resources; but the owners of the routers and the home resources must agree, otherwise it would be socially unacceptable – Section 7.3.5 discuss these aspects from the stakeholder perspective.

The four characteristics discussed above uniquely qualify wireless home routers as a scalable vehicle for an urban (edge computing) infrastructure. We will investigate to what extent the presented concepts of `SLaaP` are also transferable to `RaaP` and which distinctions have to be made (see Section 7.3.3). This involves first conducting an initial case study to (a) substantiate the unique potential of wireless home routers and (b) investigate how users with different movement patterns will actually use which routers and for how long in order to assess the potential utilization of privately used resources in the proposed sharing concept.



(a) Access points (b) Human mobility

Figure 7.6: Geographic locations of wireless routers collected via wardriving (*left*), and a heat map of location values collected from 30 volunteers over four weeks (*right*) – both presented by the example of Darmstadt (Germany)

7.3.2 *Case Study*

The initial case study was again conducted in Darmstadt (Germany)—a typical mid-size city (some 160,000 inhabitants). To this end, two datasets are collected in the course of this thesis to examine the *temporal* (*service*) *coverage* that `RaaP` can provide for mobile users.

7.3.2.1 *Collected Datasets*

This section first describes the two comprehensive real-world datasets—an *access points dataset* and a *human mobility dataset*—including the collection process, which both serve as a basis for the further analysis.

Access Points Dataset

To get a consistent up-to-date record of access points (APs) 'visible' (and thus theoretically-accessible) for mobile users, nearly two dozen (23) volunteers systematically walked through the city of Darmstadt and collected signal samples from wireless APs – this collection method is also known as *warwalking* or better known as *wardriving* [KFK06]. More precisely, the wardriving app – developed in student projects under the co-supervision of the author of this thesis – estimates the actual location of an AP using the tri-/multilateration method on at least three collected signal samples [Pan+12]. To further improve the data quality, (transient) mobile access points are removed from the resulting dataset by looking up non-router manufacturers from the Organizationally Unique Identifier (OUI) part of the MAC address.

Figure 7.6a illustrates the final dataset, which comprises 22,361 AP locations in an area of approximately $63\,km^2$ covering the core districts of the city. We can see that the distribution of wireless APs is really dense in residential areas and

sparse in parks and industrial areas – this represents a typical router distribution pattern in urban areas [Sap+15; JL07]. Assuming about 50,000 households[52] in the area covered and assuming that most of them have a (static) wireless home router, the collected dataset contains almost half of the wireless home routers in Darmstadt. The missing routers were not captured for several reasons: either because they were not reachable from the public areas or because too few signal samples were collected for location determination. Nevertheless, the dataset is extensive and appropriate for the purpose of this case study: it contains the same APs as required for `RaaP`, namely those that could be reached by mobile users in the same public-access areas as the *warwalking* volunteers who collected the dataset.

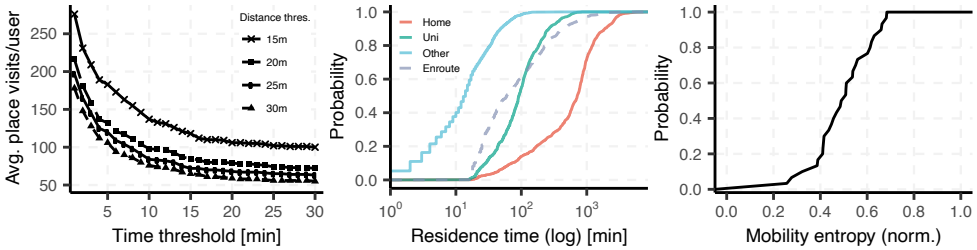
Human Mobility Dataset

To get a human mobility dataset, 30 volunteers who live in Darmstadt collected their mobility traces, among others, over four weeks. While the resulting dataset cannot be considered representative, as it is highly biased to students (N=30) and thus young people with an average age of 25.4 years (SD=2.5), it is still suitable in this initial case study to investigate which temporal (service) coverage `RaaP` could achieve (at least) for this relevant target group. To point out further limitations of the dataset in advance, the following characteristics of the target group must still be mentioned: all participants study computer science at the Technical University of Darmstadt, mostly in master program (89.8%), only a few in bachelor program (10.2%); the participants are primarily male (N=24) who have different mobility behaviors than females (N=6), as studies have shown (e.g., [Gau+20]).

Within the study, the participants use an app named `Labels` (to also collect data annotations) [Meu+15d], which is based on the `Kraken.me` platform (an automatic tracking suite) – both were mostly developed by the author of this thesis [Meu14; SS14a]. The (complete) dataset also forms the basis for further in-depth behavior analysis – for this, the reader is referred to [Meu+17c; Meu+19a] (*authorship*).

Figure 7.6b shows the mobility traces of this dataset, as only these are relevant for the case study. The final (sub-)dataset comprises over 5-million unique location values ($173,109 \pm 100,369$ location values per user) with a median (horizontal) accuracy of about $30m$. On average, the data covers more than four-fifths of the users' everyday life (83.8%, $20.1 \pm 1.6h$). The missing values are mostly in the night hours: when users are sleeping, some might have turned off their phone; the data have been smoothed accordingly, but only if the last location corresponded approximately to the first location after the gap and if the gap was not too large in time ($\leq 5h$).

[52] https://www.darmstadt.de/standort/statistik-und-stadtforschung/datenreport-2020/bevoelkerung (retrieved 06/30/2021)

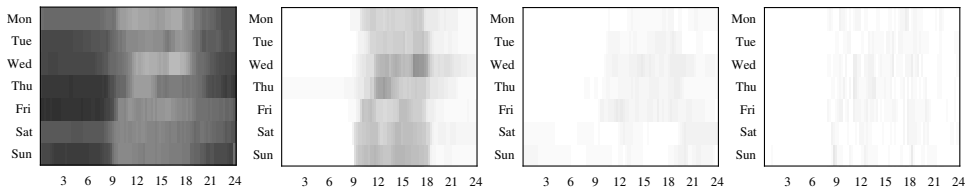


(a) Average number of (clustered) place visits per user (b) CDF of time spent in significant places (c) CDF of normalized mobility entropy of users

Figure 7.7: Analysis of location values from all 30 participants

To characterize the mobility of the participants, the most significant places and place visits are identified. The former comprises the two most important places for the student participants, namely *home* and *university*; all other places (e.g., coffee shop, friend's home) are summarized under *other places*; all location values between any places (i.e., when the user is on the move) are labeled as *en route*. The latter (place visits) are tuples of a place and a time interval, indicating how long a user stays at a particular place. The clustering algorithm used to determine these place visits was taken from [Kan+05]; it requires the specification of two threshold parameters, namely a spatial thr_s and time threshold thr_t, which have a direct influence on the cluster generation: a too small spatial threshold and time threshold could mistakenly identify too many places (the former would mistakenly generate too many clusters for the same place due to the location inaccuracy) and place visits (the latter would also mistakenly generate a cluster for short stops, e.g., when the user is waiting for the bus). The elbow method is then used to determine the best tradeoff of both thresholds [Sch+17], namely $thr_s = 25m$ and $thr_t = 10min$ – see Figure 7.7a for a visual interpretation. The labeling process of the resulting clusters is according to [TM15]: *home* is the cluster in which a user spends most of the night and early morning hours; *university* (for others: 'work') is the cluster in which a student spends most of the day hours; any other cluster is labeled as 'other places'. In total, the algorithm found 631 unique places (21.0 $\pm$ 8.2 per user) and 2,353 place visits (78.4 $\pm$ 19.2 per user).

Figure 7.7b shows the time spent in each of these places. We can see that the student participants are at *home* most of the time ($\mu_u^{home} = 860.5min$, $\sigma_u^{home} = 850.2min$, $\tilde{x}_u^{home} = 718min$) – this is obvious, as students sleep and learn primarily at home. At the university (including canteen and library), the students spend only about two hours on average ($\mu_u^{uni} = 128.6min$, $\sigma_u^{uni} = 121.5min$, $\tilde{x}_u^{uni} = 92min$) – that is about the duration of a lecture (or exercise group) plus some time for lunch or small talks with classmates and small organizational tasks. Interestingly, participants spend more time at *other places* ($\mu_u^{other} = 181.5min$, $\sigma_u^{other} = 339.1min$, $\tilde{x}_u^{other} = 60min$) than at the university. As expected, the time for transitions between two places ('enroute') is very short ($\mu_u^{trans} = 21.1min$, $\sigma_u^{trans} = 25.5min$, $\tilde{x}_u^{trans} = 14min$), as most of the participants live in Darmstadt.



(a) Home (b) University (c) Other places (d) 'Enroute'

Figure 7.8: Time distribution of daily mobility patterns of the participants (the intensity reflects the number of participants over the time of day, where *black* is a high population and *white* a low population)

A more detailed visualization of users' mobility pattern is given in Figure 7.8. We can see that the participants are at *home* especially between 6PM and 9AM (evenings and nights). There is an exception for Friday and Saturday evenings and nights when some students go out and stay overnight at *other places*. In the daytime (between 9AM to 6PM), some students go to the *university* – interestingly, some are also there on weekends (e.g., to learn in the library). As most students do not have an organized daily routine, stays in *other places* are not precisely defined. *Transitions* are similar: they usually take place between 9AM and 12 midnight with strong variations at 9AM, 12 noon, and 18PM.

To quantify the mobility of the participants, the information entropy metric is applied, which is defined by Shannon as follows [EP06; SS14b]:

$$\mathrm{H}_u(x) = -\sum_{x \in \mathcal{X}_u} p_u(x)\, log_2\, p_u(x), \tag{7.1}$$

where $p_u(x)$ is the probability measured over the respective time period that a user u has occupied a state x, which is an element of the ensemble $\mathcal{X}_u$ of distinct states. More precisely, this analysis considers the (average) weekly entropies for each user, which is calculated on the basis of 144 samples for each day (10-minute time slots)—i.e., $|\mathcal{X}_u| = 144$ and the uniform probability $p_u(x) = 1/|\mathcal{X}_u| = 1/144$. The maximal theoretical mobility entropy is about 2.32, but only normalized values are used in this analysis (also to compare it with other works, such as [EP06; Qin+12; SS14b]). Figure 7.7c shows the resulting (normalized) entropy values for the participants ($\mu_{\mathrm{H}_u} = 0.498$, $\sigma_{\mathrm{H}_u} = 0.116$, $x_{\mathrm{H}_u} \in [0.257, 0.684]$) – applying Shapiro-Wilk normality test shows that the entropy values are normally distributed ($p = 0.551$). We can see that most of the student participants live an 'entropic life', which is fairly variable and harder to predict [EP06] – there is also no apparent gender difference in this dataset.

While this human mobility dataset is biased to students and is smaller than other mobility trace datasets containing locations with GPS accuracy (e.g., GeoLife [ZXM10], Macaco [Jaf+17], Privamov [Mok+17]), this dataset offers added value in the following aspects: (1) While other datasets often contain only isolated individual mobility traces, this dataset includes mobility behavior of individual users almost throughout the day. (2) This dataset was collected in a city in which

an up-to-date dataset on access points was also available (see previous section). (3) With students, we have targeted an extreme group [EP06] that lives an 'entropic life' for the most part, often without fixed routines but very mobile. In particular, it can be difficult for such a group to achieve a high level of temporal service coverage, as it is assumed that they need to access an above-average number of routers due to their high mobility. This collected dataset is therefore well suited to study temporal service coverage as a function of mobility entropy, as will be done in the next section.

7.3.2.2 *Understanding the Temporal Coverage of Router-based Infrastructures*

While subsequent studies (e.g., [Wan+18b; Ged+18a; Ged+18b]) based on the above-introduced datasets focus on spatial coverage and placement of edge resources to optimize the overall system performance, this initial case study focuses on the possible temporal service coverage for individual mobile users. More precisely, we argue that temporal coverage is more important for the end-user experience as spatial coverage: it is of little use if almost all areas are covered, but exactly where the users spend most of their times the service is not available. On the other hand, a high spatial coverage (where hotspots are covered) can result in a high temporal (service) coverage for individual users.

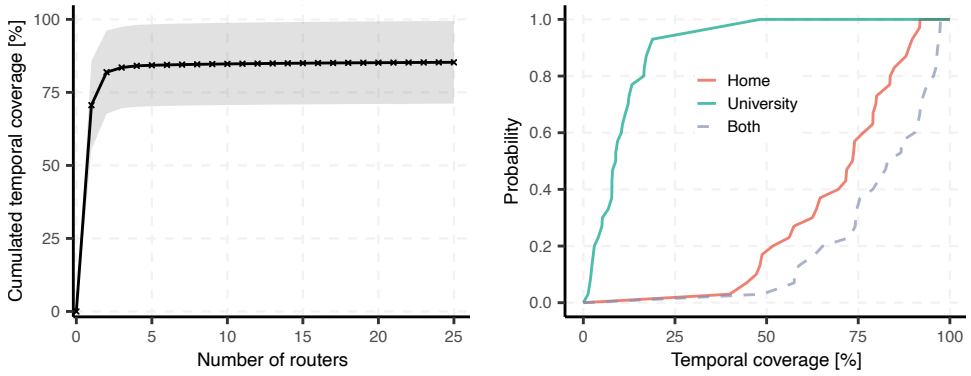
Given the available routers $\mathcal{R}$, *temporal* (*service*) *coverage* $\mathcal{C}^i$ for an individual user u_i is defined as follows [GS15; Ged+18a]:

$$\mathcal{C}^i = \frac{|\mathcal{T}^i_{covered}|}{|\mathcal{T}^i|}, \tag{7.2}$$

whereby $\mathcal{T}^i = \{\tau_1, \tau_2, .., \tau_j\}$ is the complete time span considered, $\mathcal{T}^i_{covered} \subseteq \mathcal{T}^i$ is the time span covered by routers, and τ_j specifies a small (30-second) time frame. Coverage is defined by using the *k-coverage* model; it requires at least k routers covering any time frame τ_j of the mobile user to consider it as 'covered'. In other words, a time frame is said to be covered if the user has access to at least k routers all the time. Formally, the coverage function is defined as follows:

$$(\tau_j) = \begin{cases} 1 & \text{if } \tau_j \text{ is covered by (at least) } k \text{ routers: } |\mathcal{R}_j| \geq k, \\ 0 & \text{otherwise.} \end{cases} \tag{7.3}$$

$\mathcal{R}_j \subset \mathcal{R}$ is the set of routers that fully cover the time frame τ_j. All time frames covered are then given by $\mathcal{T}^i_{covered} = \{\tau_1, \tau_2, .., \tau_l \mid c(\tau_l) = 1\}$. In concrete terms, this analysis applies a 1-coverage model ($k = 1$) to determine the time frames that are covered. We further assume a realistic 30-meter outdoor communication range of the routers [Pan+12]. In spatial terms and assuming this communication range, for example and for a later comparison, 25% of the considered home routers are already sufficient to cover over half (50%) of the city; with all routers, a maximum of about three quarters of the considered city



(a) Cumulated temporal coverage by number of routers (b) CDF of temporal coverage for user's two main places

Figure 7.9: Temporal (service) coverage for individual users

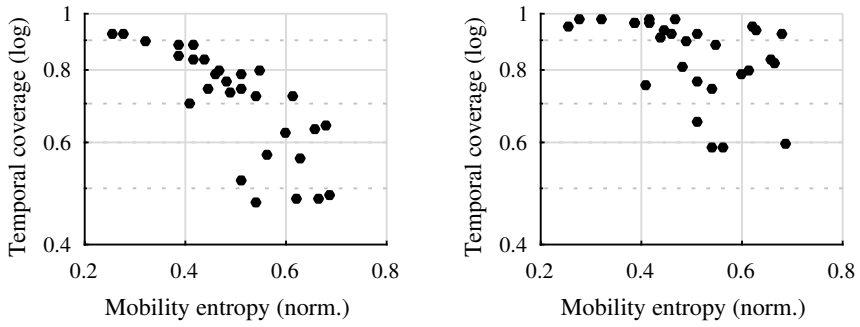
area can be covered spatially – according to [Ged+18a].

Figure 7.9a shows the cumulated temporal coverage averaged over all users as a function of the number accessible routers – the number is per user, and the routers are mostly not the same for them. We can see that only two routers located at the right places are already sufficient to achieve a high temporal coverage over 80% (μ_C = 81.9%, σ_C = 14.2%). Each additional router increases the coverage only slightly. The maximum average coverage is 86.1%; it can already be reached with an average of about 59 routers per user – additional routers or even considering all available routers have no further effect on the average (temporal) coverage.

As expected from the mobility analysis, *home* and *university* (or 'work' for others) are the two most important places for (nearly) all participants in terms of service coverage. Therefore, the coverage by a router located at each of these places is investigated in Figure 7.9b. We can see that the user's home router can already provide 80% temporal coverage for one quarter of the participants – but for over one third, it covers less than 60% of their day. An additional (second) router at the university (/workplace) increases the temporal coverage noticeably: for two thirds of the participants, both together can achieve a temporal coverage of 75% and more; for one student participant, the temporal coverage is even 97.4%.

Figure 7.10 shows the impact of the individual mobility on the temporal coverage. Users with a low mobility entropy ($H_u < 0.3$) already benefit from a high temporal coverage when using only their home routers (see Figure 7.10a). With increasing mobility entropy, the achievable temporal coverage falls rapidly. Considering the maximum number of available routers (see Figure 7.10b), even users with medium mobility entropy ($H_u < 0.5$) benefit from a high temporal coverage – as previously discovered, these users only need to have access to a medium two-digit number of routers to benefit from a similar coverage.

In view of the proposed sharing concept, the access times per router r in the user's environment were examined when the user is on the move (μ_r^{trans} = 21.5s, σ_r^{trans} = 20.5s) or stays in *other places* (μ_r^{other} = 64.1min, σ_r^{other} = 149.4min). We can



(a) n=1 (*home*) (b) n=max

Figure 7.10: Temporal (service) coverage as a function of mobility entropy of the users

see that the average access time to a router or the contact time with the router is very short (well under a minute) when the user is on the move. The situation is obviously different when the user stays in a specific place, such as a restaurant or bus stop; these places are often located in public areas and can therefore be very busy, which can be a great burden for a single router and its connected home resources as far as no further edge resources (e.g., cloudlets in the coffee shop or `SLaaP` devices) are available there.

7.3.2.3 *Key Findings and Implications*

This initial case study indicated the potential of a privately-owned router-based infrastructure, demonstrating the achievable temporal (service) coverage from the user's perspective. In the following, the key findings are summarized:

- Wireless home routers are *densely-deployed* and *large-scale* distributed throughout the city, especially in residential areas.
- With only two shared routers (possibly with LAN-connected devices) as edge resources at the user's most important places (namely at home and at work/the university), a very high temporal service coverage is already achievable, especially for users with a low entropic lifestyle ($H_u < 0.3$), which is characterized by strong patterns.
- On average, each user needs access to a medium two-digit number of routers (to most of them only for a short time, e.g., well under a minute on average when the user is on the move) to achieve a high temporal coverage over 85%. On the one hand, the results show the benefit users get when they share their own home resources; on the other hand, the results suggest that `RaaP` is particularly suitable for residential areas, where 'foreign' users are less likely to spend time in *other places*, which would result in high access times to shared routers (and their connected resources).
- All in all, a router-based infrastructure can provide high temporal (service) coverage especially for users with a low to medium mobility entropy ($H_u < 0.5$). For all others living a more entropic lifestyle, complementary infrastructures are required, especially covering public areas (such as parks and pedestrian zones), which can be either little covered by private home

routers or the shared home resources are not sufficient for long access times of many simultaneous users.

7.3.2.4 *Study Limitations*

While this case study provides some interesting findings, especially with regard to the temporal (service) coverage that can be achieved with a router-based infrastructure, it has the following limitations. *Firstly*, the access points dataset makes no claim to completeness. Although the dataset was systematically collected, the nature of the collection method (wardriving) makes it difficult to capture all access points – our conservative estimate assumes about half of all available access points in the urban area considered. Nevertheless, these are also the access points that are relevant for mobile users to offload their AI services. *Secondly*, as already mentioned at the beginning of the case study, the user base is highly biased to students and therefore young participants; the majority of them do not have very regular routines and high mobility. For instance, some users even live a more entropic lifestyle than the typical full-time employee [EP06]. This extreme user base is particularly well suited for this study, as a wide range of mobility entropy is covered. *Thridly*, this initial case study only investigates the temporal coverage from the end-user's perspective, analyzing router accesses and usages by other users to motivate the sharing concept. No statement can be made in the study about the number of simultaneous users per router. For further analysis (including investigation of spatial coverage and placement of edge resources), the reader is referred to subsequent studies based on these datasets [Ged+18a; Ged+18b; Wan+18b] (former two are *co-authored* by the author of this thesis).

7.3.3 *Upgrading Opportunity*

In the last sections, we have seen that wireless home routers by nature have the potential to form a complementary city-wide and densely-deployed (edge computing) infrastructure for operating distributed AI services. This section describes cost-effective upgrade options to exploit these home routers and possibly their LAN-connected home resources (see Figure 7.5, p. 153) – we refer to this concept as `RaaP` below.

7.3.3.1 *Hardware Components*

As wireless home routers (1) already exist in most households, (2) are permanently electrically-operated, and (3) are equipped with minimal required hardware (namely processors and networking capacities), a simple software update is all that is needed to realize `RaaP` – see next section.

First of all, however, we take a closer look at the particular HW components. By nature, home routers support IEEE 802.11 network standards that are compatible with the user's end devices; they can also handle multiple devices with low latency and high throughput. On the other hand, the processor performance

is not as optimal: the built-in CPU(s) of the more powerful routers in the upper price segment is(are) comparable with those of single-board computers such as Raspberry Pi or Odroid [Kau+18] (*co-authored* by the author of this thesis); 'standard' routers, on the other hand, are weaker. While research showed that offloading to nearby routers (of the first-mentioned category) with low utilization (i) can outperform the cloud in smaller tasks (due to lower network latency) and, what's more important, (ii) save valuable resources on the offloading mobile device [Meu+15b] (*authorship*), the wireless home routers in `RaaP` have an additional task: they act as a hub for possibly shared LAN-connected (smart) home resources, such as desktop computer, NAS, home server, notebook, tablet, and various types of IoT devices (see Figure 7.5, p. 153). In this way, home resources (in terms of computing and storage capacities) can be made available as edge resources for mobile users, and sufficient capacities can be provided to run resource-intensive AI services even from multiple simultaneous users (see Section 7.1.2).

Regarding sensors and actuators, their integration into `RaaP` makes no sense (except for the owners themselves), as the routers and other home devices are located at their owners' homes, which cannot be spatially reachable by other end-users. An exception could be ('commercial') routers, e.g., in malls, airports or train stations, when `RaaP` is applied to them.

7.3.3.2 *Software Platform*

As the hardware already exists, a cost-efficient software (firmware) update is enough to exploit routers and the LAN-connected home devices, which a user wants to share in `RaaP`. Specifically, `RaaP` uses the same two-layered software platform as `SLaaP` (see Section 7.2.3.2) but with required adjustments in the *local layer* [Meu+17g] (*authorship*): local networks between the home devices and home resources must first be virtualized to provide strict isolation between the ('private') space of the household members and the ('public') space of the edge resources provided; within the public space, a virtualization allows the isolation between AI services of different users and flexibility between the home resources is required – but this inherent tradeoff between the two aspects must be carefully balanced (see also a related discussion on this topic in Section 5.3). Virtual machines provide a good isolation but they entail a large overhead [Sat+09]; a virtual machine can be a viable path to achieve the strict isolation between private–public spaces for resource-rich desktop computers or home servers but not for resource-constrained routers or smart devices. For these (latter) devices, *Kata* containers[24] ('slimmed down' VMs) and *Google gVisor*[19] (a sandboxed container runtime) provide a good balance between isolation and resource footprint; both are also fully compatible with ARM-based architectures and container technologies (e.g., Docker, Docker Swarm, Kubernetes), which can be used to manage the distributed AI services within the available edge resources. An open issue with such edge platforms represents the (weak) protection of user data and providers' intellectual properties on third-party devices [Meu21a]. With `PDSProxy` (see Chapter 5), this thesis contributed a possible solution to

address this issue.

Similar to SLaaP, the *distributed layer* is on top of the local layers and spans the routers of several households, which can be connected to each other as WiFi mesh. Such a decentralized network would especially support mobile use cases, e.g., where results of distributed AI services are forwarded when the user gets out of range of the initially-accessed RaaP router [Ngu+18] (*co-authored* by the author of this thesis) or to proactively deploy AI services at specific edge devices (see Chapter 8). In emergency situations, RaaP can be used as emergency communication and computing infrastructure, e.g., to support emergency responses [Pan+12; Meu+17i; Ngu+17b; Ngu+17c] (the latter three are [*co-*]*authored* by the author of this thesis).

7.3.4 *Newly-enabled AI Services*

The previous section discusses technical upgrade options. This section examines newly-enabled AI services that uniquely benefit from this upgrade and the resulting RaaP infrastructure: although it is not directly suitable as a platform for device-bound AI services due to the sensor and actuator issue, its unique characteristics predestine it as a low-cost edge computing infrastructure. Specifically in the context of this thesis, RaaP can enable resource-intensive AI services on mobile platforms, such as smartphones or AR-HMDs, that overstrain mobile devices' batteries or compute power but are too bandwidth-demanding and latency-critical to be offloaded to a distant cloud or the nearest cell tower equipped with edge resources. RaaP as a fast to establish infrastructure at low cost can especially contribute to overcome the previously-mentioned (application–infrastructure) bootstrapping problem.

7.3.5 *Discussion: A Stakeholder Perspective*

RaaP promises an outstanding urban edge computing infrastructure for operating distributed AI services, clearly offering many benefits for its stakeholders. Besides *research*, the two identified stakeholders are *citizens* (acting as owner of the home resources and user at the same time) and *economy* (i.e., providers).

Given the fact that home routers and the connected home resources are privately-owned, thousands of homeowners must be convinced to reach a *critical mass* of participating devices and establish a city-wide infrastructure based on RaaP – public or commercial (infrastructure) providers could provide further incentives for this. Similar to the concepts of building a (non-)commercial public WiFi network, such as *Freifunk*[53] or *Vodafone Homespot-Service*[54], RaaP can also use such a *sharing approach*: anyone who makes their wireless home routers and possibly home resources available as edge resources for other users (who are also part of such a sharing community) can also benefit from edge resources owned

[53] https://freifunk.net/en/ (retrieved 06/30/2021)
[54] https://zuhauseplus.vodafone.de/internet-telefon/wlan-hotspots/homespot-service.html (retrieved 06/30/2021)

and shared by other users; the resource capacities that one shares can then determine how many resources of other users or complementary infrastructures (such as SLaaP, in the case the providers are the same or cooperating) are allowed to be used by this user. As found in Section 7.3.2, an average user (with a medium mobility entropy) will require access to a medium two-digit number of other routers to complement the home resources and achieve a high temporal coverage; this represents a good cost-benefit ratio, especially since other users usually only need access to a user's home resources only temporarily or for a very short time (e.g., when such users are on the move and only pass these home resources). It is also important to note that the private use of resources (private space) has higher priority than their use by others (public space); in this way, and through the use of idle resources, the homeowners should not notice a significant loss of performance.

Technically, only a software update of the routers and the home resources is required for their owners to participate in the RaaP infrastructure. Over time, this could already be preconfigured by the manufacturers to allow a quick one-click activation; manufacturers can also extend routers as a more powerful hub (for the connected home resources) or as a powerful edge resource itself.

7.4 CONCLUSION

This chapter presented SLaaP and RaaP, two decentralized open infrastructure concepts for edge computing and – in the context of this thesis – for operating distributed AI services; both infrastructures are *complementary* to each other. SLaaP proposes to exploit publicly-owned street lamps for building a full-fledged platform for distributed and in particular device-bound AI services; it also has the potential to enable a city-wide 'true' (edge computing) infrastructure and, more generally, may become *the* key driver for smart cities. RaaP proposes a sharing concept based on privately-owned home routers to enable a city-wide, densely-deployed (edge computing) platform at low (upgrading) costs. In particular, RaaP can cover specific urban areas, such as privately owned areas or indoors, which SLaaP cannot fully cover without additional deployments of street lamps. In particular, both convince by their unique characteristics, being able to achieve an unprecedented coverage for low-latency edge resources while keeping the necessary investments low.

7.4.1 *Two Unique Decentralized Infrastructure Concepts*

Table 7.2 shows a summary comparison of the two proposed open infrastructures based on the requirements established in Section 7.1.1 – supported by the case studies in this chapter. The degrees of fulfillment shown in Table 7.2 represent a first orientation for the reader; in concrete projects, dedicated measures like legal frameworks, outsourcing strategies etc. may influence these entries largely. A comprehensive comparison with the state of the art can be made using Table 7.1

Table 7.2: A summary comparison of the urban infrastructures proposed in this chapter based on the requirements/properties established in Section 7.1.1 – a comparison with other infrastructure concepts is given by Section 7.1.2 and Table 7.1.

	Global Characteristics			*Local Characteristics*			*Ownership*	
Infrastructures	(G.1) Large-scale Deployment	(G.2) Dense Deployment	(G.3) Non-isolated Infrastructure (*availability/'spatial' reachability*)	(L.1) Electrical Operation (*power grid/battery*)	(L.2) Required HW/IoT Capabilities (*Comp./Graphic Card/Storage/ Networking/Sensors/Actuators*)	(L.3) Low Deployment/Upgrading Costs	(O.1) Owner type	(O.2) Number of Owners
SLaaP	●	●	●/●	●/◐	●/◐/●/●/●/◐	◐	pu/co	1+
RaaP	●	●	●/○	●/◐	●/◐/●/●/●/●	●	pr/co	1,000+

Encoding: **fulfillment of requirements** (see Section 7.1.1): ●–fulfilled, ◐–partially fulfilled, ○–little or not fulfilled; **color coding**: ■–already-existing, ■–potentially-possible and -justifiable (but associated with corresponding costs/efforts), ○–missing for the provision of device-bound AI services; **owner type**: *pr*–private (i.e., non-commercial owners), *co*–commercial (i.e., companies), *pu*–public (e.g., cities)

(p. 136), confirming the unique cost-benefit ratios of the concepts proposed.

