



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

## TASK RECOMMENDATION IN CROWDSOURCING PLATFORMS

Worker Centered Approaches for Task Assignment and Recommendation  
in Online Task Markets based on Textual Descriptions

Dem Fachbereich Informatik  
der Technischen Universität Darmstadt  
zur Erlangung des akademischen Grades eines  
Doktor-Ingenieurs (Dr.-Ing.)  
genehmigte Dissertation

VON

STEFFEN MATTHIAS SCHNITZER, M.SC.

Geboren am 13. Februar 1987 in Mannheim-Neckarau

Vorsitz: Prof. Dr. Kristian Kersting  
Referent: Prof. Dr.-Ing. Ralf Steinmetz  
Korreferent: PD Dr.-Ing. Christoph Rensing

Tag der Einreichung: 19. Dezember 2018

Tag der Disputation: 12. Februar 2019

Hochschulkennziffer D17  
Darmstadt 2019

Steffen Matthias Schnitzer, M.Sc.: *Task Recommendation in Crowdsourcing Platforms, Worker Centered Approaches for Task Assignment and Recommendation in Online Task Markets based on Textual Descriptions.*

Darmstadt, Technische Universität Darmstadt,  
Tag der mündlichen Prüfung: 12.02.2019  
Jahr der Veröffentlichung auf TUprints: 2019

Dieses Dokument wird bereitgestellt von: This document is provided by:  
tuprints, E-Publishing-Service der Technischen Universität Darmstadt.  
<http://tuprints.ulb.tu-darmstadt.de>  
[tuprints@ulb.tu-darmstadt.de](mailto:tuprints@ulb.tu-darmstadt.de)

Bitte zitieren sie dieses Dokument als: Please cite this document as:  
URN: urn:nbn:de:tuda-tuprints-85496  
URL: <http://tuprints.ulb.tu-darmstadt.de/8549>

Die Veröffentlichung steht unter folgender Creative Common Lizenz:  
Namensnennung - Nicht kommerziell - Keine Bearbeitungen 4.0 International  
CC BY-NC-ND 4.0 international  
<https://creativecommons.org/licenses/by-nc-nd/4.0/deed.de>

This publication is licensed under the following Creative Commons License:  
Attribution-NonCommercial-NoDerivatives 4.0 International  
CC BY-NC-ND 4.0 international  
<https://creativecommons.org/licenses/by-nc-nd/4.0/deed.en>



## ABSTRACT

---

Task distribution platforms, such as micro-task markets, project assignment portals, and job search engines, support the assignment of tasks to workers. Public crowdsourcing platforms support the assignment of tasks in micro-task markets to help task requesters to complete their tasks and allow workers to earn money. Enterprise crowdsourcing platforms provide a marketplace within enterprises for the internal placement of tasks from employers to employees. Most of both types of task distribution platforms rely on the workers' selection capabilities or provide simple filtering steps to reduce the number of tasks a worker can choose from. This self-selection mechanism unfortunately allows for tasks to be performed by under- or over-qualified workers. Supporting the workers by introducing a task recommender system helps to solve such deficits of existing task distributions.

In this thesis, the requirements towards task recommendation in task distribution platforms are gathered with a focus on the worker's perspective, the design of appropriate assignment strategies is described, and innovative methods to recommend tasks based on their textual descriptions are provided. Different viewpoints are taken into account by analyzing the domains of micro-tasks, project assignments, and job postings. The requirements of enterprise crowdsourcing platforms are compiled based on the literature and a qualitative study, providing a conceptual design of task assignment strategies. The demands of workers and their perception of task similarity on public crowdsourcing platforms are identified, leading to the design and implementation of additional methods to determine the similarity of micro-tasks. The textual descriptions of micro-tasks, projects, and job postings are analyzed in order to provide innovative methods for task recommendation in these domains.