In terms of *global characteristics*, both stand out for their (*G.1*) large-scale and (*G.2*) dense deployments in urban areas, which show different but partially-complementary distribution patterns: SLaaP is densely-deployed along streets and sidewalks in public areas (due to the original function of raised outdoor light sources); RaaP, on the other hand, relies on wireless home routers that are placed at their owners' homes (RaaP can also exploit public or commercial routers analogously); this also allows coverage of low-latency edge resources in private areas and indoors. These different characteristics affect (*G.3*) the usability of the infrastructures for mobile users: edge resources in SLaaP and RaaP are available to all users, allowing distributed AI services. However, these edge devices are only spatially reachable in SLaaP for these users, allowing device-bound AI services; in the case of RaaP, this is typically reserved for owners and household members only.

In terms of *local characteristics*, both concepts rely on (*L.1*) existing electrically-operated 'devices' ('things') – this has (*L.3*) the decisive cost advantage that they only need to be upgraded and not deployed from scratch. This is exactly what the unique selling point of RaaP is all about: (*L.3*) only a simple, highly cost-effective software update is required to exploit home routers and possibly

other LAN-connected home resources as a city-wide, densely-deployed ICT infrastructure, avoiding a deadlock-causing device deployment. While routers are (*L.2*) already equipped with the minimum required hardware (namely computing and networking capabilities), which can be easily extended with other shared home resources, street lamps must first (*L.2+3*) be augmented with appropriate but cost-causing hardware. On the one hand, this offers the opportunity to (*L.2*) install versatile extensible hardware (including an integrated compute platform) perfectly tailored for distributed AI services and, in the larger context, for countless other innovative urban (and emergency) services. On the other hand, with the 'LED dividend', there is the once-in-history opportunity for cities to realize SLaaP with (*L.3*) relatively-low investments despite their notoriously-tight budgets.

In terms of *ownership*, both infrastructure concepts are initially (*O.1*) not commercially-driven and company-controlled, but a mixed ownership with companies – under certain conditions or regularities – can help to grow faster and share costs (see Section 7.1.1). This not sole dependence on companies can also establish trust in the infrastructure and better assure compliance with data protection regulations. For instance, once cities (*O.2* as sole owner) consider street lamps – beyond the lighting function – as ('true') infrastructure in the strict sense like road, water, energy or telecommunication infrastructures, SLaaP will become a sovereign duty: provision and appropriate access is regulated and balanced according to the public interest, professional maintenance is performed, and much more. With a privately-owned infrastructure like RaaP, this is not given, but it is faster and more cost-efficient to establish; the main challenge is to reach (*O.2*) a critical mass of participating home routers.

7.4.2 *Integration & Outlook*

The two decentralized infrastructure concept presented in this chapter each show one way to address the classic (application-infrastructure) bootstrapping problem for distributed AI services (see the motivation at the beginning of this chapter, p. 129). Specifically, both are large enough (densely-deployed) infrastructures that provide high service coverage for mobile users, even complementing each other: SLaaP constitutes a full-fledged platform for distributed AI services, providing ambient support in addition to edge resources. RaaP makes wireless home routers and their LAN-connected home resources, which are already densely distributed in the city, available to mobile users via a sharing concept (see Section 7.1); it additionally covers privately-owned areas (also indoors including public buildings, e.g., malls, airports, train stations, when transferring RaaP to these commercial networks), which are not fully covered by SLaaP without additional financial expenses.

All in all, SLaaP and RaaP each constitute a city-wide open platform for operating distributed (and partly device-bound) AI services, generating synergies as a hybrid infrastructure. On top of that, the open application ecosystem proposed

in Section 4.2 creates incentives for developers to create such applications. Applying the protection and personalization mechanisms proposed in Chapter 5 and Chapter 6, users and service providers can use these infrastructures and benefit from personalized AI services while keeping user data and AI algorithms confidential. However, the inherent initialization overhead of the latter two is still a challenge in ad-hoc mobile scenarios, where the one-time contact times to the edge devices can be short – this issue will be addressed in the next chapter.

8

PROACTIVE DEPLOYMENT OF DEVICE-BOUND AI SERVICES

The previous chapter (Chapter 7) introduced two open and scalable decentralized infrastructures – one is based on augmented street lamps (`SLaaP`), one is based on upgraded wireless home routers (`RaaP`) – as an urban platform for distributed and in particular device-bound AI services. This allows users *on the move* to ad hoc deploy and benefit from personalized and data-protected AI services. This chapter discusses *ad-hoc deployment approaches* that take advantage of these large-scale infrastructures to conceal the *response time* of such AI services.

> *"After 1 second, users get impatient and notice that they're waiting for a slow computer to respond. The longer the wait, the more this impatience grows; after about 10 seconds, the average attention span is maxed out. At that point, the user's mind starts wandering and doesn't retain enough information in short-term memory to easily resume the interaction [..]. More than 10 seconds, and you break the flow."*
>
> Jakob Nielsen, *Powers of 10: Time Scales in User Experience*, 2009 [Nie09]

With this example, Jakob Nielsen emphasizes the importance of responsive systems. Response time in particular is a crucial aspect of user experience [HD00]. In the context of ad-hoc deployment, the response time may also include (i) the transmission time and (ii) the installation and initialization time of AI services together with corresponding data protection mechanisms (see Chapter 5). The initialization process of AI services may include, among others, the transmission of user data and the personalization of AI services (see Chapter 6), and possibly a combination of multiple AI services (see Chapter 4). Depending on the characteristics of the AI service, the available bandwidths, and local resources, all these processes together can range from the low single-digit second range to the low double-digit second range (as shown in Section 5.5) – response times in the latter range inevitably lead to negative effects on the user experience, especially for short contact times with nearby IoT/edge devices.

Therefore, concealing the response time is quite challenging in this context: *ad-hoc deployment* approaches such as the `PDSProxy` concept proposed in Chapter 5 do not start until mobile users approach the target IoT/edge devices; at the same time, however, these users are already in the 'sphere of action' in which they want to use the AI services on the target devices or receive ambient support (see Figure 5.1, p. 80, as an example). *Service placement* approaches can reduce response time by pre-deploying AI services (see Section 5.5); but the personalization step, for example, still needs to be performed (under consideration of appropriate data protection measures). To improve timing, i.e., the desired AI services are prepared to provide personalized support once the user is in range, *routing-based deployment* approaches can utilize the decentralized infrastructure to proactively deploy AI services on predetermined

IoT devices. The identification of these relevant devices, however, is challenging: too many devices where the user never gets within reach would allocate valuable resources (the same is also true when the deployment starts too far in advance); too few devices where the user gets within reach could decrease the user experience (especially if they are not suitable to perform the service properly). *Mobility-based deployment* approaches can identify relevant devices, but they often require historical user data for this.

In the light of these deficits, this thesis contributes (1) a *proactive* deployment concept (termed `PDSProxy++`) to enable personalized (ambient) support immediately before the user even needs it. Specifically, `PDSProxy++` extends the `PDSProxy` concept by an algorithm that **'globally'** (within a specific area) *anticipates the IoT devices to be initialized* based on the user's mobility (see Section 8.2). `PDSProxy++` assumes (a) a city-/area-wide underlying infrastructure of IoT devices interconnected by a (de-)centralized mesh network (as introduced in Chapter 7) and (b) that each IoT device knows (or can determine) its own geographical location and its geographical neighbors. As an optional extension to `PDSProxy++`, this thesis also contributes (2) a probing algorithm that **locally** *estimates the current performance capacity of selected access-restricted IoT devices* without the need for running a profiling software on the target devices (see Section 8.3). Finally, this chapter (3) reports and discusses the evaluation results of a proof-of-concept implementation of `PDDProxy++` in three different urban scenarios—demonstrating its feasibility and its unique ability for just-in-time initialization (see Sections 8.4+8.5).

Contribution Statement: This chapter is based on the following (2) publications:

- Christian Meurisch, Dennis Werner, Bekir Bayrak, Florian Giger, and Max Mühlhäuser. "PDSProxy++: Proactive Proxy Deployment for Confidential Personalization of AI Services." In: *Proceedings of the 29th International Conference on Computer Communications and Networks*. ICCCN'20. IEEE, Aug. 2020, pp. 1–9. DOI: 10.1109/ICCCN49398.2020.9209747
- Christian Meurisch, Julien Gedeon, The An Binh Nguyen, Fabian Kaup, and Max Mühlhäuser. "Decision Support for Computational Offloading by Probing Unknown Services." In: *Proceedings of the 2017 26th International Conference on Computer Communication and Networks*. ICCCN'17. IEEE, July 2017, pp. 1–9. DOI: 10.1109/ICCCN.2017.8038406

I (*Christian Meurisch*) led the process of idea generation, conceptual design, implementation, evaluation, and writing. The students *Shashank Sridhar* and *Dennis Werner* implemented the proof-of-concept prototypes and conducted parts of the evaluation – the student *Florian Giger* supported both. *Bekir Bayrak*, *Julien Gedeon*, *The An Binh Nguyen*, *Fabian Kaup*, and *Max Mühlhäuser* provided helpful critique and comments on the conceptual design.

8.1 RELATED WORK

This section first discusses the related work on deployment approaches for device-bound AI services supporting (mobile) users.

8.1.1 *Specific Requirements*

First, this section presents a set of identified requirements that are specific to proactive deployment strategies (see the motivation of this chapter). The requirements are divided into four groups, namely *I: Identification*, *Q: Qualification*, *T: Timing*, and *A: Applicability*. The group concerning the identification of required IoT devices contains the following requirements:

- (*I.1*) *High Effectiveness:* Users cannot use device-bound AI services that have not been initialized. Therefore, deployment approaches should initialize those that a user wants or is going to use.
- (*I.2.*) *High Efficiency:* To avoid wasting resources, deployment approaches should not initialize device-bound AI services without necessity.

The group concerning the qualification of IoT devices, whether they are suitable for operating the device-bound AI services, contains the following requirements:

- (*Q.1*) *Suitability Assessment* (*capability/capacity*)*:* An initial assessment of the capabilities and current capacities of the identified IoT devices allows filtering those that are unsuitable for the proper operation of the given device-bound AI services. Therefore, means are required to assess and consider both at runtime.
- (*Q.2*) *Low Assessment Overhead:* To avoid negative performance effects, the overhead for suitability assessments should be low.

The group concerning the right timing for initialization contains the following requirements:

- (*T.1*) *In-time Initialization:* A pre-initialization of device-bound AI services allows direct use of them – this is especially crucial for short contact times in the mobile context. Therefore, deployment approaches should initialize the device-bound AI services in time.
- (*T.2*) *Runtime Optimization:* Other (partly unexpected) conditions may exist at runtime than at the design phase, especially in a mobile context. Therefore, approaches should dynamically adapt to these conditions at runtime.

Finally, the deployment approaches should be applicable in a variety of situations; the group concerning the applicability contains the following requirements:

- (*A.1*) *No Prior Knowledge Required:* Particularly in a mobile context, approaches requiring historical data from users in order to make adequate predictions pose a cold-start problem and can lead to data leaks. Therefore, deployment approaches should require no prior knowledge.

- (*A.2*) *Support for Decentralization* (*Infrastructure/Network*)*:* To avoid trusting in a central authority, deployment approaches should be designed to work in decentralized (large-scale) infrastructures and networks, as proposed by `SLaaP` and `RaaP` (see Section 7).
- (*A.3*) *Support for Mobile Scenarios:* To even support users on the move, deployment approaches should be designed to work in such mobile scenarios.
- (*A.4*) *Support for Access-restricted Systems:* For security reasons, the underlying systems typically restrict the access rights of device-bound AI services, especially if they are provided by other parties. Therefore, deployment approaches should be designed to work on such systems.
- (*A.5*) *Support for Ad-hoc Deployment:* To benefit from device-bound AI services ad hoc, deployment approaches should be designed to work in such manner.

8.1.2 *Deployment Approaches*

According to the underlying mechanisms, the different (selected) approaches that *can* be exploited to deploy device-bound AI services near the user are classified into the following sub-categories.

8.1.2.1 *Data-level Offloading* (*Service Placement*)

Approaches in this category rely on *pre-deployment* of services, requiring only *data-level offloading* from (mobile) users – related surveys are given in [WS08; SDL20]. As this category has been well-researched over the last decade, especially in the field of edge computing, only selected representative and latest approaches for this category are discussed here. For instance, with `ITEM`, Wang et al. study the problem of placing service entities containing the processing logic (e.g., scene rendering) for Virtual Reality (VR) applications in the edge environment [Wan+18a] – but in-time initialization cannot be achieved (*T.1*), as the personalization step of AI services still have to wait for the user data to be offloaded. `ITEM` optimizes economic aspects of the overall system against the service quality for the users – but it only works for the static/ offline case (*A.5*). In the online case (i.e., updating the placement decisions), `ITEM` is time-slotted, whereby always a set of users who have moved within a time slot is considered (*A.3*). `TSP` is similar to the former but additionally takes into account an assessment of utilization, energy costs, and privacy constraints (*Q.1*) [Xu+19]. In contrast, `Follow Me at the Edge` is an online service placement approach that considers the mobility and preferences of users as well (*A.3*) [OZC18; Ouy+19] – again, the latter two approaches optimize the overall system.

All in all, approaches in this category primarily focus on the placement of services, optimizing the performance and costs of the overall system (*Q.1*). The placement decision is less dependent on the individual mobility and the crucial proximity to the user to provide ambient support (*I.1*); available sensors and actuators that

can be required for device-bound AI services on the target device is not included in the decision as well. In addition, after the service is initialized, it typically still needs to be personalized with the offloaded user data – this makes an in-time initialization difficult (*T.1*).

8.1.2.2 *Method-level Offloading*

Approaches in this category can 'offload' AI services (or only parts of them) on nearby target devices – a related survey is given in [MB17]. For instance, `Echo` can automatically decide which device is best suited to run certain parts of a service (*Q.1*); this decision is based on expected completion times on the respective devices – but this, in turn, requires measurements from profilers running on each system involved (*Q.2*/*A.4*) [Lin+18]. `LI-RMLO` and `Edgent` are similar to this approach. The former additionally considers the input sizes, which have a decisive effect on the completion time [LV18]. The latter relies on the partitioning of single AI algorithms (neural networks) to find the best suitable environment for each partition [Li+19].

All in all, approaches in this category focus primarily on the efficient operation of services, aiming to reduce the load on mobile devices by full or partial offloading – this makes an in-time initialization difficult (*T.1*), as offloading does not start until the user is within range. Further, such approaches either require prior knowledge of the target device (*A.1*) or have a profiler running there (*A.4*) – but access to such information also requires special rights, which are rarely granted to user applications for security reasons.

8.1.2.3 *Routing-based Deployment*

Approaches in this category assume that the target devices or at least their locations are *already known* (*A.1*). Therefore, these approaches can use established routing algorithms to selectively deploy AI services to *predetermined* IoT devices – a survey of such algorithms for wireless ad-hoc and mesh networks is given in [AM12]. Specifically, *flooding* [Tan03] as one extreme can initialize personalized AI services on (almost) all IoT devices in the network temporally far in advance, which results in a high effectiveness (*I.1*) but low efficiency (*I.2*). More efficient approaches are based on routing algorithms that incorporate location information to cover certain areas around or ahead of the user. For instance, `DREAM` [Bas+98] uses *directional routing* (or selective/restricted flooding) to deliver content to target devices. Other approaches deliver content via a forwarding zone to all devices within static *target zones*—representing specific rectangular (e.g., [KV00; KV99; Lia+00]), circular (e.g., [Ngu+17a]), or more complex shaped geographical areas (e.g., [HLR03]).

All in all, efficient routing-based approaches require prior knowledge about which IoT devices or geographical area (*A.1*) they should target to be effective as well (*I.1*).

8.1.2.4 *Mobility-based Deployment*

Approaches in this category additionally incorporate a logic to *anticipate* the required IoT devices based on user mobility (*I.1*) – a related survey is given in [Bui+17]. Specifically, content delivery networks can pre-distribute or cache frequently requested content or services from providers close to the users (e.g., [Baş+15; Yua+14]). Complementary to this, `Breadcrumbs` tracks the movements of individual users to make connectivity predictions and identify IoT devices required in the near future [NN08]. In other fields such as vehicular networks [Ala+16; Zha+18], it is applied analogously and extended to cope with higher speeds (*A.3*). Approaches such as [Sun+16] analyze the user's interaction history to derive so-called contextual intents (which services users want to use where); based on this, the approaches make their predictions.

All in all, these approaches require more or less historical data from users (*A.1*) to make adequate predictions (*I.1*), which leads to a cold-start problem. Further, most approaches can only initialize 'default' services without prior personalization (as no user data are available in advance), which results in a timing issue (*T.1*); only some approaches base their predictions on the mobility of individuals (*A.3*); and almost no approach integrates crucial effective (but therefore also overhead-causing) protection mechanisms.

8.1.3 *Implications for this Thesis*

Table 8.1 summarizes relevant deployment approaches, showing an overall comparison of them based on the identified requirements in Section 8.1.1. We can see that there is *no* approach that completely fulfills all requirements for the proper deployment of device-bound AI services. Indeed, many challenges are either studied separately, just optimizing certain aspects only. Specifically, some approaches are static, e.g., only supporting a one-time deployment, not supporting mobile (*A.3*) or ad-hoc scenarios (*A.5*) – not to mention the lack of optimization of their deployment at runtime (*T.2*); some approaches do not assess the suitability of identified IoT devices for the given device-bound AI services (*Q.1*); and some approaches focus only on the overall system rather than on individuals. The few that consider individual mobility require prior knowledge (*A.1*), e.g., historical mobility data, to anticipate the IoT devices for which (mobile) users will come into range in the near future (*I.1*), which leads to a cold-start problem.

In the light of these deficits, this thesis contributes a concept that combines a *mobility-based* approach (to dynamically identify the required IoT devices **area-wide**) and a *method-level offloading* approach (to assess the identified devices for suitability **locally**). In particular, the former must anticipate the devices early enough to ensure that there is enough time to initialize the device-bound AI service in time (timing) – but only early enough so that the predictions are still accurate (effectiveness) as well as the services are not idle for too long and blocking valuable resources (efficiency). In terms of applicability, the proposed concept

Table 8.1: Summary of (selected) deployment approaches and representative groups of approaches (*italic*) discussed in the context of device-bound AI services

	Identif.		*Qualif.*		*Timing*		*Applicability*				
Approach / *Group of Approaches*	(I.1) High Effectiveness	(I.2) High Efficiency	(Q.1) Suitability Assessment (*capability/capacity*)	(Q.2) Low Assessment Overhead	(T.1) In-time Initialization	(T.2) Runtime Optimization	(A.1) No Prior Knowledge Required	(A.2) Support for Decentralization (*Infrastructure/Network*)	(A.3) Support for Mobile Scenarios	(A.4) Support for Access-restricted Systems	(A.5) Support for Ad-hoc Deployment
Data-level Offloading (Service Placement)											
`ITEM` [Wan+18a]	◐*	●	●/●	◐	○	○	○	●/○	◐*	◐	○
`TSP` [Xu+19]	◐*	●	●/●	◐	○	○	○	●/○	○	◐	○
`Follow Me at the Edge` [OZC18; Ouy+19]	◐*	●	●/●	◐	○	○	○	●/○	●	◐	○
Method-level Offloading											
`Echo` [Lin+18]	○	●	●/●	●	○	○	○	●/○	●	○	◐
`LI-RMLO` [LV18]	○	●	●/●	●	○	○	○	●/○	●	○	◐
`Edgent` [Li+19]	○	●	●/●	●	◐*	○	○	●/○	●	○	◐
Routing-based Deployment											
Flooding (e.g., [Tano3])	●	○	○/○	○	●	○	○	●/●	●	●	◐
Directional routing (e.g., [Bas+98])	●	◐	○/○	○	●	○	○	●/●	●	●	◐
Routing to target zones (e.g., [Ngu+17a])	◐	●	○/○	○	●	○	○	●/●	●	●	◐
Mobility-based Deployment											
Vehicular networks (e.g., [Zha+18])	◐	◐	○/○	○	●	◐	◐	●/◐	●	●	◐
`Breadcrumbs` [NN08]	◐	●	○/○	○	●	◐	○	◐*/○	●	●	○
Contextual Intent Tracking [Sun+16]	◐	●	○/○	○	●	◐	○	◐*/○	●	●	○

Encoding: **fulfillment of requirements** (see Section 8.1.1): ●–fulfilled, ◐–partially fulfilled, ○–little or not fulfilled; **way of fulfilling**: *–indirect.

additionally addresses the cold-start problem (refers to the issue of requiring historical data) and the assessment issue of access-restricted devices. Further, the concept is designed to work in mobile and ad-hoc scenarios on decentralized infrastructures in fully decentralized (mesh) networks—as proposed with `RaaP` and `SLaaP` in Chapter 7.

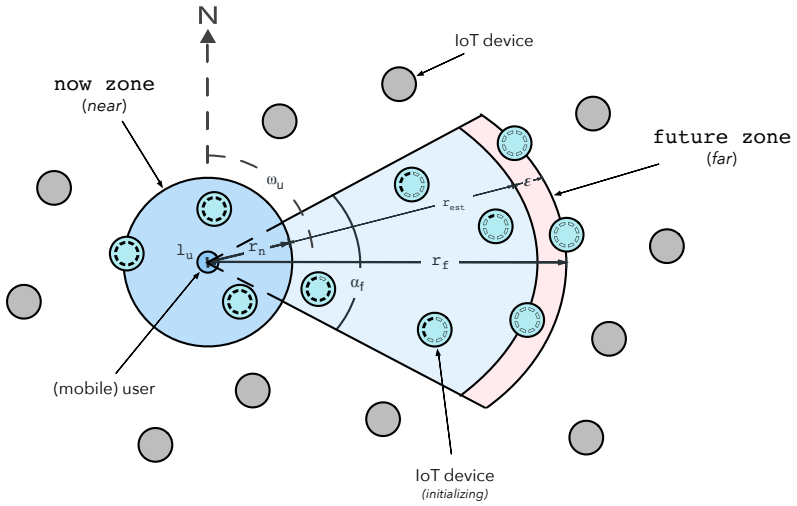


Figure 8.1: Schematic representation of the proactive deployment concept (*blue+red*): *gray circles* indicate potentially-available (nearby) IoT/edge devices; *turquoise circles* indicate identified IoT devices that must be initialized – their initialization status is visualized by the embedded progress bars.

8.2 ANTICIPATING THE IOT DEVICES REQUIRED IN THE NEAR FUTURE

This section presents the first building block of PDSProxy++: a short-term anticipatory algorithm that *identifies* required IoT devices prior to their use by mobile users – this target group primarily includes freely moving people who wear mobile devices; other mobile devices such as cars and drones are also conceivable as end users in the system. As the algorithm is based on a mobility-based approach, stationary devices due to their nature are not included.

To minimize the cold-start problem of mobility-based approaches, the proposed concept is designed to only require the most recent location values of the mobile user. This allows only short-term predictions but is sufficient to conceal the response time of ad-hoc deployment approaches such as PDSProxy, which is typically in the low- to mid-second range (as shown in Section 5.5), from the user. Figure 8.1 schematically illustrates the proposed concept – its mobility-based design and distributed operation are explained next.

8.2.1 *Mobility-based Design*

PDSProxy++ relies on two dynamic zones to identify nearby IoT devices that need to be initialized—namely the *now zone* and the *future zone*.

8.2.1.1 *The Now Zone*

The so-called *now zone* targets to identify IoT devices in the near field, namely those that are currently in the user's sphere of action. More precisely, the *now zone* defines the set of *all* IoT devices that users are able to use, e.g., due to their physical proximity. Thus, these IoT devices *must* already be ready for

use—i.e., they must be completely initialized and their provided AI services must be personalized to the user—to not negatively affect the user experience.

Spatially, PDSProxy++ defines the *now zone* as a circular area surrounding a (mobile) user u. The design decision for a circular shape takes into account spontaneous changes in movement as well as all viewing directions. The latter can be illustrated by the characteristics of the human eye [EJ86]: the peripheral vision of the human eye alone has a horizontal field of view of about 200-220° [Lun+18]; by simply turning the head (e.g., by 80-90°) in both directions, user can perceive 360°around them – even in motion. An example of a purely visual-based (interaction-free) support for mobile users includes the public display use case (see Figure 5.1c, p. 80). If a user is not moving, this user can turn 360°on the spot to interact with AI services on nearby IoT devices – as for example in the smart home use case (see Figure 5.1a, p. 80). In the following, it is therefore assumed that this circle includes all potential IoT devices that a user *may* use at the given location – independent of whether a user actually uses them.

Formally, the shape of the *now zone* can be defined by its center c_n and its radius r_n. The former is dynamically determined at the time t_i ($i \geq 0$) by the location of the user $l_u(t_i) = l_u^i = (lat^i, lng^i)_u$ (whereby both are geographic coordinates: *lat* is the latitude, and *lng* is the longitude) as follows:

$$c_n(t_i) = l_u(t_i). \tag{8.1}$$

The latter (r_n) is static at runtime; it can be configured for the current use case and/or according to the interaction or visual range of the user. The resulting set of IoT devices covered by the *now zone* is given as $\mathcal{I}_n \subset \mathcal{I}$, whereby $\mathcal{I}$ is the set of all potentially-available IoT devices.

8.2.1.2 *The Future Zone*

The so-called *future zone* targets to identify IoT devices in the far field, anticipating those that are very likely to enter the user's sphere of action in the near future. More precisely, the *future set* defines the set of all *predicted* IoT devices that potentially get within the user's range. Thus, these IoT devices *must* be initialized before they enter the user's range, or colloquially, before the user arrives.

Spatially, PDSProxy++ defines the *future zone* as a sector of a circle aligned in the direction of the user's movement. The design decision is based on the consideration that the user's future location can be determined more imprecisely from the directional vector for predictions that are further away in time (e.g., in the mid-second range and longer) than for predictions that are shorter in time (e.g., in the low-second range). Given the same speed of the user, this uncertainty is reflected in wider arcs, the further away they are from the user. When the speed changes, the *future zone* also changes as follows: the higher the speed, the larger the radius (as a larger distance are covered in the same time unit) and the smaller the opening angle (as the directional uncertainty

decreases[55] [ST19]). Conversely, the lower the speed, the smaller the radius and the higher the opening angle.

Formally, the shape of the *future zone* can be defined by its center c_f, its radius r_f, and its opening angle α_f – all three parameters (c_f, r_f, α_f) depend on the user's movement. The former (c_f) is defined analogous to the *now zone* as follows:

$$c_f(t_i) := c_n(t_i) = l_u(t_i). \tag{8.2}$$

The radius r_f is defined as follows:

$$r_f(t_i) = r_n + \underbrace{(\tau_{net} + \tau_{init}) \cdot v_u(t_i)}_{r_{est}(t_i)} + \epsilon(t_i) \tag{8.3}$$

r_{est} denotes the estimated minimum distance required to transmit the proxy code pc to the identified IoT devices (τ_{net}) and have them fully initialized (τ_{init}) by the time they enter the *now zone* – given the user's current speed v_u. The user's device must experimentally determine both τ_{net} and τ_{init} in advance – but it can adjust them at runtime by empirical values collected in the given network. More precisely, the time span τ_{net} is defined as $\tau_{net} = s_{pc} \,/\, B_{min}$, where s_{pc} depicts the size of the proxy (including the AI service and the required user data), and B_{min} is the (initially-estimated) minimal bandwidth within the network (e.g., assessable by lightweight network measurements [Kau17]). τ_{init} is the time span for the initialization of both the proxy and the AI service; the user's device can initially estimate this value based on the time it took itself for this initialization. For both parameters τ_{net} and τ_{init}, Section 8.3 contributes a *probing* approach to provide estimation support for capability-unknown, access-restricted IoT devices in a yet unexplored network.

The user's current speed $v_u(t_i)$ between two directly-successive locations l_u^i and l_u^{i-1} can be approximated by the average speed over an infinitely small time interval (Δt) as follows:

$$v_u(t_i) = \lim_{\Delta t \to 0} \frac{\Delta l_u}{\Delta t} = \lim_{t_i \to t_{i-1}} \frac{(l_u^i, l_u^{i-1})}{t_i - t_{i-1}}. \tag{8.4}$$

Equation 8.3 further introduces the ϵ parameter to compensate the estimation errors of the other parameters by dynamically adjusting r_f. In practice, these estimation errors are caused by measurement errors, non-constant network conditions or different capabilities and utilization of the heterogeneous IoT devices. In sum, these errors lead to the relevant IoT devices $\iota \in \mathcal{I}$ not being initialized in time but too late or too early – this time difference (the actual time of initial-

[55] Intuitively, it is more likely to observe more frequent and stronger directional changes at lower speeds (e.g., walking through buildings like airport hall, pedestrian zones) than at higher speeds (e.g., cycling along defined roads or driving on the highway) – in the latter only a narrower corridor ahead needs to be covered.

ization to optimal 'in-time' initialization time) is represented by $\tau^{\iota}_{wait}(t_i)$, acting as 'ground truth' retrospectively for the last iteration $i-1$ and as feedback for the current iteration i in $\epsilon(t_i)$ (see Equation 8.6). The variable τ^{ι}_{wait} is defined as follows:

$$\tau^{\iota}_{wait}(t_i) = \begin{cases} t_i - t^{\iota}_{init}, & \text{if } t_i \geq t^{\iota}_{init}. \\ \tau_{init} - \tau^{\iota}_{init}(t_i), & \text{otherwise.} \end{cases} \tag{8.5}$$

$\tau^{\iota}_{wait}(t_i)$ will be *positive*, if the AI service on the IoT device ι is already initialized: the value then indicates the time span between the completion time of the initialization t^{ι}_{init} and the current time t_i (see first line in Equation 8.5); in this case, $\tau^{\iota}_{wait}(t_i)$ represents the idle time of the initialized service until the first contact to the user. $\tau^{\iota}_{wait}(t_i)$ will be *negative*, if the AI service on the IoT device ι is not ready initialized while there is a first contact to the user (i.e., the IoT device is already in the *now zone*): the value then indicates the (estimated) missing time span that would still have been necessary to complete the initialization; it is the time span difference between the reference value τ_{init} and the current initialization progress at time t_i (see second line in Equation 8.5); the latter is given by $\tau^{\iota}_{init}(t_i) = t_i - t^{\iota}_{init_0}$ ($t^{\iota}_{init_0} \leq t_i \leq t^{\iota}_{init}$), whereby $t^{\iota}_{init_0}$ is the time when the initialization started on the given IoT device.

The ϵ-parameter in Equation 8.3 is ultimately as follows:

$$\epsilon(t_i) = \begin{cases} \epsilon(t_{i-1}) - \min\limits_{\iota \in \mathcal{I}}[\tau^{\iota}_{wait}(t_i)] \cdot v_u(t_i), & \text{if } i > 0. \\ \epsilon_0, & \text{otherwise.} \end{cases} \tag{8.6}$$

$\epsilon(t_0) = \epsilon_0$ is a constant (starting) value (see second line in Equation 8.6); if no empirical values are available yet, it is set to zero. At the time t_i ($i > 0$), the previous epsilon parameter $\epsilon(t_{i-1})$ is corrected by the additional distance that would still have been necessary in the last iteration $i-1$ to initialize all relevant IoT devices (see first line in Equation 8.6); this distance is calculated from the speed of the user $v_u(t_i)$ and the smallest $\tau^{\iota}_{wait}(t_i)$ value from all relevant IoT device $\iota \in \mathcal{I}$ (see Equation 8.5). In words, if all relevant IoT devices could be initialized early enough, $\epsilon(t_i)$ is decreased relative to the previous $\epsilon(t_{i-1})$ to shorten the idle time of the IoT devices in the future. If not all relevant IoT devices could be initialized early enough (i.e., some were initialized too late), $\epsilon(t_i)$ is increased relative to the previous $\epsilon(t_{i-1})$ to allow more time in the next iteration for the IoT devices to be completely initialized. A detailed description of how ϵ and $\tau^{\iota}_{wait}(t_i)$ is determined in practice will be given in Section 8.2.2.

As mentioned on p. 178 prior to Equation 8.2, the third parameter that describes the *future zone* is the opening angle α_f. It is defined as follows:

$$\alpha_f(t_i) = \begin{cases} \alpha_0, & \text{if } v_u(t_i) > v_{\alpha_0}. \\ \alpha_0 + (1 - \frac{v_u(t_i)}{v_{\alpha_0}})(180 - \alpha_0), & \text{otherwise.} \end{cases} \tag{8.7}$$

The design considerations from the beginning of Section 8.2.1.2 are reflected in the Equation 8.7 as follows: as the speed of the user $v_u(t_i)$ approaches a maximum threshold speed v_{α_0} (i.e., $\frac{v_u(t_i)}{v_{\alpha_0}} = 1$), the opening angle α_f decreases to the minimum constant opening angle α_0 of the *future zone* (see second line of Equation 8.7). At higher speeds ($v_u(t_i) > v_{\alpha_0}$), α_f remains constant at α_0, as only a narrow corridor ahead needs to be covered (see first line of Equation 8.7). At lower speeds ($v_u(t_i) < v_{\alpha_0}$), the opening angle α_f increases linearly ($\frac{v_u(t_i)}{v_{\alpha_0}}$, whereby $1/v_{\alpha_0}$ is the constant slope), as a wider corridor ahead needs to be covered due to the higher directional uncertainty [ST19] – until the maximum opening angle of 180 degrees is reached at a speed of $v_u(t_i) = 0$.

Finally, the direction of the user's movement at the time t_i is given as follows:

$$\vec{drt}^{\,i}_u = (lat^i_u - lat^{i-1}_u) * \vec{e}_{lat} + (lng^i_u - lng^{i-1}_u) * \vec{e}_{lng} \tag{8.8}$$

The heading angle ω_u (or absolute bearing) clockwise to the True North (or the geographical north) is then given as follows [Hun90]:

$$a1 = \cos(lat^{i-1}_u) \cdot \sin(\Delta lng) \tag{8.9}$$

$$a2 = \cos(lat^i_u)\,\sin(lat^{i-1}_u) - \sin(lat^i_u)\,\cos(lat^{i-1}_u)\,\cos(\Delta lng) \tag{8.10}$$

$$\omega_u(l^i_u, l^{i-1}_u) = \arctan2(a1, a2) \tag{8.11}$$

$$\omega^i_u := \omega_u(l^i_u, l^{i-1}_u) \tag{8.12}$$

`PDSProxy++` uses the user's current movement direction $\vec{drt}^{\,i}_u$ with the heading angle ω^i_u to align the *future zone* for the next time interval $[t_i, t_{i+1}]$. More precisely, `PDSProxy++` anticipates the user's future location $\hat{l}^{i+1}_u$ at the time t_{i+1} in the very near future, assuming a constant direction angle $\hat{\omega}_u$ and speed $\hat{v}_u$:

$$\hat{\omega}_u(\hat{l}^{i+1}_u, l^i_u) = \omega_u(l^i_u, l^{i-1}_u) \tag{8.13}$$

$$\hat{v}_u(t_{i+1}) = v_u(t_i) \tag{8.14}$$

Thus, the *future zone* is geographically in the range of $[\omega_u - \alpha_f/2, \omega_u + \alpha_f/2]$. The resulting set of IoT devices covered by the *future zone* is then given as $\mathcal{I}_f \subset \mathcal{I}$.

8.2.2 *Distributed Operation*

This section describes the operation of `PDSProxy++`. To work in a completely decentralized way on interconnected devices without a central controlling

Algorithm 4 *Proactive deployment* of AI services on nearby IoT devices from the *mobile node perspective* at the time t_i $(i > 0)$

function SENDDEPLOYMENTUPDATE(l_u^i, l_u^{i-1})
  retrieve constants $r_n, s_{pc}, \alpha_0, v_{\alpha_0}, \epsilon_0$
  initialize variables $\epsilon = \epsilon_0, B_{min}, \tau_{init}$
  $\mathcal{I}_u \leftarrow getAll1HopNeighbors(l_u^i)$
  $v_u \leftarrow determineSpeed(l_u^i, l_u^{i-1})$ (Equation 8.4)

  if $i > 1$ **then**
    $\mathcal{I}_n \leftarrow getAllNowZoneNodes(l_u^i, r_n)$
    for each $\iota \in \mathcal{I}_n$ **do**
      $\{\{\tau_{wait}^\iota\}, \{\tau_{init}^\iota\}\} \leftarrow \iota.requestStatus()$ (Equation 8.16)
    end for
    $\epsilon \leftarrow adjustErrorParameter(\epsilon, \{\tau_{wait}^\iota\}, v_u)$ (Equation 8.6)
    $\tau_{init} = max(\{\tau_{init}^\iota\}_{\delta_{95\%}})$
  end if

  $r_f \leftarrow calculateDynamicRadius(r_n, v_u, s_{pc}, B_{min}, \tau_{init}, \epsilon)$ (Equation 8.3)
  $\alpha_f \leftarrow calculateDynamicAngle(v_u, \alpha_0, v_{\alpha_0})$ (Equation 8.7)
  $\omega_u \leftarrow calculateBearing(l_u^i, l_u^{i-1})$ (Equation 8.12)

  for each $\iota \in \mathcal{I}_u$ **do**
    $\{B_{min}^\iota\} \leftarrow \iota.deployAndForward(l_u^i, r_n, r_f, \alpha_f, \omega_u)$ (Algorithm 5)
  end for
  $B_{min} = min(\{B_{min}^\iota\}_{\delta_{95\%}})$
end function

instance, `PDSProxy++` is designed as a local algorithm [Ste+17]. We further assume that the devices in the network know at least the devices that are within their wireless reachability.

Algorithm 4 shows the proactive deployment logic from the perspective of the mobile node (or the user). Specifically, the user's device runs this algorithm whenever the user moves to a new location l_u^i, as both zones must be updated (see Section 8.2.1). In practice, however, the localization of (mobile) devices is usually accompanied by inaccuracies—each time resulting in a location value l_u^i with a certain accuracy[56] $acc_{l_u^i}$. In particular, *high* inaccuracies (e.g., in case of cellular network provided locations) can result in measured location values that are very far away from the actual location of the user – this large discrepancy would cause the algorithm to initialize AI services on IoT devices near this wrong location, far away from the user. To cope with this issue, the locations with high inaccuracies are filtered first: as we have very densely-deployed IoT/edge devices in the outlined case of `SLaaP` and `RaaP` (see Chapter 7), typically closer than 25 meters to each other in the former case (see Section 7.2.2), the accuracies of the resulting locations should be in this order of magnitude.

[56] In this context, accuracy refers to the closeness of a measured location to the actual location of the device at the time of measurement.

Even if the inaccuracies are *low* (e.g., in case of GPS or partially WiFi provided locations), two different location values could be measured or obtained even though the user has not moved. Triggering the algorithm in this case would result in updates being sent that would not be necessary, as the user's real situation (or the sphere of action) has not changed – this would in turn lead to high network traffic and unnecessary use of resources. To cope with this issue and to make PDSProxy++ more resistant to, for example, such (temporary) stops and minor or frequent directional changes, the distance from the current location l_u^{i-1} (for which an update was last sent) to a new location l_u^i should be at least greater than the 'measurement error' of the localization. Unless this is the case, it is not possible to determine with a high probability whether it is an actual new location or just a 'jump' of the measured values around an actual fixed location (as detailed above). Therefore, we define the following minimum condition for a significant location change, using the mean accuracy of both measured location values as an estimate for the 'measurement error' of the localization:

$$d(l_u^i - l_u^{i-1}) > (acc_{l_u^i} + acc_{l_u^{i-1}})/2 \tag{8.15}$$

Depending on the use case, the given network conditions, the density of IoT/edge devices, and the general localization accuracy, Condition 8.15 can be further adjusted.

The algorithm itself is divided into three segments (see Algorithm 4): (*re-*)*assessment* (lines 7-14), (*re-*)*calculation* (lines 16-18), and *announcement* (lines 20-23). The former concerns the assessment of how well the previous parameters have fitted by evaluating how many device-bound AI services are actually ready for use in the *now zone*. To this end, the user's device requests the initialization status ($\tau^{\iota}_{wait}(t_i)$, see Equation 8.5) from all $\iota \in \mathcal{I}_n$ at the time t_i. As the request does not arrive at the device ι until time $t^{\iota}_{r_1}$ ($t^{\iota}_{r_1} > t_i$), the user's device must deduct this communication delay (or request time), which can be estimated with the half Round Trip Time (RTT) [TV07], from the requested value as follows:

$$\tau^{\iota}_{wait}(t_i) \approx \tau^{\iota}_{wait}(t^{\iota}_{r_1}) - RTT_{u \leftrightarrow \iota}/2 \tag{8.16}$$

Based on this assessment, PDSProxy++ adjusts the ϵ-parameter according to Equation 8.6; it also re-evaluates τ_{init} based on the requested empirical values $\{\tau^{\iota}_{init}\}$, using the 95% confidence interval $\delta_{95\%}$ (to filter outliers).

Next, PDSProxy++ recalculates the parameters (r_f, α_f) of the *future zone* and the user's movement direction (ω_u). In the latter step, PDSProxy++ announces this update to all its 1-hop surrounding IoT devices (see Algorithm 5); it also re-evaluates B_{min} based on the empirical values $\{B^{\iota}_{min}\}$ collected from the backward channel, using the 95% confidence interval $\delta_{95\%}$ (to filter outliers).

Algorithm 5 *Proactive deployment* of AI services on nearby IoT devices ι with location l_ι from an *IoT node perspective* at the (receiving) time $t^\iota_{r_2}$

```
1:  function DeployAndForward(c_n, r_n, r_f, α_f, ω_u)
2:      initialize variables l_ι, η_prec
3:
4:      if !isInsideTheZones(l_ι, c_n, r_n, r_f, α_f, ω_u) then
5:          return ∅
6:      end if
7:      if !isAlreadyInitialized() then
8:          B^ι ← requestAndInitializePDSProxy(η_prec)          (Section 5.2)
9:      end if
10:
11:     I_ι ← getAll1HopNeighbors(l_ι)
12:     for each η ∈ I_ι\{η_prec} do
13:         {B^η} ← η.deployAndForward(c_n, r_n, r_f, α_f, ω_u)          (Algorithm 5)
14:     end for
15:
16:     return {B^ι} ∪ {B^η}
17: end function
```

Algorithm 5 shows the deployment logic from the perspective of the IoT devices (or nodes). Each node ι receiving a unique[57] update at the time $t^\iota_{r_2}$ ($t^\iota_{r_2} > t_i$) runs this algorithm, which in turn is divided into two segments: *deployment* (lines 4-9) and *forwarding* (lines 11-14). The former concerns the checking of whether the node is even in one of the two zones and whether it has already been initialized: if it is not in any of the zones, the algorithm stops on the device ('terminating case'); otherwise, and unless initialized, the node requests the `PDSProxy` from the predecessor node η_{prec} (along the network path over which the update has traversed), and initializes it upon receipt (see Section 5.2) – both processes are measured to obtain empirical values (B^ι, τ^ι_{init}). At the same time, the node forwards the update to all its 1-hop neighbors (excluding η_{prec}).

Globally, this distributed algorithm terminates when each 'recursion path' has reached the terminating case, and all return values (i.e., the results of the in-network bandwidth measurements) have reached the user's device (initiator). At this point, all IoT devices within the zones are either still carrying out the initialization or have completed it. Depending on the use case and available resources, the IoT devices can finally destroy the proxy, e.g., by using timeouts, which start as soon as the devices are no longer in one of the zones or when no further updates are received, or by announcements when the user has passed.

8.3 ASSESSING THE SUITABILITY OF ACCESS-RESTRICTED IOT DEVICES

While the previous section concerns the identification of the required IoT devices prior to their use, this section concerns the *assessment* of these devices for their suitability prior to their initialization, presenting the second building block of `PDSProxy++`. Such pre-assessment may be required to support the decision

[57] All identical updates received are immediately discarded, thus terminating the algorithm locally.

process (whether to initialize the given AI service on this device). The right decision could avoid (unexpected) initialization delays and excessive processing times, which would be caused by unsuitable IoT devices. This is especially relevant in open decentralized infrastructures with heterogeneous, access-restricted devices (e.g., `RaaP`)—each with different (unknown) utilizations—as well as for device-bound AI services containing resource-intensive algorithms.

Existing assessment approaches typically require the use of profiling or monitoring systems on the target device, which can operate at different levels and collect different metrics: system-level monitoring (including VM-level and edge computing platform-level monitoring) can have access to low-level metrics such as CPU, memory, disk, and network metrics; application-level monitoring includes metrics based on response time and application throughput. [Tah+18] In our context, the response time of (offloaded) AI services is of particular interest to ensure the quality of service (QoS), e.g., in terms of a maximum tolerable response time. To this end, most approaches continuously collect system and application performance metrics to learn an accurate yet efficient prediction model, e.g., based on regression analysis [Isl+12] or deep learning [Ale19]. However, such approaches require historical performance data of the AI service or at least 'similar' services, which is challenging in ad-hoc deployment scenarios [Sie+18].

External invokers such as mobile users can either rely on the infrastructure to ensure QoS (if appropriate mechanisms exist) or make the offloading decision on their own, e.g., by requesting the above statistics or metrics from an externally accessible API (if it exists) or by collecting own empirical values [Tah+18]. Most of the approaches discussed are very accurate, taking into account the current device utilization. However, these approaches may not be applicable or available under certain constraints: target devices can be *access-restricted*, e.g., in open heterogeneous infrastructures with private owners (see Chapter 7), preventing access to monitoring system outputs; the offloading device can have *no prior knowledge* about these target devices or historical data of the service performance, which is especially common in ad-hoc scenarios (see Chapter 5).

In the light of these deficits of existing approaches, the proposed concept is particularly designed to cope with these constraints and still provide mobile users with decision support: it is based on a fast probing technique for ad-hoc and efficient collection of few empirical data required to learn a predictive model (to estimate the response time for near-future tasks) and indirectly assess the device utilization. During this short probing period, it is assumed that the device utilization and network conditions do not vary greatly. Figure 8.2 illustrates the idea behind this *probing-based* approach, which will be explained next.

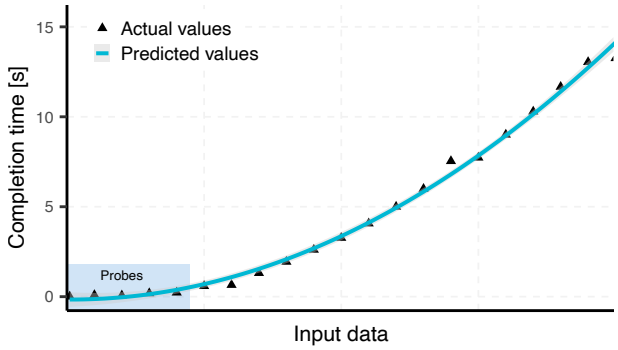


Figure 8.2: Probing-based suitability assessment of access-restricted IoT devices without requiring prior knowledge of them: small 'probes' (*blue rectangle*) are used to estimate the time (*cyan*) needed for larger tasks (characterized by more input data); *black triangles* represent the actual values.

8.3.1 *Probing-based Approach*

Inspired by network probing [RBF13], the approach employs so-called *probes* to estimate the time needed for the actual task relying on the same AI algorithm. More precisely, probes are very small computational tasks that are offloaded in advance, taking only a few milliseconds, while the actual task is far more resource-intensive, taking several seconds. This approach allows an average estimate of the response time for the actual task to be made in infrastructures with low to normal (network, device) utilization. In heavily utilized infrastructures, in which smaller tasks tend to be handled better than larger tasks, the estimate may be more optimistic than in the above case – however, if this (optimistically) estimated response time is already too high, the device can be considered unsuitable.

In the context of this thesis, such a large (resource-intensive) task may be the initialization process of the protection mechanisms (see Figure 5.7, p. 99) or the personalization of the AI services, more precisely, the training process of the underlying AI algorithms (see Table 5.3, p. 103). As the time needed for the latter typically depends on the amount of input data (see Figure 8.2), this part in particular is predestined for the proposed probing-based approach – that is why, we will focus on this in the following.

Figure 8.3 schematically shows the relevant times (*blue*) for a single *probe*. Specifically, an *assessor* device offloads a probe ρ with a data size of s^{ρ}_{req} at time t_{as_0} to a *candidate* device; it arrives there at the time t_{ca_0}. As usual, the candidate device runs the AI algorithm with the data included in the probe; the resulting completion time is given as follows:

$$\tau^{\rho}_{run} = t_{ca_1} - t_{ca_0} \qquad | \, t_{ca_1} > t_{ca_0}. \tag{8.17}$$

t_{ca_1} denotes the end of the task execution. We assume a constant utilization of the candidate node during the probing, i.e., in the very short period of $[t_{ca_0}, t_{ca_1}]$. The result (including τ^{ρ}_{run}) with the size s^{ρ}_{res} is then sent back to the assessor. Formally, the transmission times are given as follows (synchronized clocks):

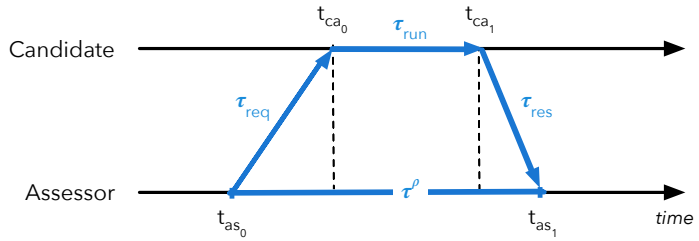


Figure 8.3: Relevant times (*blue*) for a single *probe* that is offloaded from the accessor node to the candidate node for making the suitability assessment

$$\tau^{\rho}_{req} = t_{ca_0} - t_{as_0} \quad | \, t_{ca_0} > t_{as_0}, \tag{8.18}$$

$$\tau^{\rho}_{res} = t_{as_1} - t_{ca_1} \quad | \, t_{as_1} > t_{ca_1}. \tag{8.19}$$

t_{as_1} denotes the time when the response arrives at the assessor node. Overall, the time needed for the probe is given as follows:

$$\tau^{\rho} = \tau^{\rho}_{req} + \tau^{\rho}_{run} + \tau^{\rho}_{res} \tag{8.20}$$

$$= t_{as_1} - t_{as_0} \quad | \, t_{as_1} > t_{as_0}. \tag{8.21}$$

However, the transmission times between both devices cannot be simply measured (like the completion time on only one device), as we cannot assume that their clocks are synchronized in practice. If the network and its bandwidths are known, they can be simply estimated using the following equations [TV07]:

$$B_{as \to ca} = \frac{s^{\rho}_{req}}{\tau^{\rho}_{req}}, \tag{8.22}$$

$$B_{as \leftarrow ca} = \frac{s^{\rho}_{res}}{\tau^{\rho}_{res}}. \tag{8.23}$$

$B_{as \to ca}$ denotes the upload bandwidth from the assessor node to the candidate node, while in the opposite direction $B_{as \leftarrow ca}$ denotes the download bandwidth. If these bandwidths are unknown or unstable, the proposed approach estimates them by assuming a symmetric network ($B_{as \to ca} = B_{as \leftarrow ca}$) – this is a reasonable assumption for a short RTT, small data sizes, and high bandwidths. The resulting simplified equation is given as follows:

$$\frac{s^{\rho}_{req}}{\hat{\tau}^{\rho}_{req}} = \frac{s^{\rho}_{res}}{\hat{\tau}^{\rho}_{res}}. \tag{8.24}$$

Combining Equations 8.20 and 8.24, the transmission times – and ultimately the bandwidths (using Equations 8.22 + 8.23) – can be estimated as follows:

$$\hat{\tau}^{\rho}_{req} = \frac{\tau^{\rho} - \tau^{\rho}_{run}}{1 + (s^{\rho})^{-1}}, \tag{8.25}$$

$$\hat{\tau}^{\rho}_{res} = \frac{\tau^{\rho} - \tau^{\rho}_{run}}{1 + s^{\rho}}. \tag{8.26}$$

$s^{\rho} = s^{\rho}_{req}/s^{\rho}_{res}$ is the ratio of the data sizes. This allows the assessor to measure or estimate all times on one device, bypassing the clock synchronization issue.

Next, on the basis of the executed probes $\rho_i \in \mathcal{P}$, the assessor estimates the time needed for the actual (larger) task that relies on the same algorithm and having the size s^{act}_{req} ($\gg s^{\rho_i}_{req}$). Depending on the algorithm complexity (that can be identified or investigated in the development phase), a regression model $\hat{\varphi}(s_i)$ is applied to investigates the relationship between the dependent variable (completion time) and the independent variable (size of input data) for the current case. We now exemplify this process using the following polynomial regression model:

$$\hat{\varphi}_i := \hat{\varphi}(s_i) \tag{8.27}$$

$$= a_0 + a_1 \cdot s_i + a_2 \cdot s_i^2 + ... + a_k \cdot s_i^k + \epsilon_{\hat{\varphi}}. \tag{8.28}$$

Here, the assessor knows k for the underlying algorithm; the model is then fitted to the data obtained from the probes, which determines the missing parameters $a_i (0 \leq i \leq k)$ and $\epsilon_{\hat{\varphi}}$ in the Equation 8.28. The result of $\hat{\varphi}_{act} = \hat{\varphi}(s^{act}_{req})$ then represents an estimation for the completion time τ^{act}_{run} of the actual task (see Figure 8.2).

To further estimate the transmission time τ^{act}_{req} for s^{act}_{req}, the assessor takes the average bandwidth $\overline{B}_{\rho} = |\mathcal{P}|^{-1} \sum_{\rho_i \in \mathcal{P}} \hat{B}^{\rho_i}_{as \to ca}$ obtained from the probes, using the Equation 8.22. If of interest, the transmission time τ^{act}_{res} for the response (typically of similar size as the those of the probes) can be estimated analogously.

8.3.2 *Local Operation*

Next, this section describes the local operation of `PDSProxy++` in terms of suitability assessment of access-restricted IoT devices (the so-called *candidates*).

Algorithm 6 shows the logic of probing-based suitability assessment from the perspective of the 1-hop neighboring node that has already been approved (the so-called *assessor*). Specifically, the assessor node can *optionally* run this algorithm for optimization reasons, whenever a candidate node requests the proxy code (see line 8 of Algorithm 5, p. 183) and the included AI service contains resource-intensive algorithms that may not be suitable for every device (see UC1 in Table 5.3, p. 103, for an example). It is important to note that a developer knows the time complexity and reference runtime values of the selected algorithm for probing, e.g., by analyzing it on a reference device in the

Algorithm 6 *Probing-based suitability assessment* of an access-restricted IoT device ι_{ca} (candidate) for an actual task ρ_{act} from the perspective of a 1-hop neighbor device ι_{as} (assessor)

1: **function** AssessNeighboringDevice($\iota_{ca}, \rho_{act}, \tau_{thr}$)
2: initialize variables $k, n_{\mathcal{P}}, s^{act}_{req} = size(\rho_{act}), \mathcal{M} = \{\}$
3:
4: $\mathcal{P} \leftarrow generateProbes(\rho_{act}, n_{\mathcal{P}})$
5: **for each** $\rho_i \in \mathcal{P}$ **do**
6: $s^{\rho_i}_{req} = size(\rho_i)$
7: $t_{as_0} \leftarrow now()$
8: $\{\tau^{\rho_i}_{run}, s^{\rho_i}_{res}\} \leftarrow \iota_{ca}.probeNeighboringDevice(\rho_i)$ (Equation 8.17)
9: $t_{as_1} \leftarrow now()$
10: $\tau^{\rho_i} = t_{as_1} - t_{as_0}$ (Equation 8.21)
11: $\hat{\tau}^{\rho_i}_{req} \leftarrow estimateTransmissionTime(s^{\rho_i}_{req}, s^{\rho_i}_{res}, \tau^{\rho_i}_{run})$ (Equation 8.25)
12: $\hat{B}^{\rho_i} \leftarrow estimateBandwidth(s^{\rho_i}_{req}, \hat{\tau}^{\rho_i}_{req})$ (Equation 8.22)
13: $\mathcal{M} \cup \{(s^{\rho_i}_{req}, \tau^{\rho_i}_{run}, \hat{B}^{\rho_i})\}$
14: **end for**
15:
16: $\{a_k\} \leftarrow applyRegressionModel(\mathcal{M}[s^{\rho_i}_{req}, \tau^{\rho_i}_{run}], k)$ (Equation 8.28)
17: $\hat{\varphi}_{act} \leftarrow predictRunTime(\rho_{act}, s^{act}_{req}, k, \{a_k\})$
18: $\overline{B}_{\rho} = |\mathcal{P}|^{-1} \sum_{\rho_i \in \mathcal{P}} \mathcal{M}[\hat{B}^{\rho_i}]$
19: $\hat{\tau}^{act}_{req} \leftarrow predictTransmissionTime(\overline{B}_{\rho}, s^{act}_{req})$ (Equation 8.22)
20:
21: **return** $compareWithThresholdTime(\tau_{thr}, \hat{\tau}^{act}_{req} + \hat{\varphi}_{act})$
22: **end function**

development phase.

The algorithm itself is divided into three segments: *probing* (line 4-14), *prediction* (line 16-19), and *approval* (line 21). The former concerns the performance of probes on the candidate device to obtain a quantitative estimate (see Section 8.3.1 for details). In particular, the generation of these probes $\mathcal{P}$ (line 4) is a challenge in terms of their number $n_{\mathcal{P}}$ ($:= |\mathcal{P}|$) and sizes $s^{\rho_i}_{req}$ ($\rho_i \in \mathcal{P}$): too many and too large probes are ineffective and waste unnecessary time; too few and too small probes provide too little information and are more susceptible to device load fluctuations, making the regression model inaccurate. Relatively speaking, we can say that the time needed for a probe (τ^{ρ_i}) should be several times shorter ($\tau^{\rho_i} \ll \tau^{act}$) than those of the actual task (τ^{act}); otherwise, the probing-based approach is too ineffective. Absolutely speaking, we can say that the time needed for a probe should be a few milliseconds only; otherwise, the probing-based approach generates too much overhead for the algorithms using it. This challenge is further investigated in the evaluation (see Section 8.5.2), which concludes with a final recommendation.

The next algorithm segment concerns the prediction of the times needed for the actual task. To this end, the approach first applies a regression model on the results of the probes conducted (line 16); it then uses the resulting function to predict the completion time $\hat{\varphi}_{act}$ of the actual task on the candidate device

(line 17). The approach calculates the average bandwidth (line 18) to use it for predicting the transmission time $\hat{\tau}_{req}^{act}$ of the actual task (line 19).