Aufgabenverteilungsplattformen im Internet, wie Jobsuchmaschinen, Projektzuordnungsportale und Marktplätze für Mikroaufgaben, bieten Dienste zur Zuweisung von Aufgaben an Personen. Viele öffentliche Crowdsourcing-Plattformen unterstützen die Vergabe von Mikroaufgaben, um Anfragestellern bei der Erledigung ihrer Aufgaben zu helfen und Arbeitern die Möglichkeit zu geben Geld zu verdienen. Enterprise Crowdsourcing-Plattformen bieten Marktplätze für eine unternehmensinterne Übertragung von Aufgaben vom Arbeitgeber an den Arbeitnehmer. Die meisten Aufgabenverteilungsplattformen verwenden eine manuelle Auswahl der Aufgaben durch den Arbeiter. Sie bieten oftmals einfache Filtermöglichkeiten, um die Anzahl der Aufgaben zu reduzieren, aus denen ein Arbeiter auswählen kann. Diese Form der manuellen Selbstauswahl von Aufgaben führt dazu, dass viele Aufgaben von unter- oder überqualifizierten Arbeitern ausgeführt werden. Die Einführung eines Aufgabenempfehlungssystems kann solche Defizite der Aufgabenverteilung reduzieren.

In dieser Arbeit werden die Anforderungen an die Aufgabenempfehlung in Aufgabenverteilungsplattformen mit einem Fokus auf die Perspektive des Arbeitnehmers gesammelt, das Design geeigneter Zuordnungsstrategien beschrieben, und innovative Methoden zur Empfehlung von Aufgaben auf der Grundlage ihrer textlichen Beschreibungen vorgestellt. Verschiedene Aspekte werden durch die Analyse dreier Formen von Aufgabenverteilungsplattformen berücksichtigt, nämlich Plattformen für Mikroaufgaben, Projektaufträge und Stellenausschreibungen. Die Anforderungen an Enterprise Crowdsourcing-Plattformen werden auf Basis der Literatur und einer qualitativen Studie gesammelt. Auf dieser Sammlung als Grundlage werden verschiedene Aufgabenzuordnungsstrategien konzipiert. Außerdem werden die Anforderungen der Arbeiter und ihr Verständnis einer Aufgabenähnlichkeit auf öffentlichen Crowdsourcing-Plattformen ermittelt, was die Entwicklung und Implementierung zusätzlicher Methoden zur Bestimmung der Ähnlichkeit von Mikroaufgaben motiviert. Die textlichen Beschreibungen von Mikroaufgaben, Projektaufträgen und Stellenausschreibungen werden analysiert, um innovative Methoden zur Aufgabenempfehlung in allen drei Formen von Aufgabenverteilungsplattformen realisieren zu können.



## DANKSAGUNG

---

Meine Promotion hätte ich ohne die Unterstützung einer Menge toller Menschen nicht so erfolgreich abschließen können. Diesen wichtigen Menschen möchte ich hier dafür danken, dass sie mir auf dem Weg dahin mit Rat und Tat zur Seite standen.

Zunächst möchte ich mich bei Ralf Steinmetz bedanken, der meine Arbeit betreut hat, den Grundstein legt für die gute Zusammenarbeit bei KOM, Chancen eröffnet und Möglichkeiten auftut. Mindestens im gleichen Maße darf ich mich bei meinem Gruppenleiter Christoph Rensing bedanken, der mich in seine Gruppe geholt und in allen Situationen immer unterstützt hat. Er hat immer dazu beigetragen, den richtigen Weg hin zur Promotion zu finden und hat sich regelmäßig mit meinen Arbeiten auseinandergesetzt und wertvolles Feedback geliefert. Vielen Dank auch an meinen Mentor Marek, der mich auf dem Weg immer mit großem Interesse und gutem Rat begleitet hat.

Insgesamt darf ich mich bei allen Kollegen von KOM, aber ganz besonders bei der Knowledge Media Gruppe bedanken. Sebastian, der meine Masterarbeit betreut hat und mir damit den Weg zu KOM eröffnet hat. Mit dem man immer viel Spaß im Büro haben konnte, oder auch mal beim Wandern in Nepal. Genauso aber auch an die anderen Kollegen, Irina und Wael, mit denen ich immer viel Spaß daran hatte die deutsche Sprache, Kultur und Absurditäten im Büro zu diskutieren. Eine sehr "lehrreiche" Zeit war auch die Zusammenarbeit mit Lena, gemeinsam mit Boris und Alex, bei der wir NCS bzw. CNUVS betreut haben, und mit viel Spaß unsere Zeit mit Tutoren, Lernzielen, Übungsaufgaben und Klausuren verbracht haben. Vielen Dank auch an Svenja, deren Masterarbeit mich auch inhaltlich sehr vorangebracht hat und die ich dann zunächst als Kollegin gewinnen konnte. Auch an Stephan und Tim noch ein großes Danke, ich wünsche Euch noch eine großartige Zeit bei KM!