Finally, the approach can assess the suitability of the candidate device for the given task by comparing the predictions with the given maximum tolerated time τ_{thr} (line 21) as follows:

$$\hat{\tau}_{req}^{act} + \hat{\varphi}_{act} \leq \tau_{thr} \tag{8.29}$$

This assessment provides decision support for the assessor device in order to filter devices that would take too long (i.e., the sum of the predicted times is above the case-specific threshold) for the actual task anyway (referred to as outliers) and to avoid overloading these devices. Depending on the use case and its requirements, other flexible assessment criteria are also conceivable.

8.4 EXPERIMENTAL SETUP

This section describes the experimental setup used to conduct experiments with the prototype implementation of `PDSProxy++`. Specifically, `PDSProxy++` is evaluated in various urban test scenarios and compared against baseline approaches using emulations – explained in detail below.

8.4.1 *Emulation Environment*

The decision for a test environment always involves a compromise between *proximity to reality* (e.g., abstraction level), *flexibility* (e.g., control level, condition range), and *costs* (e.g., time/effort, money). To demonstrate the functionality of `PDSProxy++` with all its advantages in mobile scenarios (see Figure 5.1c, p. 80, as an example), a large number of test devices distributed throughout the city is required. With this requirement, a real testbed like `CitySense` [Mur+08] would be too costly and less flexible, although realistic. On the other hand, simulations like `OMNeT++` (a discrete event simulator [VH08]) and `SUMO` (an urban mobility simulator [Kra+02]) would typically not be close enough to reality, but they are cheap and very flexible. Emulations are characterized by a good tradeoff: close to reality (as real resources are used), very flexible, and feasible at limited cost [LDL18; Fri+14].

For these reasons, `Mininet-WiFi` [Fon+15] (an emulator for software-defined wireless networks) is used as the test environment. `Mininet-WiFi` separates each emulated node through resource-efficient Linux kernel namespaces (see Figure 8.4). Specifically, each node has its own isolated namespace and a virtual WiFi network gateway via the module *mac80211_hwsim*. However, this module technically limits the number of emulated nodes to 100; it also supports IEEE 802.11g only, allowing a maximal total bandwidth of 54 Mbit/s.

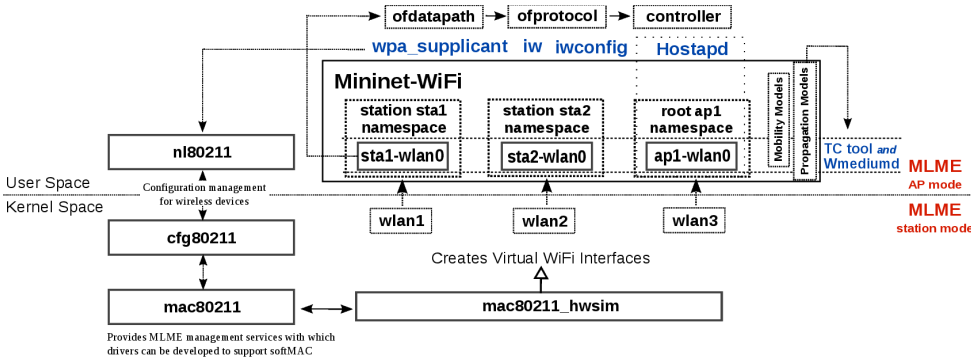


Figure 8.4: Architecture of `Mininet-WiFi` (source: *https://mininet-wifi.github.io*)

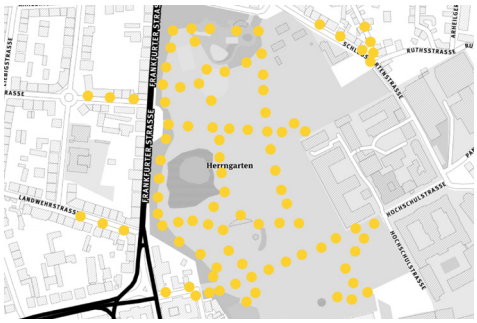
8.4.2 *Test Scenarios*

The test scenarios were constructed using a real-world dataset of an existing urban infrastructure (namely street lamps) in the city of Darmstadt, Germany – this dataset was already introduced and characterized in Section 7.2.2. More precisely, augmented street lamps serve as a platform for device-bound AI services in this setup (see the `SLaaP` concept in Section 7.2 for more details).

Figure 8.5 shows the three selected representative urban scenarios—namely *urban park* (see Figure 8.5a), *residential area* (see Figure 8.5b), and *industrial area* (see Figure 8.5c). Each scenario covers a different area with a different density of augmented street lamps. It is further important to note that all scenarios are limited to 100 nodes due to the technical limitations of `Mininet-WiFi` (see Section 8.4.1) and the resources available. Within the setup, these nodes are interconnected within a wireless mesh network, if they are in the proximity of 50 meters (see Figure 7.2, p. 139). For reproducibility, the synthetic user paths through the urban scenarios are fixed and predefined along the sidewalks.

8.4.3 *Reference Baselines*

A comparison against reference baselines from Section 8.1.2 is required to classify the performance of `PDSProxy++` in the comparative Table 8.2. To this end, three representative baselines that do not require prior knowledge were selected—namely *ad-hoc deployment*, *flooding*, and a variant of `PDSProxy++` called $\texttt{PDSProxy++}_{1\,hop}$. The former (*ad-hoc deployment*) is a reactive approach, only initializing those nodes that are already in the user's proximity – it is identical to the *now zone*. The second mentioned is a proactive approach based on *flooding*, distributing the (location) update to all reachable nodes. The latter is also a proactive one, relying on `PDSProxy++` but additionally initializing the immediately surrounding (1-hop) nodes generously (to be less sensitive to unexpected changes in movement).



(a) Urban park
($0.5\,km \times 0.5\,km = 0.25\,km^2$)

(b) Residential area
($0.7\,km \times 0.5\,km = 0.35\,km^2$)

(c) Industrial area
($1.0\,km \times 0.5\,km = 0.5\,km^2$)

Figure 8.5: Three urban test scenarios in Darmstadt (Germany) with different areas considered (*indicated in brackets*) and densities of augmented street lamps (*yellow dots*) – the possible number of devices per scenario is limited to 100 by the emulator used and the hardware resources available.

8.4.4 *Performance Metrics*

Well-known metrics from the ML community—namely *precision*, *recall*, F_1-*score* [Pow11]—are applied to assess whether the approaches correctly identify relevant IoT devices. Formally, precision and recall are defined as follows:

$$precision = \frac{tp}{tp + fp}, \tag{8.30}$$

$$recall = \frac{tp}{tp + fn}. \tag{8.31}$$

True positives (tp) are the number of nodes whose AI services are correctly initialized and *used* by the user; the cases in which they are not initialized are referred to as *false positives* (fp). Conversely, the number of nodes whose AI services are *not* used by the user is referred to as *true negatives* (tn), if they are correctly not initialized, and as *false negatives* (fn), if they are unnecessarily initialized. The latter (F_1) is the harmonic mean of the former two, defined as follows:

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{8.32}$$

$$= \frac{tp}{tp + 0.5 \cdot (tp + fn)} \tag{8.33}$$

Further, the *symmetric mean absolute percentage error* (sMAPE) is used to assess how accurately the probing approach can estimate the actual processing times of AI algorithms. Formally, this error measure is defined as follows [HK06]:

$$sMAPE = \frac{\sum_{i=1}^{vs} |\hat{\varphi}_i - \varphi_i|}{\sum_{i=1}^{vs} (\hat{\varphi}_i + \varphi_i)}. \tag{8.34}$$

By notation, vs is the number of (reference) evaluation values. φ_i denotes the actual reference value while $\hat{\varphi}_i$ denotes the forecasted value. Compared to other measures, sMAPE is a relative error measure, avoiding non-symmetric issues, e.g., by equally treating over- and under-forecasts; it is also more robust against outliers and bias effects [HK06].

Finally, it is important to note that each measurement is repeated five times, and the metrics above are applied to each; the results are then averaged.

8.5 RESULTS

This section reports on the results of the evaluation in three key aspects, namely *identification*, *qualification* and *timing*.

8.5.1 *Identification: Properly-anticipated IoT Devices*

The first set of experiments analyzes how well `PDSProxy++` can anticipate required IoT devices, especially with regard to effectiveness, efficiency, and robustness under different conditions.

8.5.1.1 *Effective and Efficient Anticipation of Required IoT Devices*

The first experiment examines the impact of the user speed v_u (*independent variable*) on the performance measured by precision, recall, and F1-score (*dependent variables*) – see Equations 8.30–8.32 for their definition. To this end, this experiment holds the following conditions (*control variables*) constant: the test scenario is the *residential area* scenario (see Figure 8.5b) with a medium density of IoT devices (approx. $285/km^2$); the test proxy (including a test AI service) is about s_{pc} = 11MB in size. The algorithm variables are initialized with $\tau_{init} = 4,913ms$ (see Section 5.5.1) and $B_{min} = 0.54 MBit/s$ (see Section 8.4.1); the constants are set to $\epsilon_0 = 0$ (as no feedback is available at t_0), $\alpha_0 = 10°$, and $v_{\alpha_0} = 10m/s$ (as only a minimum opening angle is required for fast cycling or

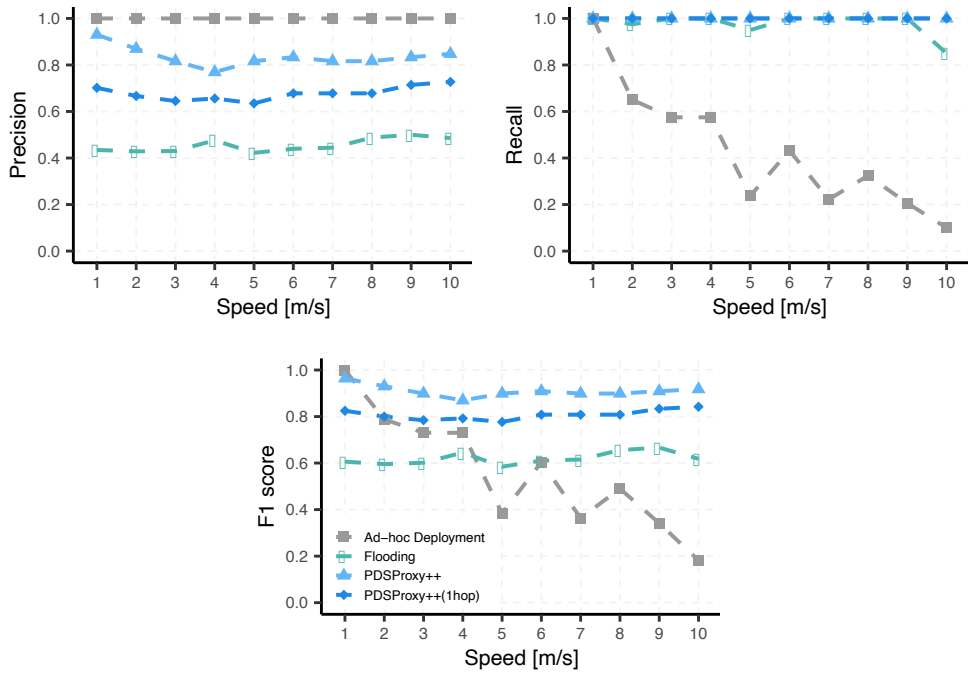


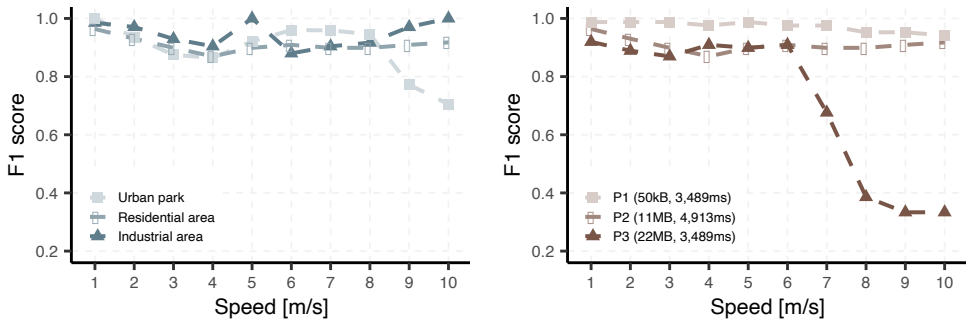
Figure 8.6: Performance comparison of the different approaches in terms of *identification*

urban driving due to fewer changes of direction).

Figure 8.6 shows the performance results of the different approaches. We can see that the *ad-hoc deployment* (*gray*) unsurprisingly achieves the highest precision, which is always 100% (per definition). The deployment approach based on *flooding* (*green*) is always under a precision of 50%, as it initializes almost all IoT devices. `PDSProxy++` (*light blue*) achieves high precision values (consistently over 76.9%; up to 93.0%). As intended, the results for $\texttt{PDSProxy++}_{1hop}$ (*dark blue*) demonstrates that the precision drops dramatically (below 72.7%) when just one hop outside the *future zone* is considered more. Regarding recall, `PDSProxy++` and its variant show their strengths by consistently achieving 100% – similar to the *flooding*-based approach. Only *ad-hoc deployment* really gets worse at speeds comparable to jogging (2-3m/s) or cycling (5-8m/s). All in all, `PDSProxy++` (almost) consistently achieves the highest F1-scores (up to 96.4%), balancing *effectiveness* and *efficiency* best, outperforming the baseline approaches.

Next, the statistical significance of these results (F1-scores) is examined. As not all data is normally distributed[58], a non-parametric Friedman test are applied in the following to support the visual impression statistically [Demo6]. According to the results for the *residential area* scenario, $\chi^2(3) = 22.4$, $p < .001$, there exists a statistically significant difference between the approaches considered. Median (and IQR) perceived effort levels for the four approaches (in alphabetical and above order) are 0.547 (0.368 to 0.730), 0.613 (0.603 to 0.638), 0.904 (0.899 to 0.916), and 0.808 (0.794 to 0.821) respectively.

[58]Shapiro-Wilk test: $p < .05$ for the *ad-hoc deployment* approach



(a) Different urban scenarios (b) Different PDS proxies

Figure 8.7: Performance of PDSProxy++ under different conditions

Post hoc analysis with Wilcoxon signed-rank tests was conducted with a Bonferroni correction applied to avoid type I errors—resulting in a new significance level set to $\alpha = .05/4 = .0125$. These pairwise comparisons reveal that there is *no* significant difference ($p = 1.000$) between an *ad-hoc deployment* ($\mu = 0.561$, $\sigma = 0.252$) and the *flooding*-based approach ($\mu = 0.620$, $\sigma = 0.027$) – but there are significant differences ($p < .001$ for both) to the better performing PDSProxy++ ($\mu = 0.909$, $\sigma = 0.025$) and its variant ($\mu = 0.808$, $\sigma = 0.021$). The latter two also differ significantly ($p = .003$) from each other, with PDSProxy++ as the best.

8.5.1.2 *Robustness in Different Scenarios*

The next experiment investigates the robustness of these results in various urban scenarios. To this end, the setup is the same as in the previous experiment (see Section 8.5.1.1), except that the scenario is also varied in this experiment.

Figure 8.7a shows the performance results of this experiment. We can see that PDSProxy++ is quite robust at low and medium speeds ($\leq 8m/s$; e.g., for walking and cycling) across all emulated urban scenarios. At higher speeds, the F1-score slightly decreases in the *urban park* scenario only. This can be explained by the different distribution of IoT devices and the more curvy footpaths within the park, which leads to more frequent (and unexpected) changes of direction.

8.5.1.3 *Robustness for Different Proxy Characteristics*

The last experiment in this group examines the robustness for different proxy variants (including the AI service). These variants are characterized by their size and initialization time – both can be very different due to the cryptographic operations required and what is transferred from the mobile device (see Section 5.2.4.1). Based on the results from Section 5.5.1.1, three sample proxy variants are extracted: $P1$ (s_{pc} = 50kB, $\tau_{init} = 3,489ms$), $P2$ (s_{pc} = 11MB, $\tau_{init} = 4,913ms$), and $P3$ (s_{pc} = 22MB, $\tau_{init} = 3,489ms$). Apart from varying the proxy variant, the setup is the same as in the first experiment (see Section 8.5.1.1).

Figure 8.7b shows the performance results of this experiment. We can see that PDSProxy++ also works quite well and is robust for different proxy variants. In

particular, PDSProxy++ can successfully adapt to the characteristics of the different proxy variants, consistently achieving an F1-score above 86.9% (and up to 98.8%) for *P1* and *P2*. For *P3* (largest in size), however, we can see that the performance decreases at higher speed ($> 6m/s$). This can be explained by the bandwidth limitation of the emulation environment used (see Section 8.4.1).

8.5.2 *Qualification: Accurate Assessments of Identified Devices*

The next experiment group analyzes how well PDSProxy++ can predict the time needed for running given algorithms on identified (access-restricted) devices.

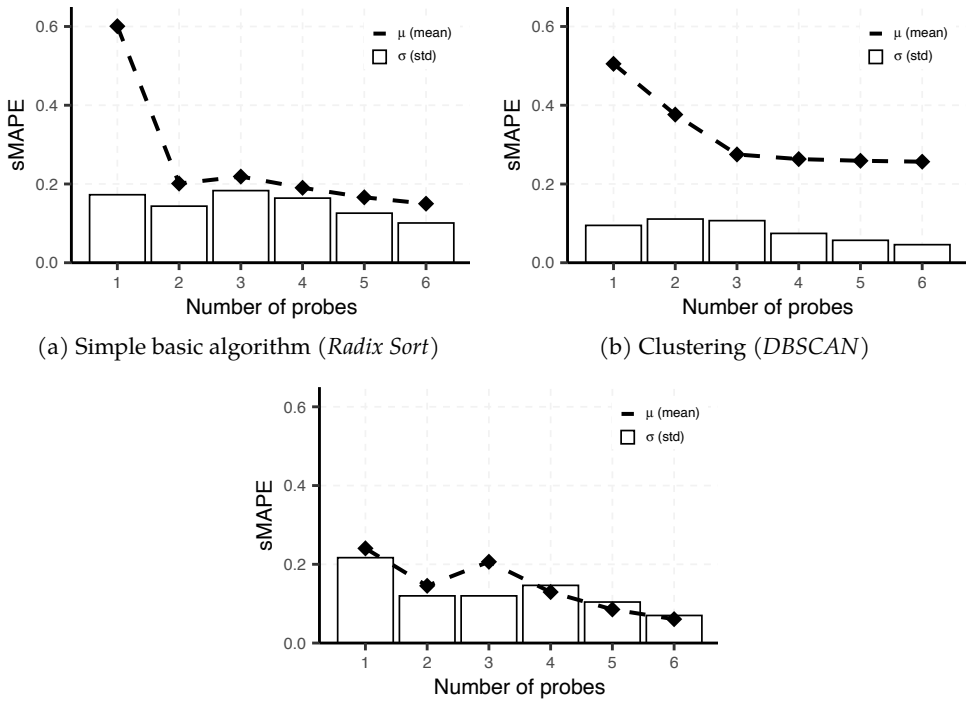
To this end, this experiment uses three algorithms of varying complexity, namely *Radix Sort*, *DBSCAN*, and *Discrete Fourier transform* (*DFT*) – extracted from the use cases introduced in Section 5.4 and Table 5.2 (p. 96). In short, the former is a simple (basic) sorting algorithm that is often used in time series analysis, such as in UC 2+3; it has linear time complexity, that is, $\mathcal{O}(n)$ [Knu98]—resulting in a linear regression model with $k = 1$. DBSCAN is a density-based data clustering algorithm (unsupervised ML) that is often used to process location data to discover mobility patterns (e.g., in UC 4); it has an average (and best) time complexity of $\mathcal{O}(n \log n)$ and $\mathcal{O}(n^2)$ in the worst case [Est+96]. The latter (DFT) is often used when processing multimedia data, such as in UC1 (face recognition); it has a time complexity of $\mathcal{O}(n^2)$ [CT65]. In this experiment, the latter two algorithms are modeled with a quadratic polynomial with $k = 2$.

As demanded in Section 8.3.2, the probes should take at least one order of magnitude less time than the actual task (to be efficient) and should be in the two to maximal middle three-digit millisecond range (to be lightweight). According to these requirements, the probes are generated, starting with the first probe with an expected (reference) time of $\tau^{ref}_{\rho_0}$ ($\ll \tau^{act}$). All further probes ρ_i ($i > 0$) are generated in such a way that their expected (reference) times are distinctly different, according to the following formula:

$$\tau^{ref}_{\rho_i} = \tau^{ref}_{\rho_0} \cdot (1 + \Delta\tau^{ref}_{\rho} \cdot i). \tag{8.35}$$

This simple strategy generates probes with equidistant distances ($\Delta\tau^{ref}_{\rho}$) between their expected times, gradually covering the above-mentioned time range. For reproducibility and compliance with the requirements, $\tau^{ref}_{\rho_0}$ is set to $100\,ms$ and $\Delta\tau^{ref}_{\rho}$ is to 0.5. This experiment further uses ten (reference) evaluation values (*evs*) for the sMAPE metric; they are selected using the same formula (see Equation 8.35), starting with $\tau^{ref}_{evs_0} = \tau^{ref}_{\rho_0} \cdot 10^1$ and setting $\Delta\tau^{ref}_{evs} = 1.0$ (to simulate actual tasks – see Table 5.2, p. 96). The measurements are repeated ten times; the results are then averaged and reported next.

Figure 8.8 shows the prediction errors (sMAPE) as a function of the number of probes conducted – lower values are better. We can see that the mean error



(a) Simple basic algorithm (*Radix Sort*)

(b) Clustering (*DBSCAN*)

(c) Signal processing (*DFT*)

Figure 8.8: Prediction error (sMAPE) as a function of the number of probes conducted for different (simple and complex) underlying algorithms

and/or the standard deviation are high for the regression models that rely on only *one* probe. In particular, the results for radix sort ($\mu_{rs_1} = 0.601$, $\sigma_{rs_1} = 0.173$) and DBSCAN ($\mu_{db_1} = 0.505$, $\sigma_{db_1} = 0.095$) indicate a very high mean error, while the results for DFT ($\mu_{dft_1} = 0.240$, $\sigma_{dft_1} = 0.217$) show a high standard deviation. When the regression models rely on *two* or more probes, the prediction errors for all algorithms decrease considerably—reaching an acceptable tradeoff between effectiveness (accurate predictions) and efficiency (as few probes as possible). For radix sort ($\mu_{rs_2} = 0.201$, $\sigma_{rs_2} = 0.144$) and DFT ($\mu_{dft_2} = 0.145$, $\sigma_{dft_2} = 0.120$), this is already the case when having two probes according to the elbow method. For DBSCAN ($\mu_{db_3} = 0.275$, $\sigma_{db_3} = 0.107$), the best tradeoff is analogously achieved when having only three probes. Each additional probe generally improves the mean error and standard deviation, but only at the expense of efficiency.

To conclude, we can say that two probes are absolutely sufficient for algorithms with accurately-modeled (average) time complexity to achieve accurate results ($\mu_2 < 0.2$, $\sigma_2 < 0.15$). For approximately-modeled (average) time complexity, at least three probes should be performed to achieve accurate results ($\mu_3 < 0.28$, $\sigma_3 < 0.11$). These recommendations indicate the minimum number of probes required to obtain useful results and high efficiency (best tradeoff) – but in general, the more probes the more accurate predictions (e.g., up to $\mu = 0.06$ and $\sigma = 0.05$ for six probes). This allows indirect conclusions to be drawn about the current utilization and the capabilities of the candidate device, respectively.

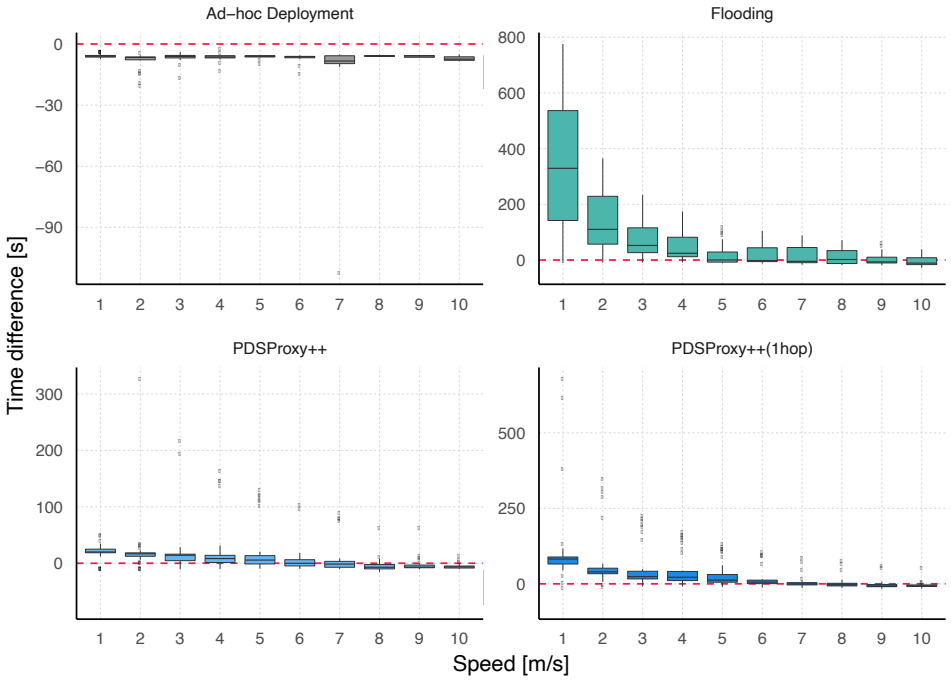


Figure 8.9: Performance comparison of the different approaches in terms of *timing*

8.5.3 *Timing: Just-in-time Initialization*

The last experiment group analyzes the timing of the approaches, or more precisely, the time difference τ^{gt}_{wait} between the time when the identified devices ($\mathcal{I}_{sel} \subseteq \mathcal{I}$) are ready for use and the time of their intended use. In contrast to the algorithms at runtime, the 'ground-truth' times can be determined in the evaluation. Formally, τ^{gt}_{wait} is defined as follows (see Equation 8.5):

$$\tau^{gt}_{wait} = t^{\iota}_{u} - t^{\iota}_{init}. \tag{8.36}$$

t^{ι}_{u} denotes the user's arrival time at the IoT device $\iota \in \mathcal{I}$ or when it enters the user's sphere of action ($\mathcal{I}_{used} \subseteq \mathcal{I}$). Negative values of Equation 8.36 indicate that a user has already arrived before the AI service is initialized, whereas positive values indicate that the AI service was initialized before. This experiment only consider the initialization process of IoT devices ($\mathcal{I}_{sel} \cap \mathcal{I}_{used}$) that are selected by the given algorithm and used by the user. Regarding the setup, it is again the same as in the first experiment (see Section 8.5.1.1).

Figure 8.9 shows the results for the timing, confirming the results from Section 8.5.1.1. The *red dashed line* in the figure ($\tau^{gt}_{wait} = 0$) represents the optimal timing – this is the case when the user's arrival coincides with the completion of the initialization. We can see that the *ad-hoc deployment* approach is always finished too late; this is obvious as it only starts when the user already needs the AI service. On the other hand, the *flooding*-based approach is almost always on time – but it achieves a very early initialization, especially at low speeds (e.g., while

Table 8.2: A summary comparison of the proactive deployment concept proposed in this chapter based on the requirements established in Section 8.1.1 – a comparison with other approaches is given by Section 8.1.2 and Table 8.1.

	Identification		*Qualification*		*Timing*		*Applicability*				
Contribution	(I.1) High Effectiveness	(I.2) High Efficiency	(Q.1) Suitability Assessment (*capability/capacity*)	(Q.2) Low Assessment Overhead	(T.1) In-time Initialization	(T.2) Runtime Optimization	(A.1) No Prior Knowledge Required	(A.2) Support for Decentralization (*Infrastructure/Network*)	(A.3) Support for Mobile Scenarios	(A.4) Support for Access-restricted Systems	(A.5) Support for Ad-hoc Deployment
`PDSProxy++`	●	●	●*/●*	◐	●	●	●	●/●	●	●	●

Encoding: **fulfillment of requirements** (see Section 8.1.1): ●–fulfilled, ◐–partially fulfilled, ○–little or not fulfilled; **way of fulfilling**: *–indirect.

walking, which is about 1-1.5 m/s). This early initialization, however, leads to inefficiency, as resources are allocated over an unnecessarily-long period of time. As designed, `PDSProxy++` almost always achieves just-in-time initialization, as can be seen from the slightly positive values; it becomes negative only at higher speeds (near v_{α_0}). The variant of `PDSProxy++` is only slightly better in terms of timing for higher speed than its base algorithm – but it is worse in terms of identification (see Section 8.5.1.1).

8.6 CONCLUSION

This chapter presented `PDSProxy++`, a proactive deployment concept that enables initializing and personalizing device-bound AI services immediately before the user even needs it. To this end, `PDSProxy++` relies on a mobility-based approach that allows short-term predictions without prior knowledge—achieving the best timing and a unique tradeoff between effectiveness and efficiency.

8.6.1 *A Unique Ad-hoc Deployment Mechanism*

Table 8.2 shows the summary assessment of `PDSProxy++` based on the requirements established in Section 8.1.1 – supported by the evaluation in this chapter. A comprehensive comparison with the state of the art can be made using Table 8.1 (p. 175), confirming the uniqueness of `PDSProxy++`.

In terms of identification, PDSProxy++ relies on very short-term predictions so that only IoT devices in the predicted area are taken into account. This results in both high effectiveness (*I.1*), as such predictions have less uncertainty, and high efficiency (*I.2*), as IoT devices are only selectively approached and initialized. This allows PDSProxy++ to *proactively* initialize device-bound AI services, which are then ready-to-use when the user needs or wants to use them.