Dann darf ich mich natürlich noch bei allen anderen KOMlern und KOM Alumnis bedanken. Björn, Dominik, Patrick, Ronny, Sophie, Viktor, und allen, mit denen die Besuche am Böllenfalltor immer ein besonderes Erlebnis sind. Aber auch allen anderen bei KOM, ohne Anspruch auf Vollständigkeit, mit denen ich mal mehr und mal weniger Zeit verbringen konnte: Binh, Daniel, Frank, Nils, Rahul, Tobi M., Tobi R., Thomas, und alle anderen.

Ein ganz besonderer Dank geht hier natürlich an unsere ATMs. Karola, die immer ein offenes Ohr für die Problemchen der Doktoranden hat. Genauso aber auch an Frau Scholz-Schmidt, Frau Ehlhardt, Frank, Sabine, Moni, Jan, Zeynep, und Britta, aber auch an die 'neuen' Julia und Michaela.

Vielen Dank auch an meine HiWis und Studierenden für ihren Einsatz an meinen Forschungsthemen. Insbesondere Christian Borg-Krebs, Nils Jansen, Daniel Specht, Pushkar Rawat, Dominik Reis und Sebastian Wollny.

Dann möchte ich noch ein paar Freunde erwähnen, denen ich es zu verdanken habe, dass ich hier in Darmstadt so gut angekommen bin. Dazu gehören Julia und

Klaus, die mich in ihre WG aufgenommen haben, sonst würde ich heute wohl noch in Mannheim wohnen. Die gesamte Quizgruppe mit Anna, Peter, Alex, Natascha, Lisa, Stefan, Strassi, etc., über die ich in Darmstadt viel Freude, Freunde, Arbeit und noch wichtigere Dinge finden konnte.

Ganz wichtig war für mich auch die Unterstützung durch meine Familie, die mich unterstützt egal was ich mache und immer hinter mir steht.

Zu guter Letzt gilt der größte Dank Dir Ellen. Den größeren Teil dieses großen Abenteuers Promotion hast Du mit mir durchgestanden und mich dabei auf allen Hochs und Tiefs begleitet die einem Unterwegs begegnen. Ohne Deine Unterstützung und große Geduld mit mir, insbesondere bis zur Einreichung der Dissertation, aber auch darüber hinaus, hätte ich diesen Weg so nicht beschreiten können. Du weißt genau was mir das bedeutet. - *Danke!*

## CONTENTS

---

<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
1.1	Motivation for Task Recommendation . . . . .	2
1.2	Research Challenges . . . . .	3
1.3	Research Goals and Contributions . . . . .	5
1.4	Structure of the Thesis . . . . .	6
<b>2</b>	<b>FOUNDATIONS</b>	<b>7</b>
2.1	Crowdsourcing Tasks . . . . .	7
2.2	Different Kinds of Task Markets . . . . .	9
2.2.1	Micro-Task Markets . . . . .	9
2.2.2	Public and Enterprise Crowdsourcing Platforms . . . . .	10
2.2.3	Location-Based Crowdsourcing Platforms . . . . .	10
2.2.4	Project Staffing . . . . .	10
2.2.5	Job Market Platforms . . . . .	11
2.3	Task Recommendation . . . . .	11
2.3.1	Recommender Systems . . . . .	11
2.3.2	Task Recommendation in Task Markets . . . . .	12
2.3.3	Task Recommendation in Micro-Task Markets . . . . .	13
2.3.4	Task Recommendation in other Task Markets . . . . .	14
2.4	Natural Language Processing . . . . .	16
2.4.1	Text Processing and Annotations . . . . .	16
2.4.2	Word and Document Representations: The Vector Space Model . . . . .	18
2.4.3	Text Similarity . . . . .	19
2.5	Clustering and Classification using Machine Learning . . . . .	21
2.5.1	Clustering Methods . . . . .	21
2.5.2	Classification Methods . . . . .	22
2.6	Evaluation Metrics . . . . .	22
<b>3</b>	<b>MODELING ENTERPRISE CROWDSOURCING PLATFORMS</b>	<b>25</b>
3.1	Related Work for Modeling Enterprise Crowdsourcing Platforms . . . . .	26
3.2	Structured Literature Review . . . . .	28
3.2.1	Conceptualization and Literature Search . . . . .	29
3.2.2	Categorization of Enterprise Crowdsourcing Scenarios . . . . .	30
3.2.3	Elements of Enterprise Crowdsourcing . . . . .	33
3.2.4	Instruments of Enterprise Crowdsourcing . . . . .	35
3.2.5	Extended Enterprise Crowdsourcing Model . . . . .	39
3.3	Qualitative Study . . . . .	41
3.3.1	Methodology of the Qualitative Study . . . . .	42
3.3.2	Quantitative Results of the Study . . . . .	43