In terms of qualification, PDSProxy++ incorporates a probing-based approach for suitability assessment (*Q.1*). The reason is to avoid unnecessary initialization of AI services on devices that are not suitable (in terms of capabilities and capacities). PDSProxy++ cannot determine these two directly – but it estimates the current load on the device (refers to capacity) and whether any necessary capabilities are missing (e.g., through error messages). The overhead for them is moderate (*Q.2*), as the probing takes relatively little time due to the use of so-called micro-tasks as probes. Although a system profiler would cause much less overhead, PDSProxy++ requires no special access rights (*A.4*)—following the Principle of Least Privilege (PoLP).

In terms of timing, PDSProxy++ achieves a just-in-time initialization (*T.1*): it initializes device-bound AI services (mostly) in time while minimizing their idle time (during which valuable device resources are blocked). PDSProxy++ manages this balancing act by reassessing the outcome of previous algorithm iterations, dynamically adapting to the given conditions at runtime in each iteration (*T.2*).

In terms of applicability, PDSProxy++ is especially designed to support mobile users in urban scenarios (*A.3*). More precisely, PDSProxy++ enables ad-hoc deployment (*A.5*) in fully-decentralized, area- or city-wide infrastructures – as proposed with SLaaP and RaaP in Section 7. To this end, it is designed as a local algorithm (*A.2*), whereby the devices know their geographically-nearest (wirelessly reachable) neighbors. Further, PDSProxy++ requires so to speak *no* prior knowledge or historical data (*A.1*) – it requires only the user's last two locations (which can be captured in a matter of seconds), minimizing the cold-start problem. Last but not least, PDSProxy++ works on access-restricted devices (*A.4*), which is well-suited for such open infrastructures and compatible with their (standard) security mechanisms.

8.6.2 *Integration & Outlook*

The proactive deployment mechanism presented in this chapter is one way of operating PDSProxy (proposed in Section 5) on decentralized urban infrastructures such as SLaaP and RaaP (proposed in Section 7). Specifically, PDSProxy++ can conceal the inherent initialization overhead of PDSProxy (caused in particular by the establishment of the underlying protection mechanisms and the ad-hoc personalization of the AI services) from mobile users – this increases the user experience, as the (non-negligible) waiting time for the just deployed and

personalized AI services is eliminated.

This can be complemented by further optimization of the initialization process of protection mechanisms and personalization of AI services; service placement approaches (e.g., from the edge computing field) can further prepare frequently-used AI services at hot spots (so that only the personalization and the protection measures for it take crucial time). For higher speeds (e.g., urban driving or driving on rural roads and highways), where the use cases are different (with less user interactions), approaches in the field of vehicular networks can be complementary (although often requiring prior knowledge).

This chapter presented the last main contribution of this thesis. Chapter 9 concludes this thesis by integrating the individual contributions and presenting directions for future work.

Part V

EPILOG

9 SUMMARY & CONCLUSIONS

This thesis argued for *data decentralization* with a strict data confinement policy in *personalized AI services* (see Section 1.2). Data decentralization provides a new framework for privacy or data protection, paving the way towards 'true' data ownership at all – even though it poses inherent challenges for AI services, especially with regard to the privacy-conflicting aspects of effective and efficient personalization and applicability. For each of the identified key challenges, this thesis proposed a building block that addresses it – all together, this thesis achieved a unique tradeoff in this field.

This chapter briefly summarizes the main contributions made by this thesis, highlighting the addressed challenges and new achievements (Section 9.1). Section 9.2 then outlines the integration of the contributions in a broader context, followed by future research directions. This thesis closes with concluding remarks.

9.1 CONTRIBUTIONS

This thesis made the following **seven (7) main contributions**. In particular, two (2) of them contribute to the *systematic understanding* of AI services and associated data protection or privacy requirements, and five are *technical contributions* – of the latter, three (3) contribute *protection mechanisms* based on data decentralization in AI services, and two (2) pave the way for a *decentralized* (*urban*) *operation*, in particular for distributed and device-bound AI services incorporating the proposed data protection mechanisms.

9.1.1 *A Systematic Understanding of AI Services*

The first two main contributions of this thesis focused on a better understanding of emerging (proactive, personalized and thus data-demanding) AI services, relying on a systematic approach (see Part II): after this thesis had explored what users really expect from them, it reviewed the state-of-the-art (data protection) approaches in terms of whether they can meet these expectations and requirements derived.

New Findings on User Expectations and Privacy Concerns Through a Proposed Study Method

First, this thesis contributed a user study that explored user expectations of proactive, personalized AI services with special attention to the privacy aspect (see Chapter 2). To this end, this thesis proposed a new study method that situated the participants in the likely context of use to capture users' (initial) expectations

towards proactive AI services. The findings revealed, among others, that most users were very open to this kind of AI services. As (proactive, personalized) AI services may be very data-demanding, partly requiring very sensitive personal data, privacy concerns, on the other hand, are one of the main reasons that prevents their breakthrough. Specifically, the study results indicated that addressing these privacy concerns would lead to greater openness of most users towards a higher level of proactivity and to the adoption of AI services at all.

A Novel Classification of Data Protection Approaches

After this thesis had explored the user expectations of emerging AI services, revealing the importance of privacy and data protection for the adoption of this technology, it was concerned with technical solutions for this problem. As the *second* contribution, this thesis first reviewed the state-of-the-art data protection approaches based on the requirements established (see Chapter 3). The result was a novel classification of these approaches along three different levels—namely management, system, and AI level. The review showed that not all of them met the established requirements of evolving AI services, i.e., there is *no* one-size-fits-all solution that completely fulfills all requirements for these AI services. Based on the review, this thesis also highlighted open research gaps. Most notably, the generally neglected challenge in the AI context is the protection of proprietary AI algorithms/ models locally or 'at the edge'. Besides this one, some other identified issues were addressed in this thesis, which will be described in more detail below.

9.1.2 *A Unique Tradeoff in Data Protection*

Taking the above findings into account, the next three main contributions were of a technical nature (see Part III). In concrete terms, this thesis proposed three coordinated data protection concepts; each of which targeted one of the above levels—namely management, system, and AI level—and achieved a unique tradeoff there.

A Unique Tradeoff at Management Level

Third, this thesis contributed a privacy-by-design platform (coined `PrivAI`) that is based on the data decentralization paradigm (see Chapter 4). Specifically, the platform includes a proposed open architecture concept (coined `OAII`) and a *data-confining* concept for personal data stores (coined `DC-PDS`). The former enabled providers to connect to the platform and operate their AI services there, both in a loosely-coupled way. In addition to its strict data confinement policy for the conventional personal data store concept, the latter also provided design and runtime support by means of a graph-based approach (coined `openAIgraph`) to address the efficiency issue of locally-running AI services. A series of evaluations showed its feasibility and efficiency. This platform forms the basis for the following contributions and building blocks.

A Unique Tradeoff at System Level

Fourth, this thesis contributed a building block (coined `PDSProxy`) to address the three-party confidentiality challenge at system level (see Chapter 5): it enables confidential processing of user data while keeping the provider's AI algorithms/models confidential (especially to protect the provider's IP); the hosting (possibly third-party) system/device is able to run AI services but without accessing the (unencrypted) code. Based on this, `PDSProxy` also proposed an ad-hoc deployment concept to deploy the protection mechanisms and personalize AI services on nearby devices. A series of evaluation showed its feasibility; the inherent (initialization) overhead caused by the protection mechanisms was significantly reduced by the optimization concepts proposed in this thesis.

A Unique Tradeoff at AI Level

Fifth, this thesis contributed a building block (coined `PNet`) to address the cold-start problem of personalization for new users by proposing a community-based approach (see Chapter 6). Specifically, this approach enabled accurate and privacy-enhancing personalization of AI services relying on supervised ML models while keeping the burden to users (in terms of labeling effort) and local devices (in terms of local resource use) lower than comparable work. `PNet` achieved this by anonymously sharing model updates with 'similar' users (i.e., users that have a similar data distribution). In this way, new users then only require to anonymously share a histogram representation of their relevant LSH-hashed data to find appropriate model updates for them through similarity comparisons. The evaluation showed the unique tradeoff between model effectiveness for new users, efficiency, and data protection.

9.1.3 *An Integrated Operation of Data-protected AI Services*

The last two main contributions dealt with the issue of how distributed and in particular device-bound AI services can be operated on a larger (*city-wide*) scale and with a *high user experience* despite the integration of the proposed overhead-causing data protection mechanisms (see Part IV). For their operation, these AI services need edge resources close to the (mobile) user to take advantage of low-latency offloading and provide ambient support – both require a densely-deployed and large-scale (edge computing) infrastructure.

Two Unique Decentralized Infrastructure Concepts

The *sixth* contribution first addressed the classic (application–infrastructure) bootstrapping problem, which is an inherent and critical issue for decentralized concepts (see Chapter 7). To this end, this thesis proposed two complementary infrastructure concepts, namely `SLaaP` (based on publicly-owned street lamps) and `RaaP` (based on privately-owned wireless home routers); initial case studies conducted showed their unique potential. `SLaaP` constitutes a full-fledged platform for distributed and in particular device-bound AI services; it also has the

potential to enable a city-wide 'true' (edge computing) infrastructure and, more generally, may become *the* key driver for smart cities. In particular, this thesis emphasized the once-in-history opportunity (that the current decade offers) for an economic realization of `SLaaP` through the 'LED dividend'. `RaaP`, on the other hand, makes wireless home routers and their LAN-connected home resources, which are already densely distributed in the city, available to mobile users via a sharing concept at low (upgrading) costs; it additionally covers privately-owned areas (also indoors including public buildings, e.g., malls, airports, train stations, when transferring `RaaP` to these commercial networks), which are not fully covered by `SLaaP` without additional financial investments. Both infrastructure concepts together can achieve a cost-efficient and unique coverage of densely-deployed, low-latency edge resources in urban environments, coping with the bootstrapping problem and enabling (novel) distributed AI services.

A Unique Proactive Deployment Approach Based on User Mobility for Just-in-time Initialization of Device-Bound AI Services

Last but not least, the *seventh* contribution proposed a way to consolidate data decentralization with decentralized urban infrastructures (as the above proposed); it added a proactive deployment mechanism (coined `PDSProxy++`) for device-bound AI services based on user mobility (see Chapter 8). A series of experiments showed that `PDSProxy++` can initialize personalized yet data-protected AI services on nearby (IoT/edge) devices just in time, shortly before mobile users would use them – this way, the inherent initialization overhead was efficiently concealed from the users, and thus, the user experience was increased.

9.2 INTEGRATION AND FUTURE WORK

This thesis provides the foundation for future research on data protection in AI services. Specifically, it contributed to the vision of data decentralization and 'true' data ownership when using especially personalized (single-user) AI services. Even though we have taken a substantial step forward, interesting new open challenges have arisen – research may address them in the future.

Further Improvement on the Personalization–Privacy Tradeoff

While this thesis already achieved a unique tradeoff between personalization of AI services and user privacy (and the applicability of the protection mechanisms), users and/or providers still have to accept a certain compromise. Therefore, future research shall further improve this tradeoff, which both parties must agree to. For example, unless a much more efficient (mobile) hardware becomes ubiquitous, the inherent initialization overhead of the `PDSProxy` concept or, more precisely, the underlying protocol for enclave initialization should be addressed in future research: a potential for improvement would offer a faster method to remote attestation, e.g., through the use of hierarchically-organized local delegates. Above all, interdisciplinary research is most important to overcome

the drawbacks of 'isolated' approaches – as this thesis showed, and what will become clearer with the next challenges as well.

Support for Confidential Interaction with AI Services

This work focused only on data protection during the storing and processing of user data. This means this thesis cannot make any data protection guarantees *before* the data is obtained from the sensors and *after* the data is given to the actuators – which in essence represents the interaction with the user. Therefore, integration into appropriate HCI concepts is required to allow personalized AI services to interact with the user in confidence – a comprehensive survey is given with [DTS19]. Especially in multi-user environments, future HCI research shall explore (interaction) concepts to convey personalized (ambient) support only to the specific user in a way that still keeps the user's private information confidential. Promising directions include Parallel Reality Displays or see-through AR-HMDs. The former allows multiple users to share a digital screen or sign at the same time while everyone sees something different (depending on the viewing angle) [DL19]; the latter enables stereoscopic output through semi-transparent displays (one per eye), what can only be seen by the wearer – but they are limited to digital/virtual support and require wearing suitable glasses. For the capturing part, research on (interdependent) privacy shall explore and integrate concepts to collect only the context or data of the target user and not those of others and, if necessary, only in an anonymous way – a comprehensive survey is given with [HTH19].

Generalization and Transfer of the Proposed Concepts

This thesis took care to design the concepts as general as possible but as AI service specific as necessary; only `PNet` is very limited, namely to supervised ML algorithms. Therefore, most (or only parts) of the proposed concepts are *transferrable* to other (sub-)fields. For instance, future research shall transfer and extend the proposed data protection mechanisms at system level (i.e., the `PDSProxy` underlying mechanisms) to the field of edge computing to meet the three-party confidentiality challenge there as well – which is an often neglected issue in this field and for urban services [Meu21a]. Future research may also explore the community-based approach of `PNet` in the context of federated learning: it can help to determine which users are 'similar' (communities) and thus suitable for forming a shared model that new users who can be assigned to the corresponding community can use directly. Last but not least, future research shall investigate how to apply the data decentralization paradigm not only to single-user AI services but also to multi-user AI services – again, the interdependent privacy challenge must be addressed in particular [HTH19].

Novel Business Models for Decentralized AI Services

This thesis contributed various technical concepts to prevent data decentralization from remaining a showstopper for business (referring, for example, to the IP protection of providers' AI algorithms/models – see also [Meu21b; Meu21a]). A very important last point, however, concerns suitable (innovative) business models for such decentralized, data-protected AI services and infrastructure concepts proposed. Although some possible ways of doing this were discussed at the end of some chapters, it is principally outside the scope of this thesis.

Besides raising broad user awareness of data privacy, which is a societal and political task, further research in this area is necessary to convince (service and infrastructure) providers and to make the transition of such data decentralizing concepts to the consumer market at all. This might result in providers losing their exclusive access to user data, but on the other hand they could again achieve a unique selling point for their AI services with innovative business models. In particular, studies such as [MZH17; MZH19] that investigate the willingness of users to pay for various AI services can be a first step – from which, among others, business models tailored to the user type can be derived.

9.3 CONCLUDING REMARKS

The recent data scandals have ultimately taught us that today's web-based services provide insufficient data protection. Even the inventor of the WWW, Tim Berners-Lee, criticizes this evolution and sees the main reasons for this in the (centralized, data-collecting) way they work today. With data decentralization, of which he (same as the author of this thesis) is a strong proponent, a new vision appeared to enable a technical path (without central data silos) towards 'true' data ownership and unconditional user privacy when using such services.

Certainly, this thesis cannot provide a conclusive picture of how these services or, more specifically, AI services will be operated in the future; it also cannot resolve the inherent conflict of personalized AI services between privacy, personalization, and applicability. However, this thesis has made a substantial contribution to a more favorable tradeoff for users and towards the vision for data decentralization in the field of emerging (data-demanding, single-user) AI services by synergistically combining data protection mechanisms affecting different levels. In addition, this thesis showed up one possible way to solve the classic (application–infrastructure) bootstrapping problem for decentralized (device-bound, data-protected) AI services and to make human environments – according to Weiser's vision [Wei91] – more responsive, supportive, and above all confidential.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACRONYMS

AI	Artificial Intelligence
AP	Access Point
API	Application Programming Interface
AR	Augmented Reality
ASL	Augmented Street Lamp
BC	Blockchain
CA	Certificate Authority
CCPA	The California Consumer Privacy Act
CDF	Cumulative Distribution Function
CPU	Central Processing Unit
DC-PDS	Data-confining Personal Data Store
DFT	Discrete Fourier transform
DSCF	Dwass-Steel-Critchlow-Fligner
EI	Edge Intelligence
EU	European Union
FOAF	Friend of a Friend
GDPR	General Data Protection Regulation
GM	General Model
GPS	Global Positioning System
GPU	Graphics Processing Unit
HCI	Human-Computer Interaction
HE	Homomorphic Encryption
HMD	Head-mounted Display
HW	Hardware
IA	Intelligence Augmentation
IC	Identity Certificate
ICT	Information and Communications Technology
II	Intelligent Infrastructures
I/O	Input/Output
IoT	Internet of Things
IP	Intellectual Property
IPO	Input–Processing–Output
IQR	Interquartile Range

LAN	Local Area Network
LSH	Location-sensitive Hashing
MAC	Media Access Control
MANET	Mobile Ad-hoc Network
ML	Machine Learning
NAS	Network Attached Storage
OS	Operating System
OUI	Organizationally Unique Identifier
PDF	Probability Density Function
PDS	Personal Data Store
PGP	Pretty Good Privacy
PoLP	Principle of Least Privilege
PKI	Public Key Infrastructure
PM-LR	Personal Model (local-retrained)
PM-LI	Personal Model (local-incremental)
RA	Root Authority
RaaP	Routers as a Platform
RSA	Rivest–Shamir–Adleman
RTT	Round Trip Time
SBC	Single-board computer
SGX	Software Guard Extensions
SLaaP	Street Lamps as a Platform
SMC	Secure Multi-party Computation
TEE	Trusted Execution Environment
UAV	Unmanned Aerial Vehicle
VANET	Vehicular Ad-hoc Network
VM	Virtual Machine
VR	Virtual Reality
WHO	World Health Organization
WMAN	Wireless Metropolitan Area Network
WOT	Web of Trust
WWW	World Wide Web

BIBLIOGRAPHY

[Aba+16] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. "Deep Learning With Differential Privacy." In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. CCS'16. ACM, 2016, pp. 308–318. DOI: 10.1145/2976749.2978318.

[Abb+17] Nasir Abbas, Yan Zhang, Amir Taherkordi, and Tor Skeie. "Mobile Edge Computing: A Survey." In: *IEEE Internet of Things Journal* 5.1 (2017), pp. 450–465. DOI: 10.1109/JIOT.2017.2750180.

[Abe+16] Tigist Abera, N Asokan, Lucas Davi, Farinaz Koushanfar, Andrew Paverd, Ahmad-Reza Sadeghi, and Gene Tsudik. "Things, Trouble, Trust: On Building Trust in IoT Systems." In: *Proceedings of the 53rd Annual Design Automation Conference*. DAC'16. ACM, 2016, pp. 121–126. DOI: 10.1145/2897937.2905020.

[AEY10] Annie I Antón, Julia B Earp, and Jessica D Young. "How Internet Users' Privacy Concerns Have Evolved Since 2002." In: *IEEE Security & Privacy* 8.1 (Feb. 2010), pp. 21–27. DOI: 10.1109/MSP.2010.38.

[AF18] Islam Tharwat Abdel-Halim and Hossam Mahmoud Ahmed Fahmy. "Prediction-based Protocols for Vehicular ad hoc Networks: Survey and Taxonomy." In: *Computer Networks* 130 (Jan. 2018), pp. 34–50. DOI: 10.1016/j.comnet.2017.10.009.

[Agg+05] Gagan Aggarwal, Tomas Feder, Krishnaram Kenthapadi, Rajeev Motwani, Rina Panigrahy, Dilys Thomas, and An Zhu. "Approximation algorithms for k-anonymity." In: *Journal of Privacy Technology (JOPT)* (2005). URL: http://ilpubs.stanford.edu:8090/645/.

[AI19] Google AI. *Federated Learning*. 2019. URL: https://federated.withgoogle.com.

[AIM10] Luigi Atzori, Antonio Iera, and Giacomo Morabito. "The Internet of Things: A Survey." In: *Computer Networks* 54.15 (Oct. 2010), pp. 2787–2805. DOI: 10.1016/j.comnet.2010.05.010.

[Ala+16] Mahmoud Abdulwahab Alawi, Raed A Alsaqour, Elankovan Sundararajan, and Mahamod Ismail. "Prediction Model for Offloading in Vehicular Wi-Fi Network." In: *International Journal on Advanced Science, Engineering and Information Technology* 6.6 (Dec. 2016), pp. 944–951. DOI: 10.18517/ijaseit.6.6.1411.

[ALC12] Saeed Abdullah, Nicholas D Lane, and Tanzeem Choudhury. "Towards Population Scale Activity Recognition: A Framework for Handling Data Diversity." In: *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*. AAAI'12. AAAI Press, 2012, pp. 851–857. ISBN: 9781577355687. URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI12/paper/view/5169.

[Ale19] Abdulhameed Alelaiwi. "An Efficient Method of Computation Offloading in an Edge Cloud Platform." In: *Journal of Parallel and Distributed Computing* 127 (2019), pp. 58–64. DOI: 10.1016/j.jpdc.2019.01.003.

[AM12] Eiman Alotaibi and Biswanath Mukherjee. "A Survey on Routing Algorithms for Wireless Ad-hoc and Mesh Networks." In: *Computer networks* 56.2 (Feb. 2012), pp. 940–965. DOI: 10.1016/j.comnet.2011.10.011.

[An+19] JongGwan An, Wenbin Li, Franck Le Gall, Ernoe Kovac, Jaeho Kim, Tarik Taleb, and JaeSeung Song. "EiF: Toward an Elastic IoT Fog Framework for AI Services." In: *IEEE Communications Magazine* 57.5 (May 2019), pp. 28–33. DOI: 10.1109/MCOM.2019.1800215.

[AP08] Charu C Aggarwal and S Yu Philip. "A General Survey of Privacy-preserving Data Mining Models and Algorithms." In: *Privacy-preserving Data Mining*. Springer, 2008, pp. 11–52. DOI: 0.1007/978-0-387-70992-5_2.

[Arn+16] Sergei Arnautov et al. "SCONE: Secure Linux Containers With Intel SGX." In: *12th USENIX Symposium on Operating Systems Design and Implementation*. OSDI'16. USENIX Association, 2016, pp. 689–703. URL: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/arnautov.

[AS00] Rakesh Agrawal and Ramakrishnan Srikant. "Privacy-preserving Data Mining." In: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. SIGMOD'00 2. ACM, 2000, pp. 439–450. DOI: 10.1145/342009.335438.

[Bar+11] Mauro Barni, Pierluigi Failla, Riccardo Lazzeretti, Ahmad-Reza Sadeghi, and Thomas Schneider. "Privacy-Preserving eCG Classification With Branching Programs and Neural Networks." In: *IEEE Transactions on Information Forensics and Security* 6.2 (June 2011), pp. 452–468. DOI: 10.1109/TIFS.2011.2108650.

[Bar+20] Allan de Barcelos Silva, Marcio Miguel Gomes, Cristiano André da Costa, Rodrigo da Rosa Righi, Jorge Luis Victoria Barbosa, Gustavo Pessin, Geert De Doncker, and Gustavo Federizzi. "Intelligent Personal Assistants: A Systematic Literature Review." In: *Expert Systems with Applications* 147 (June 2020), p. 113193. DOI: 10.1016/j.eswa.2020.113193.

[Baş+15] Ejder Baştuğ, Mehdi Bennis, Engin Zeydan, Manhal Abdel Kader, Ilyas Alper Karatepe, Ahmet Salih Er, and Mérouane Debbah. "Big Data Meets Telcos: A Proactive Caching Perspective." In: *Journal of Communications and Networks* 17.6 (Dec. 2015), pp. 549–557. DOI: 10.1109/JCN.2015.000102.

[Bas+98] Stefano Basagni, Imrich Chlamtac, Violet R Syrotiuk, and Barry A Woodward. "A Distance Routing Effect Algorithm for Mobility (DREAM)." In: *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking*. MobiCom'98. ACM, 1998, pp. 76–84. DOI: 10.1145/288235.288254.

[Bay+20] Sebastian P Bayerl, Tommaso Frassetto, Patrick Jauernig, Korbinian Riedhammer, Ahmad-Reza Sadeghi, Thomas Schneider, Emmanuel Stapf, and Christian Weinert. "Offline Model Guard: Secure and Private ML on Mobile Devices." In: *23. Design, Automation and Test in Europe Conference*. DATE'20. IEEE, 2020, pp. 460–465. DOI: 10.23919/DATE48585.2020.9116560.

[BB08] Enrico Blanzieri and Anton Bryl. "A Survey of Learning-based Techniques of Email Spam Filtering." In: *Artificial Intelligence Review* 29.1 (July 2008), pp. 63–92. DOI: 10.1007/s10462-009-9109-6.

[BC11] Dario Bonino and Fulvio Corno. "What Would you ask to Your Home if it Were Intelligent? Exploring User Expectations About Next-generation Homes." In: *Journal of Ambient Intelligence and Smart Environments* 3.2 (2011), pp. 111–126. DOI: 10.3233/AIS-2011-0099.

[BD03] Louise Barkhuus and Anind K Dey. "Location-based Services for Mobile Telephony: A Study of Users' Privacy Concerns." In: *Proceedings of the 9th IFIP TC13 International Conference on Human-Computer Interaction*. Vol. 3. INTERACT'03. IOS Press, 2003, pp. 702–712.

[BD17] Susanne Barth and Menno DT De Jong. "The Privacy Paradox–Investigating Discrepancies Between Expressed Privacy Concerns and Actual Online Behavior–A Systematic Literature Review." In: *Elsevier Telematics and Informatics* 34.7 (Nov. 2017), pp. 1038–1058. DOI: 10.1016/j.tele.2017.04.013.

[Ben+18] Frank Bentley, Chris Luvogt, Max Silverman, Rushani Wirasinghe, Brooke White, and Danielle Lottrjdge. "Understanding the Long-Term use of Smart Speaker Assistants." In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2.3 (Sept. 2018), 91:1–91:24. DOI: 10.1145/3264901.

[Ber17] Tim Berners-Lee. *Three Challenges for the web, According to its Inventor*. World Wide Web Foundation. 2017. URL: https://webfoundation.org/2017/03/web-turns-28-letter.

[BF81] Avron Barr and Edward A Feigenbaum. *The Handbook of Artificial Intelligence*. Butterworth-Heinemann, 1981. ISBN: 978-0-86576-089-9. DOI: 10.1016/C2013-0-07690-6.

[BFK17] Niels Van Berkel, Denzil Ferreira, and Vassilis Kostakos. "The Experience Sampling Method on Mobile Devices." In: *ACM Computing Surveys* 50.6 (Dec. 2017), 93:1–93:40. DOI: 10.1145/3123988.

[BGK11] Elisa Bertino, Gabriel Ghinita, and Ashish Kamra. "Access Control for Databases: Concepts and Systems." In: *Foundations and Trends® in Databases* 3.1–2 (2011), pp. 1–148. DOI: 10.1561/1900000014.

[Blu+05] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. "Practical Privacy: The SuLQ Framework." In: *Proceedings of the Twenty-fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. PODS'05. ACM, 2005, pp. 128–138. DOI: 10.1145/1065167.1065184.

[Bon+17] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. "Practical Secure Aggregation for Privacy-Preserving Machine Learning." In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. CCS'17. ACM, 2017, pp. 1175–1191. DOI: 10.1145/3133956.3133982.

[Bou+18] Sol Boucher, Anuj Kalia, David G Andersen, and Michael Kaminsky. "Putting the "Micro" Back in Microservice." In: *Proceedings of the 2018 USENIX Annual Technical Conference*. ATC'18. USENIX Association, 2018, pp. 645–650. ISBN: 978-1-939133-01-4. URL: https://www.usenix.org/conference/atc18/presentation/boucher.

[BPH14] Andrew Baumann, Marcus Peinado, and Galen Hunt. "Shielding Applications From an Untrusted Cloud With Haven." In: *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation*. OSDI'14. USENIX Association, 2014, pp. 267–283. ISBN: 9781931971164. URL: https://www.usenix.org/conference/osdi14/technical-sessions/presentation/baumann.

[Bra+18] Ferdinand Brasser, Tommaso Frassetto, Korbinian Riedhammer, Ahmad-Reza Sadeghi, Thomas Schneider, and Christian Weinert. "VoiceGuard: Secure and Private Speech Processing." In: *Proc. Interspeech 2018*. ISCA. 2018, pp. 1303–1307. DOI: 10.21437/Interspeech.2018-2032.

[Bra+19] Ferdinand Brasser, Srdjan Capkun, Alexandra Dmitrienko, Tommaso Frassetto, Kari Kostiainen, and Ahmad-Reza Sadeghi. "DR.SGX: Automated and Adjustable Side-Channel Protection for SGX Using Data Location Randomization." In: *Proceedings of the 35th Annual Computer Security Applications Conference*. ACSAC'19. ACM, 2019, pp. 788–800. DOI: 10.1145/3359789.3359809.

[Bre+14] Claude Breining et al. *Orchestrating Infrastructure for Sustainable Smart Cities (White Paper)*. Tech. rep. https://www.ceps.eu/wp-content/uploads/2015/01/iecWP-smartcities-LR-en.pdf. IEC, 2014, pp. 1–62.

[Bri18] Thomas M Brill. "Siri, Alexa, and Other Digital Assistants: A Study of Customer Satisfaction With Artificial Intelligence Applications." PhD thesis. University of Dallas, 2018.

[BSR18] Dries H Bostyn, Sybren Sevenhant, and Arne Roets. "Of mice, men, and trolleys: Hypothetical judgment versus real-life behavior in trolley-style moral dilemmas." In: *Psychological science* 29.7 (2018), pp. 1084–1093. DOI: 10.1177/0956797617752640.

[Bui+17] Nicola Bui, Matteo Cesana, S Amir Hosseini, Qi Liao, Ilaria Malanchini, and Joerg Widmer. "A Survey of Anticipatory Mobile Networking: Context-based Classification, Prediction Methodologies, and Optimization Techniques." In: *IEEE Communications Surveys & Tutorials* 19.3 (Apr. 2017), pp. 1790–1821. DOI: 10.1109/COMST.2017.2694140.