3.3.3	Scenario Identification and Description . . . . .	44
3.4	Design Aspects of Enterprise Crowdsourcing Platforms . . . . .	47
3.4.1	Basic Structure of Enterprise Crowdsourcing Platforms . . . . .	47
3.4.2	Design of Task Assignment Strategies and Workflows . . . . .	49
3.5	User Evaluation of Selected Workflows . . . . .	51
3.6	Conclusion . . . . .	53
<b>4</b>	<b>WORKER’S DEMANDS FOR TASK RECOMMENDATION IN PUBLIC CROWD-SOURCING PLATFORMS</b>	<b>55</b>
4.1	Related Work for User Studies in Public Crowdsourcing Platforms . . . . .	55
4.2	Workers Preferences on Task Recommendation . . . . .	56
4.2.1	Methodology . . . . .	56
4.2.2	Results . . . . .	57
4.2.3	Conclusion . . . . .	58
4.3	Similarity of Tasks from the Worker’s Perspective . . . . .	59
4.3.1	Methodology . . . . .	59
4.3.2	Results . . . . .	61
4.3.3	Conclusion . . . . .	63
<b>5</b>	<b>TASK RECOMMENDATION</b>	<b>65</b>
5.1	Related Work for Task Recommendation . . . . .	66
5.1.1	Recommendation of Micro-Tasks and Task Classification . . . . .	66
5.1.2	Crowd Selection . . . . .	68
5.1.3	Project Staffing and Skill Extraction . . . . .	68
5.1.4	Recommendation of Job Postings . . . . .	70
5.2	Determining Micro-Task Similarity for Task Recommendation . . . . .	72
5.2.1	Classification of Micro-Tasks . . . . .	73
5.2.2	Micro-Task Similarities Regarding the Required Action . . . . .	78
5.2.3	Conclusion . . . . .	85
5.3	Crowd Selection in Location-Based Crowdsourcing . . . . .	86
5.3.1	Methodology for Crowd Selection based on Textual Similarities . . . . .	86
5.3.2	Evaluation of Crowd Selection and Task Recommendation . . . . .	88
5.3.3	Conclusion . . . . .	91
5.4	Skill Extraction for Project Assignment Recommendation . . . . .	92
5.4.1	Methodology of Skill Extraction and Project Assignments . . . . .	92
5.4.2	Evaluation of Skill Extraction and Project Assignments . . . . .	96
5.4.3	Conclusion . . . . .	97
5.5	Preselection of Documents for the Recommendation of Job Postings . . . . .	98
5.5.1	Methodology of Preselection and Job Recommendation . . . . .	99
5.5.2	Evaluation of Preselection and Job Recommendation . . . . .	106
5.5.3	Conclusion . . . . .	111
<b>6</b>	<b>SUMMARY, CONCLUSIONS, AND OUTLOOK</b>	<b>113</b>
6.1	Summary of the Thesis . . . . .	113
6.1.1	Contributions . . . . .	113

6.1.2 Conclusions . . . . .	115
6.2 Outlook . . . . .	115
<b>BIBLIOGRAPHY</b>	<b>117</b>
<b>A APPENDIX</b>	<b>133</b>
A.1 Interview Guideline . . . . .	133
A.2 POS-Tags . . . . .	138
A.3 Additional Results for Determining Task Similarities . . . . .	139
A.4 Acronyms and Keywords . . . . .	143
A.4.1 Acronyms . . . . .	143
A.4.2 Keywords . . . . .	143
A.5 Supervised Student Theses . . . . .	145
<b>B AUTHOR'S PUBLICATIONS</b>	<b>147</b>
<b>C CURRICULUM VITÆ</b>	<b>149</b>
<b>D ERKLÄRUNG LAUT DER PROMOTIONSORDNUNG</b>	<b>151</b>



THE widespread use of the Internet and the global availability of services it provides, has changed many industries. Users can interact with each other and pick up tasks of service providers, located anywhere in the world, without the necessity to know each other personally. A significant change provided in a global scope is how such services affect the distribution of tasks to humans in general. Some users provide their work for free, when writing reviews, adding information on *Wikipedia* or developing *open source* software. Other users may be paid for performing specific tasks. In general, the term *crowdsourcing* describes a process where tasks are performed by more or less anonymous users which make up the crowd.

The way tasks are distributed has changed dramatically within different areas. The job market has shifted from classified ads in newspapers to job platforms on the Internet. Project staffing in enterprises can be supported by services that rely on knowledge about the qualifications of employees. Micro-tasks are distributed to the anonymous workforce of the crowd that can be reached through the Internet. Jobs, projects and micro-tasks are often assigned by online platforms providing the corresponding task markets for task requesters to publish their tasks, and for workers to find tasks to work on.

Throughout these different scenarios of distributing tasks, there is a common objective to be found: Finding a fitting task for a worker, or vice versa. This is a challenge especially if the number of available tasks respectively workers is high. A system that provides matching items to a user is called a recommender system. Recommender systems have been studied in various application areas. Within a recommender system for task distribution, the *tasks* and the *workers* have to be focused as items and users, respectively. Regarding the distribution of tasks by Public Crowdsourcing Platforms (PCPs), there exist three main stakeholders with different views and expectations towards the recommender system: The task *requester*, the crowd *worker* and the platform *provider*. Task distribution platforms are usually created and financed by the means of task requesters and platform providers. The experiences of the workers on such platforms thus very often deviate from their expectations. Therefore, this thesis provides an analysis of requirements for task recommendation, focusing on, while not being limited to, the worker's perspective.

Until now, task recommendation approaches mostly rely on static categorizations of tasks often provided by the task requester. Tasks provided in online platforms require detailed descriptions of requirements for the worker to match. Methods of Natural Language Processing (NLP) and Machine Learning (ML) offer the possibility to analyze the textual descriptions in detail and leverage semantic information. Therefore, this thesis focuses on methods leveraging the semantic information extracted from textual descriptions of tasks, in order to provide task recommendations.

## 1.1 MOTIVATION FOR TASK RECOMMENDATION

Crowdsourcing enables requesters to outsource tasks and projects to an anonymous crowd over the Internet. The definitions of crowdsourcing are extensive, reaching from including *Wikipedia*<sup>1</sup> and *open source* projects to micro-task markets like *Amazon Mechanical Turk (MTurk)*<sup>2</sup> and *Microworkers*<sup>3</sup>. In micro-task markets, individual workers complete very small tasks for corresponding rewards. Employers accomplish whole projects that are outsourced to the crowd through such Public Crowdsourcing Platforms (PCPs) by aggregating the results in a later step. Apart from these PCPs, there is the possibility for enterprises to restrict the crowd to already existing employees of the enterprise, e.g. for confidentiality reasons. Such Enterprise Crowdsourcing Platforms (ECPs) have different requirements and require different assignment strategies, as well as specialized incentive and community management. Besides micro-task markets, there are similar task markets which also manage the task distribution but focus on larger tasks like whole projects within an enterprise or even full-time jobs. Project markets enable employers to assign matching employees of the enterprise to upcoming projects. Job markets provide the job postings of employers for possible future employees to find.