[Cai+19] Yang Cai, Angelo Genovese, Vincenzo Piuri, Fabio Scotti, and Mel Siegel. "IoT-based Architectures for Sensing and Local Data Processing in Ambient Intelligence: Research and Industrial Trends." In: *Proceedings of the 2019 IEEE International Instrumentation and Measurement Technology Conference*. I2MTC'19. IEEE. 2019, pp. 1–6. DOI: 10.1109/I2MTC.2019.8827110.

[Cap+19] Maurizio Capra, Riccardo Peloso, Guido Masera, Massimo Ruo Roch, and Maurizio Martina. "Edge computing: A survey on the hardware requirements in the internet of things world." In: *Future Internet* 11.4 (2019), p. 100. DOI: 10.3390/fi11040100.

[CD16] Victor Costan and Srinivas Devadas. *Intel SGX Explained*. IACR Cryptology Archive. http://css.csail.mit.edu/6.858/2020/readings/costan-sgx.pdf. 2016.

[CDA14] John Criswell, Nathan Dautenhahn, and Vikram Adve. "Virtual Ghost: Protecting Applications From Hostile Operating Systems." In: *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*. ASPLOS'14. ACM, 2014, pp. 81–96. DOI: 10.1145/2541940.2541986.

[CF10] Henry Corrigan-Gibbs and Bryan Ford. "Dissent: Accountable Anonymous Group Messaging." In: *Proceedings of the 17th ACM Conference on Computer and Communications Security*. CCS'10. 2010, pp. 340–350. DOI: 10.1145/1866307.1866346.

[Cha+12] Jason Chambers, Theresa Robison, Dameion Dorsner, Sridhar Manickam, and Daniel Konisky. *Certificate-based Mutual Authentication for Data Security*. US Patent App. 13/527,867. 2012.

[Cha+15] Amir Chaudhry, Jon Crowcroft, Heidi Howard, Anil Madhavapeddy, Richard Mortier, Hamed Haddadi, and Derek McAuley. "Personal Data: Thinking Inside the Box." In: *Proceedings of the Fifth Decennial Aarhus Conference on Critical Alternatives*. AA'15. ACM, 2015, pp. 29–32. DOI: 10.7146/aahcc.v1i1.21312.

[Cha+17] Swarup Chandra, Vishal Karande, Zhiqiang Lin, Latifur Khan, Murat Kantarcioglu, and Bhavani Thuraisingham. "Securing Data Analytics on SGX With Randomization." In: *Proceedings of the European Symposium on Research in Computer Security*. ESORICS'17. Springer. 2017, pp. 352–369. DOI: 10.1007/978-3-319-66402-6_21.

[Che+17] Siyun Chen, Ting Liu, Feng Gao, Jianting Ji, Zhanbo Xu, Buyue Qian, Hongyu Wu, and Xiaohong Guan. "Butler, not Servant: A Human-Centric Smart Home Energy Management System." In: *IEEE Communications Magazine* 55.2 (Feb. 2017), pp. 27–33. DOI: 10.1109/MCOM.2017.1600699CM.

[Che+19] Raymond Cheng, Fan Zhang, Jernej Kos, Warren He, Nicholas Hynes, Noah Johnson, Ari Juels, Andrew Miller, and Dawn Song. "Ekiden: A Platform for Confidentiality-preserving, Trustworthy, and Performant Smart Contract Execution." In: *Proceedings of the 2019 IEEE European Symposium on Security and Privacy*. EuroS&P. IEEE. 2019, pp. 185–200. DOI: 10.1109/EuroSP.2019.00023.

[Cho+18] Mohammad Jabed Morshed Chowdhury, Alan Colman, Muhammad Ashad Kabir, Jun Han, and Paul Sarda. "Blockchain as a Notarization Service for Data Sharing With Personal Data Store." In: *Proceedings of the 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering*. TrustCom/BigDataSE'18. IEEE. 2018, pp. 1330–1335. DOI: 10.1109/TrustCom/BigDataSE.2018.00183.

[CK17] Minho Choi and Sang Woo Kim. "Online SVM-based Personalizing Method for the Drowsiness Detection of Drivers." In: *Proceedings of the 2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. EMBC'17. IEEE. 2017, pp. 4195–4198. DOI: 10.1109/EMBC.2017.8037781.

[CM15] Luca Canzian and Mirco Musolesi. "Trajectories of Depression: Unobtrusive Monitoring of Depressive States by Means of Smartphone Mobility Traces Analysis." In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. UbiComp'15. ACM, 2015, pp. 1293–1304. DOI: 10.1145/2750858.2805845.

[CM16] Riccardo Coppola and Maurizio Morisio. "Connected car: Technologies, Issues, Future Trends." In: *ACM Computing Surveys* 49.3 (Oct. 2016), pp. 1–36. DOI: 10.1145/2971482.

[Coh88] Jacob Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Vol. 2. 1. Lawrence Erlbaum Associates New Jersey, NJ, USA, 1988, pp. 284–288. ISBN: 0-8058-0283-5.

[Com02] C/LM – LAN/MAN Standards Committee. *IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture*. IEEE Standards Association. https://standards.ieee.org/standard/802-2001.html. 2002.

[Cor18] Zoë Corbyn. *Decentralisation: The Next Big Step for the World Wide Web*. The Observer (The Guardian). 2018. URL: https://www.theguardian.com/technology/2018/sep/08/decentralisation-next-big-step-for-the-world-wide-web-dweb-data-internet-censorship-brewster-kahle.

[Cow+17] Benjamin R Cowan, Nadia Pantidi, David Coyle, Kellie Morrissey, Peter Clarke, Sara Al-Shehri, David Earley, and Natasha Bandeira. "What can I Help you With?: Infrequent Users' Experiences of Intelligent Personal Assistants." In: *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI'17. ACM, 2017, 43:1–43:12. DOI: 10.1145/3098279.3098539.

[Cra+16] Andy Crabtree, Tom Lodge, James Colley, Chris Greenhalgh, Richard Mortier, and Hamed Haddadi. "Enabling the New Economic Actor: Data Protection, the Digital Economy, and the Databox." In: *Personal and Ubiquitous Computing* 20.6 (Aug. 2016), pp. 947–957. DOI: 10.1007/s00779-016-0939-3.

[CRM09] Barak Chizi, Lior Rokach, and Oded Maimon. "A survey of feature selection techniques." In: *Encyclopedia of Data Warehousing and Mining, Second Edition*. IGI Global, 2009, pp. 1888–1895. DOI: 10.4018/978-1-60566-010-3.ch289.

[CS05] Ramnath K Chellappa and Raymond G Sin. "Personalization Versus Privacy: An Empirical Examination of the Online Consumer's Dilemma." In: *Information Technology and Management* 6.2-3 (Apr. 2005), pp. 181–202. DOI: 10.1007/s10799-005-5879-y.

[CT65] JW Cooley and JW Tukey. "An Algorithm for the Machine Calculation of Complex Fourier Series, vol. 19." In: *Mathematics of Computation* 19 (1965), pp. 297–301. DOI: 10.1090/S0025-5718-1965-0178586-1.

[Cun+09] Vincenzo D Cunsolo, Salvatore Distefano, Antonio Puliafito, and Marco Scarpa. "Cloud@ home: Bridging the gap between volunteer and cloud computing." In: *International Conference on Intelligent Computing*. Springer. 2009, pp. 423–432. DOI: 10.1007/978-3-642-04070-2_48.

[Dav+16] Nigel Davies, Nina Taft, Mahadev Satyanarayanan, Sarah Clinch, and Brandon Amos. "Privacy Mediators: Helping IoT Cross the Chasm." In: *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications*. HotMobile'16. 2016, pp. 39–44. DOI: 10.1145/2873587.2873600.

[Dem06] Janez Demšar. "Statistical Comparisons of Classifiers Over Multiple Data Sets." In: *Journal of Machine Learning Research* 7.1 (Jan. 2006), pp. 1–30. URL: http://jmlr.org/papers/v7/demsar06a.html.

[Den18] Li Deng. "Artificial intelligence in the rising wave of deep learning: The historical path and future outlook [perspectives]." In: *IEEE Signal Processing Magazine* 35.1 (2018), pp. 180–177. DOI: 10.1109/MSP.2017.2762725.

[DH08] George Demiris and Brian K Hensel. "Technologies for an Aging Society: A Systematic Review of "Smart Home" Applications." In: *Yearbook of medical informatics* 17.01 (Jan. 2008), pp. 33–40. DOI: 10.1055/s-0038-1638580.

[DHA18] Mateusz Dubiel, Martin Halvey, and Leif Azzopardi. *A Survey Investigating Usage of Virtual Personal Assistants*. 2018. arXiv: 1807.04606 [cs.HC].

[DL19] Paul H. Dietz and Matt Lathrop. "Adaptive Environments With Parallel Reality Displays." In: *ACM SIGGRAPH 2019 Talks*. SIGGRAPH'19. ACM, 2019, 34:1–34:2. DOI: 10.1145/3306307.3328153.

[Dow+16] Nathan Dowlin, Ran Gilad-Bachrach, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. "CryptoNets: Applying Neural Networks to Encrypted Data With High Throughput and Accuracy." In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. ICML'16. JMLR: W&CP, 2016, pp. 201–210. URL: https://dl.acm.org/doi/abs/10.5555/3045390.3045413.

[DTS19] Jaybie A De Guzman, Kanchana Thilakarathna, and Aruna Seneviratne. "Security and Privacy Approaches in Mixed Reality: A Literature Survey." In: *ACM Computing Surveys*. CSUR 52.6 (Oct. 2019), 110:1–110:37. DOI: 10.1145/3359626.

[Dwo+06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. "Calibrating Noise to Sensitivity in Private Data Analysis." In: *Proceedings of the Theory of Cryptography Conference*. TCC'06. Springer. 2006, pp. 265–284. DOI: 10.1007/11681878_1.

[Dwo06] Cynthia Dwork. "Differential Privacy." In: *Proceedings of the 33rd International Conference on Automata, Languages and Programming - Volume Part II*. ICALP'06. Springer, 2006, pp. 1–12. DOI: 10.1007/11787006_1.

[EJ86] Charles W Eriksen and James D St James. "Visual Attention Within and Around the Field of Focal Attention: A Zoom Lens Model." In: *Perception & Psychophysics* 40.4 (July 1986), pp. 225–240. DOI: 10.3758/BF03211502.

[EN20] Tarek Elgamal and Klara Nahrstedt. *Serdab: An IoT Framework for Partitioning Neural Networks Computation Across Multiple Enclaves*. 2020. arXiv: 2005.06043 [cs.DC].

[EP06] Nathan Eagle and Alex Sandy Pentland. "Reality Mining: Sensing Complex Social Systems." In: *Personal and ubiquitous computing* 10.4 (May 2006), pp. 255–268. DOI: 10.1007/s00779-005-0046-3.

[EPK14] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. "RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response." In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. CCS'14. ACM, 2014, pp. 1054–1067. DOI: 10.1145/2660267.2660348.

[Erd+17] Milan Erdelj, Enrico Natalizio, Kaushik R Chowdhury, and Ian F Akyildiz. "Help From the sky: Leveraging UAVs for Disaster Management." In: *IEEE Pervasive Computing* 16.1 (Jan. 2017), pp. 24–32. DOI: 10.1109/MPRV.2017.11.

[Est+96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases With Noise." In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. KDD'96 34. AAAI Press, 1996, pp. 226–231. URL: https://dl.acm.org/doi/10.5555/3001460.3001507.

[Fel+12] Oriel FeldmanHall, Dean Mobbs, Davy Evans, Lucy Hiscox, Lauren Navrady, and Tim Dalgleish. "What we say and what we do: The relationship between real and hypothetical moral choices." In: *Cognition* 123.3 (2012), pp. 434–441. DOI: 10.1016/j.cognition.2012.02.001.

[Fer+16] Earlence Fernandes, Justin Paupore, Amir Rahmati, Daniel Simionato, Mauro Conti, and Atul Prakash. "FlowFence: Practical Data Protection for Emerging IoT Application Frameworks." In: *Proceedings of the 25th USENIX Security Symposium*. Security'16. USENIX Association, 2016, pp. 531–548. URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/fernandes.

[FHR13] Katsuya Fujii, Keita Higuchi, and Jun Rekimoto. "Endless Flyer: A Continuous Flying Drone With Automatic Battery Replacement." In: *Proceedings of the 2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing*. UIC/ATC'13. IEEE. 2013, pp. 216–223. DOI: 10.1109/UIC-ATC.2013.103.

[FJ15] Liyue Fan and Hongxia Jin. "A Practical Framework for Privacy-preserving Data Analytics." In: *Proceedings of the 24th International Conference on World Wide Web*. WWW'15. ACM, 2015, pp. 311–321. DOI: 10.1145/2736277.2741122.

[FKM21] Natasha Fernandes, Yusuke Kawamoto, and Takao Murakami. *Locality Sensitive Hashing with Extended Differential Privacy*. 2021. arXiv: 2010.09393.

[Fog99] Brian J Fogg. "Persuasive Technologies." In: *Communications of the ACM* 42.5 (May 1999), pp. 26–29. DOI: 10.1145/301353.301396.

[Fon+15] Ramon R Fontes, Samira Afzal, Samuel HB Brito, Mateus AS Santos, and Christian Esteve Rothenberg. "Mininet-WiFi: Emulating Software-defined Wireless Networks." In: *Proceedings of the 2015 11th International Conference on Network and Service Management*. CNSM'15. IEEE. 2015, pp. 384–389. DOI: 10.1109/CNSM.2015.7367387.

[Fri+14] Jesús Friginal, David de Andrés, Juan-Carlos Ruiz, and Miquel Martínez. "A Survey of Evaluation Platforms for ad hoc Routing Protocols: A Resilience Perspective." In: *Computer Networks* 75.Part A (Dec. 2014), pp. 395–413. DOI: 10.1016/j.comnet.2014.09.010.

[Gam95] Erich Gamma. *Design Patterns: Elements of Reusable Object-oriented Software*. Pearson Education India, 1995. ISBN: 978-0201633610.

[Gar95] Simson Garfinkel. *PGP: Pretty Good Privacy*. O'Reilly Media, Inc., 1995.

[Gas18] Mila Gascó-Hernandez. "Building a Smart City: Lessons From Barcelona." In: *Communications of the ACM* 61.4 (Apr. 2018), pp. 50–57. DOI: 10.1145/3117800.

[Gau+20] Laetitia Gauvin, Michele Tizzoni, Simone Piaggesi, Andrew Young, Natalia Adler, Stefaan Verhulst, Leo Ferres, and Ciro Cattuto. "Gender gaps in urban mobility." In: *Humanities and Social Sciences Communications* 7.1 (2020), pp. 1–13. DOI: 10.1057/s41599-020-0500-x.

[GCB07] Mike Graves, Adam Constabaris, and Dan Brickley. "FOAF: Connecting People on the Semantic Web." In: *Cataloging & Classification Quarterly* 43.3-4 (2007), pp. 191–202. DOI: 10.1300/J104v43n03_10.

[GDM18] Tim Grube, Jorg Daubert, and Max Muhlhauser. "Asymmetric DC-nets for Effective and Efficient Sender Anonymity." In: *2018 IEEE Global Communications Conference*. GLOBECOM'18. IEEE. 2018, pp. 1–7. DOI: 10.1109/GLOCOM.2018.8647607.

[Gen09] Craig Gentry. *A Fully Homomorphic Encryption Scheme*. Stanford University, 2009. ISBN: 978-1-109-44450-6.

[Gen11] Suiyan Geng. "Millimeter Wave and UWB Propagation for High Throughput Indoor Communications." Doctoral thesis. 2011. URL: http://urn.fi/URN:ISBN:978-952-60-4318-0.

[GGV18] Nina Gerber, Paul Gerber, and Melanie Volkamer. "Explaining the Privacy Paradox: A Systematic Review of Literature Investigating Privacy Attitude and Behavior." In: *Computers & Security* 77 (Aug. 2018), pp. 226–261. DOI: 10.1016/j.cose.2018.04.002.

[GJ19] Jonathan Grudin and Richard Jacques. "Chatbots, Humbots, and the Quest for Artificial General Intelligence." In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI'19. ACM, 2019, 209:1–209:11. DOI: 10.1145/3290605.3300439.

[GL08] Bugra Gedik and Ling Liu. "Protecting Location Privacy With Personalized K-anonymity: Architecture and Algorithms." In: *IEEE Transactions on Mobile Computing* 7.1 (Jan. 2008), pp. 1–18. DOI: 10.1109/TMC.2007.1062.

[GLN12] Thore Graepel, Kristin Lauter, and Michael Naehrig. "ML Confidential: Machine Learning on Encrypted Data." In: *Proceedings of the 15th International Conference on Information Security and Cryptology*. ICISC'12. Springer. 2012, pp. 1–21. DOI: 10.1007/978-3-642-37682-5_1.

[Goy+06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data." In: *Proceedings of the 13th ACM conference on Computer and communications security*. CCS'06. ACM, 2006, pp. 89–98. DOI: 10.1145/1180405.1180418.

[GS15] Julien Gedeon and Immanuel Schweizer. "Understanding Spatial and Temporal Coverage in Participatory Sensor Networks." In: *Proceedings of the 2015 IEEE 40th Local Computer Networks Conference Workshops (LCN Workshops)*. IEEE. 2015, pp. 699–707. DOI: 10.1109/LCNW.2015.7365917.

[Hal18] Hanna Halaburda. "Blockchain Revolution Without the Blockchain?" In: *Communications of the ACM* 61.7 (July 2018), pp. 27–29. DOI: 10.1145/3225619.

[Har+19] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. *Federated Learning for Mobile Keyboard Prediction*. 2019. arXiv: 1811.03604 [cs.CL].

[HD00] John A Hoxmeier and Chris DiCesare. "System Response Time and User Satisfaction: An Experimental Study of Browser-based Applications." In: *AMCIS 2000 Proceedings*. AIS Electronic Library, 2000, pp. 140–145. URL: https://aisel.aisnet.org/amcis2000/347.

[HDC14] Thomas Hardjono, Patrick Deegan, and John Henry Clippinger. "Social use Cases for the iD3 Open Mustard Seed Platform." In: *IEEE Technology and Society Magazine* 33.3 (Sept. 2014), pp. 48–54. DOI: 10.1109/MTS.2014.2345197.

[Heu+17] Lutz Heuser et al. *Humble Lamppost DIN SPEC 91347: Integrated Smart Lighting Infrastructure*. Beuth Verlag GmbH, 2017. ISBN: 9783410268918.

[HK06] Rob J Hyndman and Anne B Koehler. "Another Look at Measures of Forecast Accuracy." In: *International journal of forecasting* 22.4 (Oct. 2006), pp. 679–688. DOI: 10.1016/j.ijforecast.2006.03.001.

[HLH15] Deng Huaqiu, Xiao Li, and Xue Huaqiang. "Wireless network node of LED street lamps." In: *2015 IEEE International Conference on Electro/Information Technology (EIT)*. IEEE. 2015, pp. 207–209. DOI: 10.1109/EIT.2015.7293341.

[HLR03] Qingfeng Huang, Chenyang Lu, and Gruia-Catalin Roman. "Mobicast: Just-in-time Multicast for Sensor Networks Under Spatiotemporal Constraints." In: *Information Processing in Sensor Networks*. IPSN'03. Springer. 2003, pp. 442–457. DOI: 10.1007/3-540-36978-3_30.

[Hof+13] Owen S. Hofmann, Sangman Kim, Alan M. Dunn, Michael Z. Lee, and Emmett Witchel. "InkTag: Secure Applications on an Untrusted Operating System." In: *Proceedings of the Eighteenth International Conference on Architectural Support for Programming Languages and Operating Systems*. Vol. 48. ASPLOS'13 4. ACM, 2013, pp. 265–278. DOI: 10.1145/2451116.2451146.

[Hol+11] Ralph Holz, Lothar Braun, Nils Kammenhuber, and Georg Carle. "The SSL Landscape: A Thorough Analysis of the X.509 PKI Using Active and Passive Measurements." In: *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*. IMC'11. ACM, 2011, pp. 427–444. DOI: 10.1145/2068816.2068856.

[Hon+21] Wei Hong, Zhi Hao Jiang, Chao Yu, Debin Hou, Haiming Wang, Chong Guo, Yun Hu, Le Kuai, Yingrui Yu, Zhengbo Jiang, et al. "The Role of Millimeter-Wave Technologies in 5G/6G Wireless Communications." In: *IEEE Journal of Microwaves* 1.1 (2021), pp. 101–122. DOI: 10.1109/JMW.2020.3035541.

[Hov17] Robert van den Hoven van Genderen. "Privacy and Data Protection in the Age of Pervasive Technologies in AI and Robotics." In: *Eur. Data Prot. L. Rev.* 3 (Jan. 2017), pp. 338–352. DOI: 10.21552/edpl/2017/3/8.

[Hoy14] Mathieu Hoyrup. "Irreversible Computable Functions." In: *31st Symposium on Theoretical Aspects of Computer Science*. STACS'14. Dagstuhl Publishing, 2014, pp. 1–17.

[HP19] Thomas Hardjono and Alex Pentland. *Data Cooperatives: Towards a Foundation for Decentralized Personal Data Management*. 2019. arXiv: 1905.08819 [cs.CY].

[HS15] Waldemar Hummer and Stefan Schulte. "Context-aware Personalization for Smart Mobile Cloud Services." In: *Proceedings of the International Conference on Service-Oriented Computing*. ICSOC'15 Workshops. Springer. 2015, pp. 171–183. DOI: 10.1007/978-3-662-50539-7_14.

[HSP19] Thomas Hardjono, David L Shrier, and Alex Pentland. *Trusted Data: A New Framework for Identity and Data Sharing*. MIT Connection Science & Engineering, 2019. ISBN: 9780262356053.

[HT14] Michael Henson and Stephen Taylor. "Memory Encryption: A Survey of Existing Techniques." In: *ACM Computing Surveys*. CSUR 46.4 (Mar. 2014), 53:1–53:26. DOI: 10.1145/2566673.

[HTH19] Mathias Humbert, Benjamin Trubert, and Kévin Huguenin. "A Survey on Interdependent Privacy." In: *ACM Computing Surveys*. CSUR 52.6 (Oct. 2019), 122:1–122:40. DOI: 10.1145/3360498.

[Hua+21] Huawei Huang, Wei Kong, Sicong Zhou, Zibin Zheng, and Song Guo. "A Survey of State-of-the-Art on Blockchains: Theories, Modelings, and Tools." In: 54.2 (2021), pp. 1–42. DOI: 10.1145/3441692.

[Hun+16] Tyler Hunt, Zhiting Zhu, Yuanzhong Xu, Simon Peter, and Emmett Witchel. "Ryoan: A Distributed Sandbox for Untrusted Computation on Secret Data." In: *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*. OSDI'16. USENIX Association, 2016, pp. 533–549. ISBN: 978-1-931971-33-1. URL: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/hunt.

[Hun+18] Tyler Hunt, Congzheng Song, Reza Shokri, Vitaly Shmatikov, and Emmett Witchel. *Chiron: Privacy-preserving Machine Learning as a Service*. 2018. arXiv: 1803.05961 [cs.CR].

[Hun90] AJ Hunt. *Land Navigation: Routefinding With map and Compass*. 1990.

[HV19] Cheol-Ho Hong and Blesson Varghese. "Resource management in fog/edge computing: a survey on architectures, infrastructure, and algorithms." In: *ACM Computing Surveys (CSUR)* 52.5 (2019), pp. 1–37. DOI: 10.1145/3326066.

[Ino+16] Sozo Inoue, Naonori Ueda, Yasunobu Nohara, and Naoki Nakashima. "Recognizing and Understanding Nursing Activities for a Whole day With a Big Dataset." In: *Journal of Information Processing* 24.6 (June 2016), pp. 853–866. DOI: 10.2197/ipsjjip.24.853.

[Isl+12] Sadeka Islam, Jacky Keung, Kevin Lee, and Anna Liu. "Empirical prediction models for adaptive resource provisioning in the cloud." In: *Future Generation Computer Systems* 28.1 (2012), pp. 155–162. DOI: 10.1016/j.future.2011.05.027.

[Jaf+17] Katia Jaffres-Runser, Gentian Jakllari, Tao Peng, and Vlad Nitu. "Crowdsensing mobile content and context data: Lessons learned in the wild." In: *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE. 2017, pp. 311–315. DOI: 10.1109/PERCOMW.2017.7917579.

[Jam+13] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning*. Vol. 112. Springer, 2013. DOI: 10.1007/978-1-4614-7138-7.

[JCL15] Mike Jia, Jiannong Cao, and Weifa Liang. "Optimal Cloudlet Placement and User to Cloudlet Allocation in Wireless Metropolitan Area Networks." In: *IEEE Transactions on Cloud Computing* 5.4 (Oct. 2015), pp. 725–737. DOI: 10.1109/TCC.2015.2449834.

[Ji+12] Jianqiu Ji, Jianmin Li, Shuicheng Yan, Bo Zhang, and Qi Tian. "Super-bit Locality-sensitive Hashing." In: *Advances in Neural Information Processing Systems*. Vol. 25. NIPS'12. Curran Associates, Inc., 2012, pp. 108–116. URL: https://proceedings.neurips.cc/paper/2012/file/072b030ba126b2f4b2374f342be9ed44-Paper.pdf.

[Jia+16] Mike Jia, Weifa Liang, Zichuan Xu, and Meitian Huang. "Cloudlet Load Balancing in Wireless Metropolitan Area Networks." In: *Proceedings of the 35th Annual IEEE International Conference on Computer Communications*. INFOCOM'16. IEEE. 2016, pp. 1–9. DOI: 10.1109/INFOCOM.2016.7524411.

[Jia+18] Gangyong Jia, Guangjie Han, Aohan Li, and Jiaxin Du. "SSL: Smart Street Lamp Based on Fog Computing for Smarter Cities." In: *IEEE Transactions on Industrial Informatics* 14.11 (Oct. 2018), pp. 4995–5004. DOI: 10.1109/TII.2018.2857918.

[JL07] Kipp Jones and Ling Liu. "What Where Wi: An Analysis of Millions of Wi-fi Access Points." In: *2007 IEEE International Conference on Portable Information Devices*. Portable'07. IEEE. 2007, pp. 1–4. DOI: 10.1109/PORTABLE.2007.45.

[JL11] Anil K Jain and Stan Z Li. *Handbook of Face Recognition*. Vol. 1. Springer, 2011. ISBN: 978-0-85729-932-1.

[JLE14] Zhanglong Ji, Zachary C. Lipton, and Charles Elkan. *Differential Privacy and Machine Learning: A Survey and Review*. 2014. arXiv: 1412.7584.

[Jor19] Michael Jordan. "Artificial Intelligence–The Revolution Hasn't Happened Yet." In: *Harvard Data Science Review* 1.1 (July 2019), pp. 1–9. DOI: 10.1162/99608f92.f06c6e61.

[JS07] Alejandro Jaimes and Nicu Sebe. "Multimodal Human–Computer Interaction: A Survey." In: *Computer Vision and Image Understanding* 108.1-2 (Oct. 2007), pp. 116–134. DOI: 10.1016/j.cviu.2006.10.019.

[JW18] Corey Brian Jackson and Yang Wang. "Addressing the Privacy Paradox Through Personalized Privacy Notifications." In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2.2 (July 2018), pp. 1–25. DOI: 10.1145/3214271.

[Kan+05] Jong Hee Kang, William Welbourne, Benjamin Stewart, and Gaetano Borriello. "Extracting Places From Traces of Locations." In: *ACM SIGMOBILE Mobile Computing and Communications Review* 9.3 (July 2005), pp. 58–68. DOI: 10.1145/1094549.1094558.

[Kan+17] Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor Mudge, Jason Mars, and Lingjia Tang. "Neurosurgeon: Collaborative Intelligence Between the Cloud and Mobile Edge." In: *ACM SIGARCH Computer Architecture News* 45.1 (Apr. 2017), pp. 615–629. DOI: 10.1145/3093337.3037698.

[Kan+18] Thivya Kandappu, Archan Misra, Shih-Fen Cheng, Randy Tandriansyah, and Hoong Chuin Lau. "Obfuscation At-source: Privacy in Context-aware Mobile Crowd-sourcing." In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2.1 (Mar. 2018), pp. 1–24. DOI: 10.1145/3191748.

[Kap12] Deepak Kapgate. "Wireless streetlight control system." In: *International Journal of Computer Applications* 41.2 (2012), pp. 1–7.

[Kau17] Fabian Kaup. "Energy-efficiency and Performance in Communication Networks: Analyzing Energy-performance Trade-offs in Communication Networks and Their Implications on Future Network Structure and Management." PhD thesis. Technische Universität, 2017.

[KBS16] Andrew L Kun, Susanne Boll, and Albrecht Schmidt. "Shifting Gears: User Interfaces in the Age of Autonomous Driving." In: *IEEE Pervasive Computing* 15.1 (Jan. 2016), pp. 32–38. DOI: 10.1109/MPRV.2016.14.

[KF18] Sebastian Kauschke and Johannes Fürnkranz. "Batchwise Patching of Classifiers." In: *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. AAAI'18. AAAI. 2018, pp. 3374–3381. URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16099.

[KFF19] Sebastian Kauschke, Lukas Fleckenstein, and Johannes Fürnkranz. "Mending Is Better Than Ending: Adapting Immutable Classifiers to Nonstationary Environments Using Ensembles of Patches." In: *Proceedings of the 2019 International Joint Conference on Neural Networks*. IJCNN'19. IEEE. 2019, pp. 1–8. DOI: 10.1109/IJCNN.2019.8852148.

[KFK06] Minkyong Kim, Jeffrey J Fielding, and David Kotz. "Risks of Using AP Locations Discovered Through war Driving." In: *Proceedings of the International Conference on Pervasive Computing*. Pervasive'06. Springer. 2006, pp. 67–82. DOI: 10.1007/11748625_5.

[Kis+16] Julia Kiseleva, Kyle Williams, Jiepu Jiang, Ahmed Hassan Awadallah, Aidan C. Crook, Imed Zitouni, and Tasos Anastasakos. "Understanding User Satisfaction With Intelligent Assistants." In: *Proceedings of the 2016 ACM on Conference on Human Information Interaction and Retrieval*. CHIIR'16. ACM, 2016, pp. 121–130. DOI: 10.1145/2854946.2854961.

[KLF19] Sebastian Kauschke, David H Lehmann, and Johannes Fürnkranz. "Patching Deep Neural Networks for Nonstationary Environments." In: *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2019, pp. 1–8. DOI: 10.1109/IJCNN.2019.8852222.

[KML06] Piotr Kaminski, Hausi Müller, and Marin Litoiu. "A Design for Adaptive Web Service Evolution." In: *Proceedings of the 2006 International Workshop on Self-Adaptation and Self-Managing Systems*. SEAMS'06. ACM, 2006, pp. 86–92. DOI: 10.1145/1137677.1137694.

[Kni+18] Pascal Knierim, Steffen Maurer, Katrin Wolf, and Markus Funk. "Quadcopter-projected In-situ Navigation Cues for Improved Location Awareness." In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI'18. ACM, 2018, pp. 1–6. DOI: 10.1145/3173574.3174007.

[Knu98] Donald E Knuth. *The Art of Computer Programming: Volume 3: Sorting and Searching*. Addison-Wesley Professional, 1998. ISBN: 978-0201896855.

[Koh+17] Florian Kohnhäuser, Milan Stute, Lars Baumgärtner, Lars Almon, Stefan Katzenbeisser, Matthias Hollick, and Bernd Freisleben. "SEDCOS: A Secure Device-to-device Communication System for Disaster Scenarios." In: *Proceedings of the 2017 IEEE 42nd Conference on Local Computer Networks*. LCN'17. IEEE. 2017, pp. 195–198. DOI: 10.1109/LCN.2017.47.

[Kra+02] Daniel Krajzewicz, Georg Hertkorn, Christian Rössel, and Peter Wagner. "SUMO (Simulation of Urban Mobility) - An Open-source Traffic Simulation." In: *Proceedings of the 4th Middle East Symposium on Simulation and Modelling*. MESM'02. MESM. 2002, pp. 183–187. ISBN: 90-77039-09-0.

[Kra+18] Klaudia Krawiecka, Arseny Kurnikov, Andrew Paverd, Mohammad Mannan, and N Asokan. "SafeKeeper: Protecting web Passwords Using Trusted Execution Environments." In: *Proceedings of the 2018 World Wide Web Conference*. WWW'18. ACM, 2018, pp. 349–358. DOI: 10.1145/3178876.3186101.

[Ku+16] Meng-Lin Ku, Wei Li, Yan Chen, and KJ Ray Liu. "Advances in Energy Harvesting Communications: Past, Present, and Future Challenges." In: *IEEE Communications Surveys & Tutorials* 18.2 (Nov. 2016), pp. 1384–1412. DOI: 10.1109/COMST.2015.2497324.

[KV00] Young-Bae Ko and Nitin H Vaidya. "Location-Aided Routing (LAR) in Mobile ad hoc Networks." In: *Wireless networks* 6.4 (Sept. 2000), pp. 307–321. DOI: 10.1023/A:1019106118419.

[KV99] Y-B Ko and Nitin H Vaidya. "Geocasting in Mobile ad hoc Networks: Location-based Multicast Algorithms." In: *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*. WMCSA'99. IEEE. 1999, pp. 101–110. DOI: 10.1109/MCSA.1999.749282.

[KWM11] Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. "Activity Recognition Using Cell Phone Accelerometers." In: *ACM SigKDD Explorations Newsletter* 12.2 (Mar. 2011), pp. 74–82. DOI: 10.1145/1964897.1964918.

[KZ16] Rida Khatoun and Sherali Zeadally. "Smart Cities: Concepts, Architectures, Research Opportunities." In: *Communications of the ACM* 59.8 (Aug. 2016), pp. 46–57. DOI: 10.1145/2858789.

[Lan+10] Nicholas D Lane, Emiliano Miluzzo, Hong Lu, Daniel Peebles, Tanzeem Choudhury, and Andrew T Campbell. "A Survey of Mobile Phone Sensing." In: *IEEE Communications Magazine* 48.9 (Sept. 2010), pp. 140–150. DOI: 10.1109/MCOM.2010.5560598.

[Lan+11] Nicholas D Lane, Ye Xu, Hong Lu, Shaohan Hu, Tanzeem Choudhury, Andrew T Campbell, and Feng Zhao. "Enabling Large-scale Human Activity Inference on Smartphones Using Community Similarity Networks (CSN)." In: *Proceedings of the 13th International Conference on Ubiquitous Computing*. UbiComp'11. ACM, 2011, pp. 355–364. DOI: 10.1145/2030112.2030160.

[Lan+14] Nicholas D. Lane, Mu Lin, Mashfiqui Mohammod, Xiaochao Yang, Hong Lu, Giuseppe Cardone, Shahid Ali, Afsaneh Doryab, Ethan Berke, Andrew T. Campbell, and Tanzeem Choudhury. "BeWell: Sensing Sleep, Physical Activities and Social Interactions to Promote Wellbeing." In: *Mobile Networks and Applications* 19.3 (Jan. 2014), pp. 345–359. DOI: 10.1007/s11036-013-0484-5.

[Lan02] Marc Langheinrich. "A Privacy Awareness System for Ubiquitous Computing Environments." In: *Proceedings of the International Conference on Ubiquitous Computing*. UbiComp'02. Springer. 2002, pp. 237–245. DOI: 10.1007/3-540-45809-3_19.

[Lan19] Carl Landwehr. "2018: A Big Year for Privacy." In: *Communications of the ACM* 62.2 (Feb. 2019), pp. 20–22. DOI: 10.1145/3300224.

[Lat+13] Neal Lathia, Veljko Pejovic, Kiran K Rachuri, Cecilia Mascolo, Mirco Musolesi, and Peter J Rentfrow. "Smartphones for Large-Scale Behavior Change Interventions." In: *IEEE Pervasive Computing* 12.3 (July 2013), pp. 66–73. DOI: 10.1109/MPRV.2013.56.

[LDL18] Nhan Ly-Trong, Chuong Dang-Le-Bao, and Quan Le-Trung. "Towards a Large-scale IoT Emulation Testbed Based on Container Technology." In: *Proceedings of the 2018 IEEE Seventh International Conference on Communications and Electronics*. ICCE'18. IEEE. 2018, pp. 63–68. DOI: 10.1109/CCE.2018.8465578.

[Lee+13] Sangmin Lee, Edmund L Wong, Deepak Goel, Mike Dahlin, and Vitaly Shmatikov. "πBox: A Platform for Privacy-preserving Apps." In: *10th USENIX Symposium on Networked Systems Design and Implementation*. NSDI'13. USENIX Association, 2013, pp. 501–514. URL: https://www.usenix.org/conference/nsdi13/technical-sessions/presentation/lee_sangmin.

[Li+14] Yanlin Li, Jonathan McCune, James Newsome, Adrian Perrig, Brandon Baker, and Will Drewry. "MiniBox: A Two-way Sandbox for x86 Native Code." In: *Proceedings of the 2014 USENIX Annual Technical Conference*. ATC'14. USENIX Association, 2014, pp. 409–420. URL: https://www.usenix.org/conference/atc14/technical-sessions/presentation/li_yanlin.

[Li+19] En Li, Liekang Zeng, Zhi Zhou, and Xu Chen. "Edge AI: On-demand Accelerating Deep Neural Network Inference via Edge Computing." In: *IEEE Transactions on Wireless Communications* 19.1 (Oct. 2019), pp. 447–457. DOI: 10.1109/TWC.2019.2946140.

[Lia+00] Wen-Hwa Liao, Yu-Chee Tseng, Kuo-Lun Lo, and Jang-Ping Sheu. "GeoGRID: A Geocasting Protocol for Mobile ad hoc Networks Based on GRID." In: *Journal of Internet Technology* 1.2 (Dec. 2000), pp. 23–32.

[Lia+07] Lin Liao, Donald J Patterson, Dieter Fox, and Henry Kautz. "Learning and Inferring Transportation Routines." In: *Artificial Intelligence* 171.5-6 (Apr. 2007), pp. 311–331. DOI: 10.1016/j.artint.2007.01.006.

[Lin+18] Li Lin, Peng Li, Xiaofei Liao, Hai Jin, and Yu Zhang. "Echo: An Edge-centric Code Offloading System With Quality of Service Guarantee." In: *IEEE Access* 7 (Nov. 2018), pp. 5905–5917. DOI: 10.1109/ACCESS.2018.2883291.

[Lin05] Yehida Lindell. "Secure Multiparty Computation for Privacy Preserving Data Mining." In: *Encyclopedia of Data Warehousing and Mining*. IGI Global, 2005, pp. 1005–1009. DOI: 10.4018/978-1-59140-557-3.ch189.

[Liu+12] Bin Liu, Yurong Jiang, Fei Sha, and Ramesh Govindan. "Cloud-enabled Privacy-preserving Collaborative Learning for Mobile Sensing." In: *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*. SenSys'12. ACM, 2012, pp. 57–70. DOI: 10.1145/2426656.2426663.

[LKH14] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. "Facing the Cold Start Problem in Recommender Systems." In: *Expert Systems with Applications* 41.4 (Mar. 2014), pp. 2065–2073. DOI: 10.1016/j.eswa.2013.09.005.

[LLV07] Ninghui Li, Tiancheng Li, and S. Venkatasubramanian. "T-Closeness: Privacy Beyond K-Anonymity and L-Diversity." In: *Proceedings of the 2007 IEEE 23rd International Conference on Data Engineering*. ICDE'07. IEEE, 2007, pp. 106–115. DOI: 10.1109/ICDE.2007.367856.

[LM18] Yaniv Leviathan and Yossi Matias. *Google Duplex: An AI System for Accomplishing Real-world Tasks Over the Phone*. Google AI Blog. https://ai.googleblog.com/2018/05/duplex-ai-system-for-natural-conversation.html. 2018.

[LQG17] Gustavo López, Luis Quesada, and Luis A Guerrero. "Alexa vs. Siri vs. Cortana vs. Google Assistant: A Comparison of Speech-based Natural User Interfaces." In: *Advances in Human Factors and Systems Interaction*. AHFE'17. Springer. 2017, pp. 241–250. DOI: 10.1007/978-3-319-60366-7_23.

[LS16] Ewa Luger and Abigail Sellen. "Like Having a Really bad pA: The Gulf Between User Expectation and Experience of Conversational Agents." In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI'16. ACM, 2016, pp. 5286–5297. DOI: 10.1145/2858036.2858288.

[LS93] George F. Luger and William A. Stubblefield. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Benjamin/Cummings Pub. Co., 1993. ISBN: 9780805347814.

[Lu+12] Hong Lu, Denise Frauendorfer, Mashfiqui Rabbi, Marianne Schmid Mast, Gokul T Chittaranjan, Andrew T Campbell, Daniel Gatica-Perez, and Tanzeem Choudhury. "StressSense: Detecting Stress in Unconstrained Acoustic Environments Using Smartphones." In: *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. UbiComp'12. ACM, 2012, pp. 351–360. DOI: 10.1145/2370216.2370270.

[Lun+18] Pietro Lungaro, Rickard Sjöberg, Alfredo Jose Fanghella Valero, Ashutosh Mittal, and Konrad Tollmar. "Gaze-aware Streaming Solutions for the Next Generation of Mobile VR Experiences." In: *IEEE transactions on visualization and computer graphics* 24.4 (Apr. 2018), pp. 1535–1544. DOI: 10.1109/TVCG.2018.2794119.

[LV18] AJ Shalini Lakshmi and M Vijayalakshmi. "LI-RMLO: Loop Iteration Based Runtime Method Level Offloading for Mobile Applications." In: *Proceedings of the 2018 International Conference on Smart Systems and Inventive Technology*. ICSSIT'18. IEEE. 2018, pp. 136–140. DOI: 10.1109/ICSSIT.2018.8748500.

[Lyn19] Dorian Lynskey. *'Alexa, Are you Invading my Privacy?' – The Dark Side of our Voice Assistants*. The Guardian. 2019. URL: https://www.theguardian.com/technology/2019/oct/09/alexa-are-you-invading-my-privacy-the-dark-side-of-our-voice-assistants.

[Mac+07] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. "L-diversity: Privacy Beyond K-anonymity." In: *ACM Transactions on Knowledge Discovery From Data* 1.1 (Mar. 2007), p. 3. DOI: 10.1145/1217299.1217302.

[Mar+08] Ivan Marsa-Maestre, Miguel A Lopez-Carmona, Juan R Velasco, and Andres Navarro. "Mobile Agents for Service Personalization in Smart Environments." In: *Journal of Networks* 3.5 (May 2008), pp. 30–41. DOI: 10.4304/jnw.3.5.30-41.

[Mar19] Bernard Marr. *What Is Homomorphic Encryption? And Why Is It So Transformative?* https://www.forbes.com/sites/bernardmarr/2019/11/15/what-is-homomorphic-encryption-and-why-is-it-so-transformative. 2019.

[Mas+18] Rahat Masood, Dinusha Vatsalan, Muhammad Ikram, and Mohamed Ali Kaafar. "Incognito: A Method for Obfuscating web Data." In: *Proceedings of the 2018 World Wide Web Conference*. WWW'18. ACM, 2018, pp. 267–276. DOI: 10.1145/3178876.3186093.

[Mau+16] Markus Maurer, J Christian Gerdes, Barbara Lenz, and Hermann Winner. *Autonomous Driving*. Springer Berlin Heidelberg, 2016. ISBN: 978-3-662-48845-4. DOI: 10.1007/978-3-662-48847-8.

[MB17] Pavel Mach and Zdenek Becvar. "Mobile Edge Computing: A Survey on Architecture and Computation Offloading." In: *IEEE Communications Surveys & Tutorials* 19.3 (Mar. 2017), pp. 1628–1656. DOI: 10.1109/COMST.2017.2682318.

[McC08] Evie McCrum-Gardner. "Which is the correct statistical test to use?" In: *British Journal of Oral and Maxillofacial Surgery* 46.1 (2008), pp. 38–41. DOI: 10.1016/j.bjoms.2007.09.002.

[McM+17] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. "Communication-Efficient Learning of Deep Networks From Decentralized Data." In: *Proceedings of Machine Learning Research*. AISTATS'17. PMLR, 2017, pp. 1273–1282. URL: http://proceedings.mlr.press/v54/mcmahan17a.html.

[Men+17] Hamid Menouar, Ismail Guvenc, Kemal Akkaya, A Selcuk Uluagac, Abdullah Kadri, and Adem Tuncer. "UAV-enabled Intelligent Transportation Systems for the Smart City: Applications and Challenges." In: *IEEE Communications Magazine* 55.3 (Mar. 2017), pp. 22–28. DOI: 10.1109/MCOM.2017.1600238CM.

[Mik+19] Mateusz Mikusz, Kenny Tsu Wei Choo, Rajesh Krishna Balan, Nigel Davies, and Youngki Lee. "New Challenges in Display-Saturated Environments." In: *IEEE Pervasive Computing* 18.2 (2019), pp. 67–75. DOI: 10.1109/MPRV.2019.2906992.

[Mit97] Thomas M. Mitchell. *Machine Learning*. 1st ed. New York, NY, USA: McGraw-Hill, Inc., 1997. ISBN: 0070428077, 9780070428072.

[MM17] Abhinav Mehrotra and Mirco Musolesi. *Intelligent Notification Systems: A Survey of the State of the Art and Research Challenges*. 2017. arXiv: 1711.10171.

[Mok+17] Sonia Ben Mokhtar, Antoine Boutet, Louafi Bouzouina, Patrick Bonnel, Olivier Brette, Lionel Brunie, Mathieu Cunche, Stephane D'Alu, Vincent Primault, Patrice Raveneau, et al. "PRIVA'MOV: Analysing Human Mobility Through Multi-Sensor Datasets." In: *NetMob 2017*. 2017. URL: https://hal.inria.fr/hal-01578557.

[Mon+14] Yves-Alexandre de Montjoye, Erez Shmueli, Samuel S Wang, and Alex Sandy Pentland. "OpenPDS: Protecting the Privacy of Metadata Through Safeanswers." In: *PloS One* 9.7 (July 2014), pp. 1–9. DOI: 10.1371/journal.pone.0098790.

[Moo03] Adam D Moore. "Privacy: Its Meaning and Value." In: *American Philosophical Quarterly* 40.3 (July 2003), pp. 215–227. URL: https://www.jstor.org/stable/20010117.

[Mor+16] Richard Mortier et al. "Personal Data Management With the Databox: What's Inside the Box?" In: *Proceedings of the 2016 ACM Workshop on Cloud-Assisted Networking*. CAN'16. ACM, 2016, pp. 49–54. DOI: 10.1145/3010079.3010082.

[Mun+10] Min Mun, Shuai Hao, Nilesh Mishra, Katie Shilton, Jeff Burke, Deborah Estrin, Mark Hansen, and Ramesh Govindan. "Personal Data Vaults: A Locus of Control for Personal Data Streams." In: *Proceedings of the 6th International Conference on Emerging Networking Experiments and Technologies*. CoNEXT'10. ACM, 2010, 17:1–17:12. DOI: 10.1145/1921168.1921191.

[Mur+08] Rohan Narayana Murty, Geoffrey Mainland, Ian Rose, Atanu Roy Chowdhury, Abhimanyu Gosain, Josh Bers, and Matt Welsh. "CitySense: An Urban-scale Wireless Sensor Network and Testbed." In: *Proceedings of the 2008 IEEE Conference on Technologies for Homeland Security*. HST'08. IEEE. 2008, pp. 583–588. DOI: 10.1109/THS.2008.4534518.

[MZ17] Payman Mohassel and Yupeng Zhang. "SecureML: A System for Scalable Privacy-preserving Machine Learning." In: *Proceedings of the 2017 IEEE Symposium on Security and Privacy*. SP'17. IEEE. 2017, pp. 19–38. DOI: 10.1109/SP.2017.12.

[MZH17] Cristina Mihale-Wilson, Jan Zibuschka, and Oliver Hinz. "About User Preferences and Willingness to Pay for a Secure and Privacy Protective Ubiquitous Personal Assistant." In: *Proceedings of the 25th European Conference on Information Systems*. ECIS'17. AIS Electronic Library, 2017, pp. 32–47. ISBN: 978-989-20-7655-3.

[MZH19] Cristina Mihale-Wilson, Jan Zibuschka, and Oliver Hinz. "User Preferences and Willingness to Pay for In-vehicle Assistance." In: *Electronic Markets* 29.1 (Feb. 2019), pp. 37–53. DOI: 10.1007/s12525-019-00330-5.

[Nah+16] Klara Nahrstedt, Daniel Lopresti, Ben Zorn, Ann W. Drobnis, Beth Mynatt, Shwetak Patel, and Helen V. Wright. *Smart Communities Internet of Things*. 2016. arXiv: 1604.02028 [cs.CY].

[Nak+12] Shinsuke Nakajima, Daisuke Kitayama, Yoshitaka Sushita, Kazutoshi Sumiya, Naiwala P Chandrasiri, and Kazunari Nawa. "Route Recommendation Method for Car Navigation System Based on Estimation of Driver's Intent." In: *Proceedings of the 2012 IEEE International Conference on Vehicular Electronics and Safety*. ICVES'12. IEEE. 2012, pp. 318–323. DOI: 10.1109/ICVES.2012.6294305.

[Nar+12] Arvind Narayanan, Vincent Toubiana, Solon Barocas, Helen Nissenbaum, and Dan Boneh. *A Critical Look at Decentralized Personal Data Architectures*. 2012. arXiv: 1202.4503.

[NHH07] Patricia A Norberg, Daniel R Horne, and David A Horne. "The privacy paradox: Personal information disclosure intentions versus behaviors." In: *Journal of consumer affairs* 41.1 (2007), pp. 100–126. DOI: 10.1111/j.1745-6606.2006.00070.x.

[NI15] Huseyin Naci and John PA Ioannidis. "Evaluation of Wellness Determinants and Interventions by Citizen Scientists." In: *Jama* 314.2 (July 2015), pp. 121–122. DOI: 10.1001/jama.2015.6160.

[Nie09] Jakob Nielsen. *Powers of 10: Time Scales in User Experience*. Web page. 2009. URL: https://www.nngroup.com/articles/powers-of-10-time-scales-in-ux.

[NN08] Anthony J Nicholson and Brian D Noble. "Breadcrumbs: Forecasting Mobile Connectivity." In: *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*. MobiCom'08. ACM, 2008, pp. 46–57. DOI: 10.1145/1409944.1409952.

[Ohr+16] Olga Ohrimenko, Felix Schuster, Cédric Fournet, Aastha Mehta, Sebastian Nowozin, Kapil Vaswani, and Manuel Costa. "Oblivious Multi-party Machine Learning on Trusted Processors." In: *Proceedings of the 25th USENIX Conference on Security Symposium*. SEC'16. USENIX Association, 2016, pp. 619–636. URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/ohrimenko.

[Ole+18] Oleksii Oleksenko, Bohdan Trach, Robert Krahn, Mark Silberstein, and Christof Fetzer. "Varys: Protecting SGX Enclaves From Practical Side-Channel Attacks." In: *2018 USENIX Annual Technical Conference*. ATC'18. USENIX Association, 2018, pp. 227–240. ISBN: 978-1-939133-02-1. URL: https://www.usenix.org/conference/atc18/presentation/oleksenko.

[Org01] World Health Organization. *The World Health Report 2001: Mental Health: New Understanding, New Hope*. World Health Organization, 2001. ISBN: 9241562013.

[Ort+16] Sergio Orts-Escolano et al. "Holoportation: Virtual 3D Teleportation in Real-time." In: *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. UIST'16. ACM, 2016, pp. 741–754. DOI: doi.org/10.1145/2984511.2984517.

[Osi+17] Seyed Ali Osia, Ali Shahin Shamsabadi, Sina Sajadmanesh, Ali Taheri, Kleomenis Katevas, Hamid R. Rabiee, Nicholas D. Lane, and Hamed Haddadi. *A Hybrid Deep Learning Architecture for Privacy-Preserving Mobile Analytics*. 2017. arXiv: 1703.02952.

[Ouy+19] Tao Ouyang, Rui Li, Xu Chen, Zhi Zhou, and Xin Tang. "Adaptive User-managed Service Placement for Mobile Edge Computing: An Online Learning Approach." In: *Proceedings of the IEEE Conference on Computer Communications*. INFOCOM'19. IEEE. 2019, pp. 1468–1476. DOI: 10.1109/INFOCOM.2019.8737560.

[OZC18] Tao Ouyang, Zhi Zhou, and Xu Chen. "Follow me at the Edge: Mobility-aware Dynamic Service Placement for Mobile Edge Computing." In: *IEEE Journal on Selected Areas in Communications* 36.10 (Oct. 2018), pp. 2333–2345. DOI: 10.1109/JSAC.2018.2869954.

[Pan+12] Kamill Panitzek, Immanuel Schweizer, Axel Schulz, Tobias Bönning, Gero Seipel, and Max Mühlhäuser. "Can we use Your Router, Please?: Benefits and Implications of an Emergency Switch for Wireless Routers." In: *International Journal of Information Systems for Crisis Response and Management (IJISCRAM)* 4.4 (Apr. 2012), pp. 59–70. DOI: 10.4018/jiscrm.2012100104.

[Pap+17] Elias P. Papadopoulos, Michalis Diamantaris, Panagiotis Papadopoulos, Thanasis Petsas, Sotiris Ioannidis, and Evangelos P. Markatos. "The Long-Standing Privacy Debate: Mobile Websites vs Mobile Apps." In: *Proceedings of the 26th International Conference on World Wide Web*. WWW'17. ACM, 2017, pp. 153–162. DOI: 10.1145/3038912.3052691.

[Pat12] Manas A Pathak. *Privacy-preserving machine learning for speech processing*. Springer Science & Business Media, 2012.

[Pen+19] Zhenhui Peng, Yunhwan Kwon, Jiaan Lu, Ziming Wu, and Xiaojuan Ma. "Design and Evaluation of Service Robot's Proactivity in Decision-Making Support Process." In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI'19. ACM, 2019, 98:1–98:13. DOI: 10.1145/3290605.3300328.

[Pen09] Alex Pentland. "Reality Mining of Mobile Communications: Toward a New Deal on Data." In: *Social Computing and Behavioral Modeling*. Springer-Verlag US, 2009, pp. 75–80. ISBN: 978-1-4419-0056-2. DOI: 10.1007/978-1-4419-0056-2_1.

[Per+17] Charith Perera, Susan YL Wakenshaw, Tim Baarslag, Hamed Haddadi, Arosha K Bandara, Richard Mortier, Andy Crabtree, Irene CL Ng, Derek McAuley, and Jon Crowcroft. "Valorising the IoT Databox: Creating Value for Everyone." In: *Transactions on Emerging Telecommunications Technologies* 28.1 (Jan. 2017), pp. 1–17. DOI: 10.1002/ett.3125.

[PGD17] Otto Julio Ahlert Pinno, Andre Ricardo Abed Gregio, and Luis CE De Bona. "ControlChain: Blockchain as a Central Enabler for Access Control Authorizations in the IoT." In: *Proceedings of the 2017 IEEE Global Communications Conference*. GLOBECOM'17. IEEE. 2017, pp. 1–6. DOI: 10.1109/GLOCOM.2017.8254521.

[Pie+17] Martin Pielot, Bruno Cardoso, Kleomenis Katevas, Joan Serrà, Aleksandar Matic, and Nuria Oliver. "Beyond Interruptibility: Predicting Opportune Moments to Engage Mobile Phone Users." In: *IProceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1.3 (Sept. 2017), 91:1–91:25. DOI: 10.1145/3130956.

[Pin02] Benny Pinkas. "Cryptographic Techniques for Privacy-preserving Data Mining." In: *ACM SigKDD Explorations Newsletter* 4.2 (Dec. 2002), pp. 12–19. DOI: 10.1145/772862.772865.

[PM14] Veljko Pejovic and Mirco Musolesi. "InterruptMe: Designing Intelligent Prompting Mechanisms for Pervasive Applications." In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. UbiComp'14. ACM, 2014, pp. 897–908. DOI: 10.1145/2632048.2632062.

[PM15] Veljko Pejovic and Mirco Musolesi. "Anticipatory Mobile Computing: A Survey of the State of the Art and Research Challenges." In: *ACM Computing Surveys* 47.3 (Apr. 2015), 47:1–47:29. DOI: 10.1145/2693843.

[Pol18] Jeff Pollard. *The top Five Emerging Technologies Security Leaders Need to Prepare For*. Forrester. 2018. URL: https://www.forrester.com/report/The+Top+Five+Emerging+Technologies+Security+Leaders+Need+To+Prepare+For/-/E-RES143513.

[Pow11] David Martin Powers. "Evaluation: From Precision, Recall and F-measure to ROC, Informedness, Markedness and Correlation." In: *Journal of Machine Learning Technologies* 2.1 (Feb. 2011), pp. 37–63. ISSN: 2229-3981. DOI: 10.9735/2229-3981.

[Pri+20] Christian Priebe, Divya Muthukumaran, Joshua Lind, Huanzhou Zhu, Shujie Cui, Vasily A. Sartakov, and Peter Pietzuch. *SGX-LKL: Securing the Host OS Interface for Trusted Execution*. 2020. arXiv: 1908.11143 [cs.OS].

[PW10] Ofir Pele and Michael Werman. "The quadratic-chi histogram distance family." In: *European Conference on Computer Vision*. Springer. 2010, pp. 749–762. DOI: 10.1007/978-3-642-15552-9_54.

[PY09] Sinno Jialin Pan and Qiang Yang. "A Survey on Transfer Learning." In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (Oct. 2009), pp. 1345–1359. DOI: 10.1109/TKDE.2009.191.

[QA11] Ben Quinn and Charles Arthur. *PlayStation Network Hackers Access Data of 77 Million Users*. The Guardian. 2011. URL: https://www.theguardian.com/technology/2011/apr/26/playstation-network-hackers-data.

[Qin+12] Shao-Meng Qin, Hannu Verkasalo, Mikael Mohtaschemi, Tuomo Hartonen, and Mikko Alava. "Patterns, Entropy, and Predictability of Human Mobility and Life." In: *PloS one* 7.12 (Dec. 2012), e51353. DOI: 10.1371/journal.pone.0051353.

[Qiu+18] Hang Qiu, Fawad Ahmad, Fan Bai, Marco Gruteser, and Ramesh Govindan. "AVR: Augmented Vehicular Reality." In: *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. MobiSys'18. ACM, 2018, pp. 81–95. DOI: 10.1145/3210240.3210319.

[Rab+15] Mashfiqui Rabbi, Min Hane Aung, Mi Zhang, and Tanzeem Choudhury. "MyBehavior: Automatic Personalized Health Feedback From User Behaviors and Preferences Using Smartphones." In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. UbiComp'15. ACM, 2015, pp. 707–718. DOI: 10.1145/2750858.2805840.