Within such task markets, three dominant roles can be identified. On the one hand, there are the task *requesters*, who are basically employers offering their tasks through the platform by providing a description. On the other hand, there are the crowd *workers*, who select or get selected to complete the tasks of the requesters. The platform *provider* mainly publishes the tasks with their descriptions on the task market and manages the assignment of workers to tasks. As the requester and the worker interact only by means of the platform, the interests of the platform provider play a significant role in how the task market is shaped, and especially how the task assignment is managed.

Up to now, crowdsourcing platforms and especially micro-task market platforms leave the task assignment mostly to the workers and therefore rely on their selection capabilities. Usually, the workers are provided with a list of tasks that are available on the platform, which can be filtered or sorted by basic criteria like *newest* or *payment*. Workers then select the tasks they hope to be qualified for, to perform them successfully. Only if the results are verified by the task requester or the platform, the corresponding incentive (payment) reaches the worker; otherwise, the work is rejected. The task distribution following this self-selection assignment strategy is often referred to as “*marketplace*” [92], and the respective platforms are known as micro-task markets. The task assignment process is an essential factor to reach the goals of worker and requester on the platform and of the platform itself. In a real-world job market, the job assignment is pursued with great effort by both employer and employee to find qualified and motivated workers for the best matching job. Compared to this, the task selection process in crowdsourcing systems is mostly unsupported.

---

<sup>1</sup> [www.wikipedia.org](http://www.wikipedia.org)

<sup>2</sup> [www.mturk.com](http://www.mturk.com)

<sup>3</sup> [www.microworkers.com](http://www.microworkers.com)

It was shown in several studies that workers have difficulties to find tasks matching their skills or preferences and want to be supported by recommender systems [87, 144, 185]. Many studies report a high rejection rate of low quality work with up to 60% rejections [38, 73, 86, 144, 146], which could be due to the under-qualification of the assigned workers. Lacking motivation or the intent to reduce the taken effort by cheating or spamming is a manifestation of this problem. This leads to an increased effort for quality control and rescheduling of tasks for the employer. The outcome for the workers and the quality for the requesters suffer from this problem, while the platform provider gains nothing from low success rates as well. It may also be possible, that workers who are over-qualified for a task get assigned, broadening the problem of unmatched potential and demotivated workers. The selection of matching tasks for a worker is a challenge not only for crowdsourcing systems but for project assignment and job market platforms as well.

In e-commerce and other platforms, recommender systems are used to provide recommendations where a user interacts with the platform to find or select items provided by the platform. A recommender which enables task recommendation in task distribution platforms could support the task selection process and therefore influence the task assignment to generate better outcomes for worker and requester and follow the guidelines of the platform provider.

To leverage this potential, the goal of this thesis is to identify requirements for task recommendation, design appropriate assignment strategies and develop innovative methods for task recommendation based on the textual information of heterogeneous task descriptions.

## 1.2 RESEARCH CHALLENGES

Considering certain viewpoints in task markets, the dynamic nature of task markets and the nature of task descriptions provides interesting research challenges within the focused domain of task recommendation for task distribution platforms.

### **Challenge:** *Consideration of the worker's perspective*

The task assignment in crowdsourcing platforms is often shaped by the platform provider, answering the requirements of the task requesters. The task requesters paying for the completion of tasks on the platform also provide the primary support in financing the respective platform. Applying the self-selection assignment strategy often leads to workers working on tasks they are not qualified for. Such workers find their work rejected and can be excluded from other tasks they may be qualified for. Other workers find themselves working on tasks they are over-qualified for and do not earn a reward reflecting their potential value. This can lead to a worse outcome for worker and requester than what would be possible on the platform. Therefore, it is necessary to not only focus on the requirements of task requester and platform provider, but also on the demands of the crowd worker. Focusing on the demands of the workers towards task recommendation [146] and their perception of tasks on the platform [144] allows the proposition of new strategies for recommendation

in Chapter 4. Additionally, the concepts of enterprise crowdsourcing need to take the view of the workers into account, as they are employees of the enterprise and therefore already a valued asset of the company, which is discussed in Chapter 3.