[Rac+10] Kiran K Rachuri, Mirco Musolesi, Cecilia Mascolo, Peter J Rentfrow, Chris Longworth, and Andrius Aucinas. "EmotionSense: A Mobile Phones Based Adaptive Platform for Experimental Social Psychology Research." In: *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*. UbiComp'10. ACM, 2010, pp. 281–290. DOI: 10.1145/1864349.1864393.

[Rac+11] Kiran K Rachuri, Cecilia Mascolo, Mirco Musolesi, and Peter J Rentfrow. "SociableSense: Exploring the Trade-Offs of Adaptive Sampling and Computation Offloading for Social Sensing." In: *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*. MobiCom'11. ACM, 2011, pp. 73–84. DOI: 10.1145/2030613.2030623.

[Rag15] Dave Raggett. "The Web of Things: Challenges and Opportunities." In: *Computer* 48.5 (May 2015), pp. 26–32. DOI: 10.1109/MC.2015.149.

[RBF13] Amr Rizk, Zdravko Bozakov, and Markus Fidler. "Estimating Traffic Correlations From Sampling and Active Network Probing." In: *Proceedings of the 2013 IFIP Networking Conference*. IEEE. 2013, pp. 1–9. ISBN: 978-3-901882-55-5.

[RGO13] Moo-Ryong Ra, Ramesh Govindan, and Antonio Ortega. "P3: Toward Privacy-Preserving Photo Sharing." In: *Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation*. NSDI'13. USENIX Association, 2013, pp. 515–528. URL: https://www.usenix.org/conference/nsdi13/technical-sessions/presentation/ra.

[RJL21] Pasika Ranaweera, Anca Delia Jurcut, and Madhusanka Liyanage. "Survey on multi-access edge computing security and privacy." In: *IEEE Communications Surveys & Tutorials* 23.2 (2021), pp. 1078–1124. DOI: 10.1109/COMST.2021.3062546.

[RN16] Stuart J Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education Limited, 2016. ISBN: 0132071487.

[RR98] Michael K. MK Reiter and Aviel D. AD Rubin. "Crowds: Anonymity for web Transactions." In: *ACM Transactions on Information and System Security* 1.1 (Nov. 1998), pp. 66–92. DOI: 10.1145/290163.290168.

[RRK18] Bita Darvish Rouhani, M Sadegh Riazi, and Farinaz Koushanfar. "DeepSecure: Scalable Provably-secure Deep Learning." In: *Proceedings of the 55th Annual Design Automation Conference*. DAC'18. ACM, 2018, 2:1–2:6. DOI: 10.1145/3195970.3196023.

[Sad11] Fariba Sadri. "Ambient Intelligence: A Survey." In: *ACM Computing Surveys* 43.4 (Oct. 2011), 36:1–36:66. DOI: 10.1145/1978802.1978815.

[Sam+16] Andrei Vlad Sambra, Essam Mansour, Sandro Hawke, Maged Zereba, Nicola Greco, Abdurrahman Ghanem, Dmitri Zagidulin, Ashraf Aboulnaga, and Tim Berners-Lee. *Solid: A Platform for Decentralized Social Applications Based on Linked Data*. Tech. rep. MIT CSAIL & Qatar Computing Research Institute, 2016.

[Sam94] Ferdinando Silvestro Samaria. "Face Recognition Using Hidden Markov Models." PhD thesis. University of Cambridge Cambridge, UK, 1994.

[San+17] Eduardo Felipe Zambom Santana, Ana Paula Chaves, Marco Aurelio Gerosa, Fabio Kon, and Dejan S Milojicic. "Software Platforms for Smart Cities: Concepts, Requirements, Challenges, and a Unified Reference Architecture." In: *ACM Computing Surveys* 50.6 (Nov. 2017), pp. 1–37. DOI: 10.1145/3124391.

[Sap+15] Piotr Sapiezynski, Radu Gatej, Alan Mislove, and Sune Lehmann. "Opportunities and Challenges in Crowdsourced Wardriving." In: *Proceedings of the 2015 Internet Measurement Conference*. IMC'15. ACM, 2015, pp. 267–273. DOI: 10.1145/2815675.2815711.

[Sar17] Ruhi Sarikaya. "The Technology Behind Personal Digital Assistants: An Overview of the System Architecture and key Components." In: *IEEE Signal Processing Magazine* 34.1 (Jan. 2017), pp. 67–81. DOI: 10.1109/MSP.2016.2617341.

[Sat+09] Mahadev Satyanarayanan, Paramvir Bahl, Ramón Caceres, and Nigel Davies. "The Case for VM-based Cloudlets in Mobile Computing." In: *IEEE Pervasive Computing* 8.4 (Oct. 2009), pp. 14–23. DOI: 10.1109/MPRV.2009.82.

[Sat17] Mahadev Satyanarayanan. "The Emergence of Edge Computing." In: *Computer* 50.1 (Jan. 2017), pp. 30–39. DOI: 10.1109/MC.2017.9.

[Sch+11] Klaus Schwab, Alan Marcus, JO Oyola, William Hoffman, and M Luzi. *Personal Data: The Emergence of a New Asset Class*. Tech. rep. http://www3.weforum.org/docs/WEF_ITTC_PersonalDataNewAsset_Report_2011.pdf. 2011.

[Sch+15b] Felix Schuster, Manuel Costa, Cédric Fournet, Christos Gkantsidis, Marcus Peinado, Gloria Mainar-Ruiz, and Mark Russinovich. "VC3: Trustworthy Data Analytics in the Cloud Using SGX." In: *Proceedings of the 2015 IEEE Symposium on Security and Privacy*. SP'15. IEEE. 2015, pp. 38–54. DOI: 10.1109/SP.2015.10.

[Sch+17] Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. "DBSCAN Revisited, Revisited: Why and how you Should (still) use DBSCAN." In: *ACM Transactions on Database Systems (TODS)* 42.3 (July 2017), pp. 1–21. DOI: 10.1145/3068335.

[SDL20] Farah Aït Salaht, Frédéric Desprez, and Adrien Lebre. "An Overview of Service Placement Problem in Fog and Edge Computing." In: *ACM Computing Surveys* 53.3 (June 2020), pp. 1–35. DOI: 10.1145/3391196.

[Ser+18] Sandra Servia-Rodríguez, Liang Wang, Jianxin Zhao, Richard Mortier, and Hamed Haddadi. "Privacy-preserving Personal Model Training." In: *Proceedings of the 2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation*. IoTDI'18. IEEE. 2018, pp. 153–164. DOI: 0.1109/IoTDI.2018.00024.

[Set+14] Endah Setyaningsih, Lydwina Wardhani, Joni Fat, and Ida Zureidar. "Performance of LED lights installed on DKI Jakarta streets (Case study on Pattimura street & Satrio street, South Jakarta)." In: *2014 Electrical Power, Electronics, Communicatons, Control and Informatics Seminar (EECCIS)*. IEEE. 2014, pp. 45–50. DOI: 10.1109/EECCIS.2014.7003717.

[Sha+17] Hossein Shafagh, Lukas Burkhalter, Anwar Hithnawi, and Simon Duquennoy. "Towards Blockchain-based Auditable Storage and Sharing of IoT Data." In: *Proceedings of the 2017 on Cloud Computing Security Workshop*. CCSW '17. ACM, 2017, pp. 45–50. DOI: 10.1145/3140649.3140656.

[She18] Sam Shead. *Apple CEO Tim Cook Issues AI Warning: 'It Must Respect Human Values, Including Privacy'*. Forbes. 2018. URL: https://www.forbes.com/sites/samshead/2018/10/24/apple-ceo-tim-cook-issues-ai-warning-it-must-respect-human-values-including-privacy/%5C#51cf86f584d4.

[Shi+18] Fatemeh Shirazi, Milivoj Simeonovski, Muhammad Rizwan Asghar, Michael Backes, and Claudia Diaz. "A survey on routing in anonymous communication protocols." In: *ACM Computing Surveys (CSUR)* 51.3 (2018), pp. 1–39. DOI: 10.1145/3182658.

[Sho+17] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. "Membership inference attacks against machine learning models." In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2017, pp. 3–18. DOI: 10.1109/SP.2017.41.

[Shv+19] Yan Shvartzshnaider, Zvonimir Pavlinovic, Ananth Balashankar, Thomas Wies, Lakshminarayanan Subramanian, Helen Nissenbaum, and Prateek Mittal. "VACCINE: Using Contextual Integrity for Data Leakage Detection." In: *Proceedings of the World Wide Web Conference*. WWW'19. ACM, 2019, pp. 1702–1712. DOI: 10.1145/3308558.3313655.

[Sie+18] Volkmar Sieh, Robert Burlacu, Timo Hönig, Heiko Janker, Phillip Raffeck, Peter Wägemann, and Wolfgang Schröder-Preikschat. "Combining Automated Measurement-Based Cost Modeling With Static Worst-Case Execution-Time and Energy-Consumption Analyses." In: *IEEE Embedded Systems Letters* 11.2 (2018), pp. 38–41. DOI: 10.1109/LES.2018.2868823.

[Sim+16] Meryem Simsek, Adnan Aijaz, Mischa Dohler, Joachim Sachs, and Gerhard Fettweis. "5G-enabled Tactile Internet." In: *IEEE J-SAC* 34.3 (Feb. 2016), pp. 460–473. DOI: 10.1109/JSAC.2016.2525398.

[SKH18] Milan Stute, David Kreitschmann, and Matthias Hollick. "One Billion Apples' Secret Sauce: Recipe for the Apple Wireless Direct Link ad hoc Protocol." In: *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. MobiCom'18. ACM, 2018, pp. 529–543. DOI: 10.1145/3241539.3241566.

[Sol06] Daniel J Solove. "A Taxonomy of Privacy." In: *U. Pa. L. Rev.* 154.3 (Jan. 2006), pp. 477–564. DOI: 10.2307/40041279.

[Sol21] Daniel J Solove. "The myth of the privacy paradox." In: *Geo. Wash. L. Rev.* 89 (2021), p. 1. URL: https://scholarship.law.gwu.edu/faculty_publications/1482.

[SS14a] Immanuel Schweizer and Benedikt Schmidt. "Kraken.me: Multi-device User Tracking Suite." In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. UbiComp'14 Adjunct. ACM, 2014, pp. 853–862. DOI: 10.1145/2638728.2641307.

[SS14b] Roberta Sinatra and Michael Szell. "Entropy and the Predictability of Online Life." In: *Entropy* 16.1 (Jan. 2014), pp. 543–556. DOI: 10.3390/e16010543.

[SS15] Reza Shokri and Vitaly Shmatikov. "Privacy-preserving Deep Learning." In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. CCS'15. ACM, 2015, pp. 1310–1321. DOI: 10.1145/2810103.2813687.

[ST19] Gürkan Solmaz and Damla Turgut. "A Survey of Human Mobility Models." In: *IEEE Access* 7 (Sept. 2019), pp. 125711–125731. DOI: 10.1109/ACCESS.2019.2939203.

[Ste+17] Michael Stein, Mathias Fischer, Immanuel Schweizer, and Max Mühlhäuser. "A Classification of Locality in Network Research." In: *ACM Computing Surveys* 50.4 (Aug. 2017), pp. 1–37. DOI: 10.1145/3092693.

[Sun+16] Yu Sun, Nicholas Jing Yuan, Yingzi Wang, Xing Xie, Kieran McDonald, and Rui Zhang. "Contextual Intent Tracking for Personal Assistants." In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD'16. ACM, 2016, pp. 273–282. DOI: 10.1145/2939672.2939676.

[SWV15] Philipp M Scholl, Matthias Wille, and Kristof Van Laerhoven. "Wearables in the Wet Lab: A Laboratory System for Capturing and Guiding Experiments." In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. UbiComp'15. ACM, 2015, pp. 589–599. DOI: 10.1145/2750858.2807547.

[Tah+18] Salman Taherizadeh, Andrew C Jones, Ian Taylor, Zhiming Zhao, and Vlado Stankovski. "Monitoring self-adaptive applications within edge computing frameworks: A state-of-the-art review." In: *Journal of Systems and Software* 136 (2018), pp. 19–38. DOI: 10.1016/j.jss.2017.10.033.

[Tan03] Andrew S. Tanenbaum. *Computer Networks*. Prentice hall, 2003. ISBN: 978-0130661029.

[TAP17] Paul Tran-Van, Nicolas Anciaux, and Philippe Pucheral. "SWYSWYK: A Privacy-by-design Paradigm for Personal Information Management Systems." In: *Proceedings of the 26th International Conference on Information Systems Development*. ISD'17. AIS, 2017, pp. 1–12.

[Tei+13] Thiago S. F. X. Teixeira, George Teodoro, Eduardo Valle, and Joel H. Saltz. *Scalable Locality-Sensitive Hashing for Similarity Search in High-Dimensional, Large-Scale Multimedia Datasets*. 2013. arXiv: 1310.4136.

[TM15] Fani Tsapeli and Mirco Musolesi. "Investigating Causality in Human Behavior From Smartphone Sensor Data: A Quasi-experimental Approach." In: *EPJ Data Science* 4.1 (Dec. 2015), p. 24. DOI: 10.1140/epjds/s13688-015-0061-1.

[TPV17] Chia-Che Tsai, Donald E Porter, and Mona Vij. "Graphene-SGX: A Practical Library OS for Unmodified Applications on SGX." In: *Proceedings of the 2017 USENIX Annual Technical Conference*. ATC'17. USENIX Association, 2017, pp. 645–658. ISBN: 978-1-931971-38-6. URL: https://www.usenix.org/conference/atc17/technical-sessions/presentation/tsai.

[TR05] Daniel Teghe and Kathryn Rendell. "Social Well-being: A Literature Review." In: *School of Social Work and Welfare Studies* 10 (Aug. 2005), pp. 1–20. DOI: 10.13140/RG.2.2.28891.26406.

[TV07] Andrew S Tanenbaum and Maarten Van Steen. *Distributed Systems: Principles and Paradigms*. Prentice-Hall, 2007. ISBN: 978-1530281756.

[VB18] Blesson Varghese and Rajkumar Buyya. "Next Generation Cloud Computing: New Trends and Research Directions." In: *Future Generation Computer Systems* 79.3 (Feb. 2018), pp. 849–861. DOI: 10.1016/j.future.2017.09.020.

[VH08] András Varga and Rudolf Hornig. "An Overview of the OMNeT++ Simulation Environment." In: *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*. Simutools'08. ICST. 2008, pp. 1–10. ISBN: 9789639799202.

[VO14] Max Van Kleek and Kieron OHara. "The Future of Social Is Personal: The Potential of the Personal Data Store." In: *Social Collective Intelligence*. Springer, 2014, pp. 125–158. DOI: 10.1007/978-3-319-08681-1_7.

[Wan+18a] Lin Wang, Lei Jiao, Ting He, Jun Li, and Max Mühlhäuser. "Service Entity Placement for Social Virtual Reality Applications in Edge Computing." In: *Proceedings of the 2018 IEEE Conference on Computer Communications*. INFOCOM'18. IEEE. 2018, pp. 468–476. DOI: 10.1109/INFOCOM.2018.8486411.

[Wan+18b] Lin Wang, Lei Jiao, Jun Li, Julien Gedeon, and Max Mühlhäuser. "Moera: Mobility-agnostic Online Resource Allocation for Edge Computing." In: *IEEE Transactions on Mobile Computing* 18.8 (Aug. 2018), pp. 1843–1856. DOI: 10.1109/TMC.2018.2867520.

[Wan+18c] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan. "When Edge Meets Learning: Adaptive Control for Resource-Constrained Distributed Machine Learning." In: *Proceedings of the IEEE Conference on Computer Communications*. INFOCOM'18. IEEE. 2018, pp. 63–71. DOI: 10.1109/INFOCOM.2018.8486403.

[Wei91] Mark Weiser. "The Computer for the 21st Century." In: *Scientific American* 265.3 (Sept. 1991), pp. 94–104. DOI: 10.1038/scientificamerican0991-94.

[Wes68] Alan F Westin. "Privacy and Freedom." In: *Washington and Lee Law Review* 25.1 (1968), pp. 166–170. DOI: 10.1093/sw/13.4.114-a.

[Wie+17] Jason Wiese, Sauvik Das, Jason I. Hong, and John Zimmerman. "Evolving the Ecosystem of Personal Behavioral Data." In: *Human–Computer Interaction* 32.5-6 (Nov. 2017), pp. 447–510. DOI: 10.1080/07370024.2017.1295857.

[WLW98] Huaiqing Wang, Matthew KO Lee, and Chen Wang. "Consumer Privacy Concerns About Internet Marketing." In: *Communications of the ACM* 41.3 (Mar. 1998), pp. 63–70. DOI: 10.1145/272287.272299.

[WS08] Georg Wittenburg and Jochen Schiller. "A Survey of Current Directions in Service Placement in Mobile Ad-hoc Networks." In: *Proceedings of the 2008 Sixth Annual IEEE International Conference on Pervasive Computing and Communications*. PerCom'08. IEEE. 2008, pp. 548–553. DOI: 10.1109/PERCOM.2008.43.

[WW14] Jiaqiu Wang and Zhongjie Wang. "A Survey on Personal Data Cloud." In: *The Scientific World Journal* 2014 (Aug. 2014), pp. 1–13. DOI: 10.1155/2014/969150.

[Xu+11] Heng Xu, Xin Robert Luo, John Carroll, and Mary Rosson. "The Personalization Privacy Paradox: An Exploratory Study of Decision Making Process for Location-aware Marketing." In: *Decision Support Systems* 51.1 (Apr. 2011), pp. 42–52. DOI: 10.1016/j.dss.2010.11.017.

[Xu+15] Zichuan Xu, Weifa Liang, Wenzheng Xu, Mike Jia, and Song Guo. "Efficient Algorithms for Capacitated Cloudlet Placements." In: *IEEE Transactions on Parallel and Distributed Systems* 27.10 (Oct. 2015), pp. 2866–2880. DOI: 10.1109/TPDS.2015.2510638.

[Xu+19] Xiaolong Xu, Xihua Liu, Zhanyang Xu, Fei Dai, Xuyun Zhang, and Lianyong Qi. "Trust-oriented IoT Service Placement for Smart Cities in Edge Computing." In: *IEEE Internet of Things Journal* 7.5 (Dec. 2019), pp. 4084–4091. DOI: 10.1109/JIOT.2019.2959124.

[XYW19] Depeng Xu, Shuhan Yuan, and Xintao Wu. "Achieving Differential Privacy and Fairness in Logistic Regression." In: *Companion Proceedings of The 2019 World Wide Web Conference*. WWW'19: Companion. ACM, 2019, pp. 594–599. DOI: 10.1145/3308560.3317584.

[XZ15] Zhi Xu and Sencun Zhu. "SemaDroid: A Privacy-Aware Sensor Management Framework for Smartphones." In: *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*. CODASPY'15. ACM, 2015, pp. 61–72. DOI: 10.1145/2699026.2699114.

[Yan+19] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. "Federated Machine Learning: Concept and Applications." In: *ACM Transactions on Intelligent Systems and Technology*. TIST 10.2 (Jan. 2019), 12:1–12:19. DOI: 10.1145/3298981.

[Yee17] Derek Yee. *High Performance NetScaler in the Cloud*. https://www.citrix.com/blogs/2017/07/18/high-performance-netscaler-in-the-cloud. 2017.

[YGR17] Zhu Yan, Guhua Gan, and Khaled Riad. "BC-PDS: Protecting Privacy and Self-sovereignty Through Blockchains for OpenPDS." In: *Proceedings of the 2017 IEEE Symposium on Service-Oriented System Engineering*. SOSE'17. IEEE. 2017, pp. 138–144. DOI: 10.1109/SOSE.2017.30.

[Yil15] Pinar Yildirim. "Filter based feature selection methods for prediction of risks in hepatitis disease." In: *International Journal of Machine Learning and Computing* 5.4 (2015), p. 258. DOI: 10.7763/IJMLC.2015.V5.517.

[Yor+12] Neil Yorke-Smith, Shahin Saadati, Karen L Myers, and David N Morley. "The Design of a Proactive Personal Agent for Task Management." In: *International Journal on Artificial Intelligence Tools* 21.01 (Jan. 2012), p. 1250004. DOI: 10.1142/S0218213012500042.

[You+19] Ethan G Young, Pengfei Zhu, Tyler Caraza-Harter, Andrea C Arpaci-Dusseau, and Remzi H Arpaci-Dusseau. "The true cost of containing: A gVisor case study." In: *11th {USENIX} Workshop on Hot Topics in Cloud Computing (HotCloud 19)*. 2019.

[Yua+14] Hua Yuan, Yu Qian, Rui Yang, and Ming Ren. "Human Mobility Discovering and Movement Intention Detection With GPS Trajectories." In: *Decision Support Systems* 63 (July 2014), pp. 39–51. DOI: 10.1016/j.dss.2013.09.010.

[Zha+16] Yan Zhai, Lichao Yin, Jeffrey Chase, Thomas Ristenpart, and Michael Swift. "CQSTR: Securing Cross-Tenant Applications With Cloud Containers." In: *Proceedings of the Seventh ACM Symposium on Cloud Computing*. SoCC'16. ACM, 2016, pp. 223–236. DOI: 10.1145/2987550.2987558.

[Zha+17] Kuan Zhang, Jianbing Ni, Kan Yang, Xiaohui Liang, Ju Ren, and Xuemin Sherman Shen. "Security and Privacy in Smart City Applications: Challenges and Solutions." In: *IEEE Communications Magazine* 55.1 (Jan. 2017), pp. 122–129. DOI: 10.1109/MCOM.2017.1600267CM.

[Zha+18] Zhongliang Zhao, Lucas Guardalben, Mostafa Karimzadeh, Jose Silva, Torsten Braun, and Susana Sargento. "Mobility Prediction-assisted Over-the-top Edge Prefetching for Hierarchical VANETs." In: *IEEE journal on selected areas in communications* 36.8 (June 2018), pp. 1786–1801. DOI: 10.1109/JSAC.2018.2844681.

[Zhe+08a] Yu Zheng, Quannan Li, Yukun Chen, Xing Xie, and Wei-Ying Ma. "Understanding Mobility Based on GPS Data." In: *Proceedings of the 10th International Conference on Ubiquitous computing*. UbiComp'08. ACM, 2008, pp. 312–321. DOI: 10.1145/1409635.1409677.

[Zhe+08b] Yu Zheng, Like Liu, Longhao Wang, and Xing Xie. "Learning Transportation Mode From raw GPS Data for Geographic Applications on the Web." In: *Proceedings of the 17th International Conference on World Wide Web*. WWW'08. ACM, 2008, pp. 247–256. DOI: 10.1145/1367497.1367532.

[Zhe+09] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. "Mining Interesting Locations and Travel Sequences From GPS Trajectories." In: *Proceedings of the 18th International Conference on World Wide Web*. WWW'09. ACM, 2009, pp. 791–800. DOI: 10.1145/1526709.1526816.

[Zhe+15] Kan Zheng, Long Zhao, Jie Mei, Mischa Dohler, Wei Xiang, and Yuexing Peng. "10 gb/s Hetsnets With Millimeter-wave Communications: Access and Networking-challenges and Protocols." In: *IEEE Communications Magazine* 53.1 (Jan. 2015), pp. 222–231. DOI: 10.1109/MCOM.2015.7010538.

[Zhe+17] Wenting Zheng, Ankur Dave, Jethro G Beekman, Raluca Ada Popa, Joseph E Gonzalez, and Ion Stoica. "Opaque: An Oblivious and Encrypted Distributed Analytics Platform." In: *Proceedings of the 14th USENIX Symposium on Networked Systems Design and Implementation*. NSDI'17. USENIX Association, 2017, pp. 283–298. ISBN: 978-1-931971-37-9. URL: https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/zheng.

[ZHK19] Jan Zibuschka, Moritz Horsch, and Michael Kubach. "The ENTOURAGE Privacy and Security Reference Architecture for Internet of Things Ecosystems." In: *Open Identity Summit 2019*. Gesellschaft für Informatik, Bonn, 2019, pp. 119–130. ISBN: 978-3-88579-687-9.

[Zho+19] Zhi Zhou, Xu Chen, En Li, Liekang Zeng, Ke Luo, and Junshan Zhang. "Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing." In: *Proceedings of the IEEE* 107.8 (Aug. 2019), pp. 1738–1762. DOI: 10.1109/JPROC.2019.2918951.

[Zhu+19] Qingyi Zhu, Seng W. Loke, Rolando Trujillo-Rasua, Frank Jiang, and Yong Xiang. "Applications of Distributed Ledger Technologies to the Internet of Things: A Survey." In: *ACM Computing Surveys*. CSUR 52.6 (Nov. 2019), 120:1–120:34. DOI: 10.1145/3359982.

[Zia+16] M Tarek Ibn Ziad, Amr Alanwar, Moustafa Alzantot, and Mani Srivastava. "CryptoImg: Privacy Preserving Processing Over Encrypted Images." In: *Proceedings of the IEEE Conference on Communications and Network Security*. CNS'16. IEEE. 2016, pp. 570–575. DOI: 10.1109/CNS.2016.7860550.

[Zim95] Philip R Zimmermann. *The Official PGP User's Guide*. Vol. 5. MIT press Cambridge, 1995.

[ZNP15] Guy Zyskind, Oz Nathan, and Alex Pentland. *Enigma: Decentralized Computation Platform with Guaranteed Privacy*. 2015. arXiv: 1506.03471 [cs.CR].

[ZXL19] Rui Zhang, Rui Xue, and Ling Liu. "Security and Privacy on Blockchain." In: *ACM Computing Surveys*. CSUR 52.3 (July 2019), 51:1–51:34. DOI: 10.1145/3316481.

[ZXM10] Yu Zheng, Xing Xie, and Wei-Ying Ma. "GeoLife: A Collaborative Social Networking Service Among User, Location and Trajectory." In: *IEEE Data(base) Engineering Bulletin* 33.2 (June 2010), pp. 32–39.

All web pages cited in this work have been checked in June 2021. However, due to the dynamic nature of the World Wide Web, their long-term availability cannot be guaranteed.

APPENDIX

A

USER EXPECTATIONS STUDY

This section provides additional information to the user expectations study presented in Section 2.3 (p. 22 ff.). The material is taken from [Meu+20a].

A.1 USE CASES

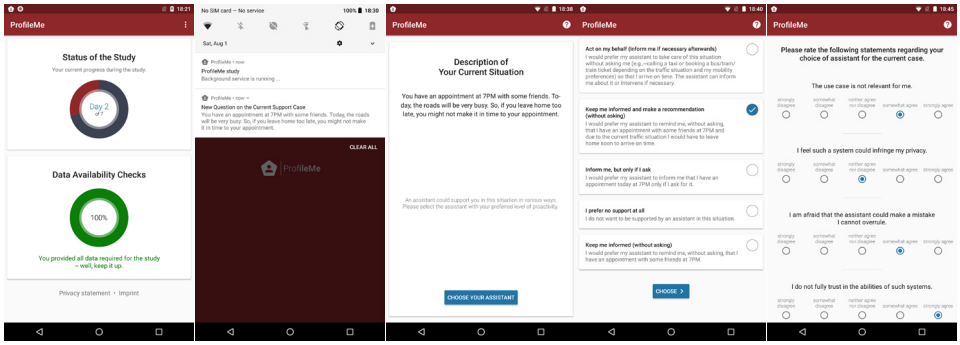
The following 23 use cases were used for the study, covering the different application domains introduced in Section 2.1.5.

Table A.1: Overview of the selected *use cases* used in the study

Area	Sub-area	Use cases
Healthcare & Well-being (12)	*physical*	(#1) sleepiness, (#2) fitness plan, (#3) medication intake, (#4) physical complaints
	mental	(#5) exhaustion/stress, (#6) concentration, (#7) loneliness, (#8) boredom/addiction
	social	(#9) appointments, (#10) annual reminders, (#11) reservations, (#12) postponements
Activity Support (6)	*physical*	(#13) public transportation, (#14) travel planning, (#15) shopping
	digital	(#16) finance, (#17) emails, (#18) *do not disturb* handling
Ambient Intelligence (5)	–	(#19) lighting and climate control, (#20) housekeeping, (#21) laundry management, (#22) reorders, (#23) assignments

A.2 STUDY TOOL

The following mobile app, namely `ProfileMe`, was developed as a study tool specifically for the new study methodology (see Section 2.3.1).



(a) Overview (progress) (b) In-situ Notification (c) Use Case Description (d) Proactivity Rating (e) Additional Questions

Figure A.1: Screenshots of the `ProfileMe` application

A.3 ADDITIONAL QUESTIONS

Participants were asked to answer the following set of questions after they had made their choice regarding the level of proactivity in the respective in-situ survey. Each question was rated on a five-item Likert scale ranging from *strongly disagree* to *strongly agree*.

Table A.2: Overview of the *additional questions* to better understand the choices made by the participants regarding the proactivity level – the term 'assistant' is used as a substitute for 'AI service' to better communicate the more technical concept of AI services to the participants.

ID	Additional Questions
Q1	The use case is not relevant for me.
Q2	The use case is too important to let an assistant decide for me.
Q3	I am afraid that the assistant could make a mistake I cannot overrule.
Q4	I do not fully trust in the abilities of such systems.
Q5	An assistant which has more functionalities would be too intrusive.
Q6	I feel such a system could infringe my privacy.

B

EHRENWÖRTLICHE ERKLÄRUNG

Hiermit erkläre ich, die vorgelegte Arbeit zur Erlangung des akademischen Grades *Doktor-Ingenieur* (*Dr.-Ing.*) mit dem Titel

Data Protection in Personalized AI Services: A Decentralized Approach

selbständig und ausschließlich unter Verwendung der angegebenen Hilfsmittel erstellt zu haben. Ich habe bisher noch keinen Promotionsversuch unternommen.

Darmstadt, 30.06.2021

Christian Meurisch