**Challenge:** *The dynamic nature of task markets and the need for methods to calculate the similarity of tasks*

A primary challenge for task recommendation is the very dynamic nature of task markets. In contrast to e-commerce or media platforms, where items remain available indefinitely, a task disappears from the platform, as soon as the micro-task is completed or the job position is assigned. Many tasks on micro-task markets are provided in so-called *campaigns*, providing several same tasks on the platform at the same time, e.g., to find many contributors for a survey. However, as soon as one campaign is completed, the task is also not available on the platform anymore. Well studied recommender systems used in e-commerce or media recommendation are often based on collaborative approaches, which rely on the fact that a single item can be interacted with by more than one user. Therefore, such approaches cannot be directly applied to task recommendation problems. Content-based recommender approaches, using the given information of items to identify relevant recommendations can be applied instead. Different user studies show that the similarity of tasks is an essential factor for workers when selecting tasks and should be considered by recommendations [14, 146]. To exploit the potential of the available information about workers and tasks on the platform, methods to determine the similarity of tasks have to be identified. Such similarities help to relate completed tasks from the history of the workers with tasks which are currently available on the platform. They could also be used to identify workers with similar interests and therefore broaden the recommendation possibilities. The demands of the workers and the perceived similarity of tasks are discussed in Chapter 4.

**Challenge:** *Semantic analysis of unstructured task descriptions*

Micro-task, project and job descriptions are usually created for humans to be processed. Most of the information provided is therefore represented in natural language. Many examples can be found, where such descriptions do not necessarily adhere to standards of grammar and structure. Methods of NLP enable the extraction of information from human-readable documents. The field of NLP has shifted in recent years towards methods applying neural networks and deep learning to extract semantic meaning from text. Applying such approaches to task descriptions allows designing unexplored methods for task recommendation, some of which are provided in Chapter 5. However, these methods rely on huge datasets for training the respective ML approaches. Therefore, these research endeavours are undergone in cooperation with partners from the industry, providing data corpora for job postings, project descriptions and micro-tasks.

### 1.3 RESEARCH GOALS AND CONTRIBUTIONS

The main goal of this work is to examine requirements of task recommendation, design appropriate assignment strategies and explore how heterogeneous textual descriptions of tasks can be used to create personalized task recommender systems. This objective is divided into the following major research goals.

**Research Goal 1:** *Definition of requirements towards task recommendation systems.*

In the scope of RG<sub>1</sub>, the requirements towards task recommendation for crowdsourcing systems are gathered, analyzed and integrated in a model for crowdsourcing platforms. Therefore, the requirements towards an ECP are gathered in a structured literature review [142] and by providing a qualitative study with experts (n=11) from the industry, which fill the different roles of platform provider, task requester and worker in micro-task markets. Additionally, the demands of the workers towards task recommendation are substantiated in a user study on a major crowdsourcing platform [146]. Also, a survey is presented, which provides insights into the worker's perspective of crowdsourcing and shows different aspects of task similarity that are preferred by certain workers [144].

**Research Goal 2:** *Design of strategies for task assignment in crowdsourcing systems.*

Based on the gathered requirements in relation to RG<sub>1</sub>, RG<sub>2</sub> focuses on the design of different task assignment strategies. On the one hand, a conceptual design for an ECP is derived from specific industry scenarios that are modeled from the literature review [142] and the qualitative study, to provide task assignment strategies. On the other hand, the user studies on crowdsourcing platforms [144, 146] are used as a base to provide tailored methods to determine the similarity of tasks [143, 145].

**Research Goal 3:** *Providing task recommendations based on textual descriptions.*

After understanding the requirements of workers towards task recommendation in crowdsourcing systems in RG<sub>1</sub> and modeling respective assignment strategies in RG<sub>2</sub>, RG<sub>3</sub> focuses on the question whether the textual descriptions of tasks can be used to create effective task recommendation systems. The field of NLP provides the opportunity to create new foundational methods to process and derive knowledge from text in general, e.g., with methods derived from the relatively new concepts of word embeddings or new deep learning structures. While this a fast developing research area and many ground-breaking methods have been developed in recent years, this thesis focuses on the application of these methods in the domain of task descriptions and how representations of task descriptions provide possible ways for recommending tasks to workers. To answer RG<sub>3</sub> conclusively, different manifestations of tasks, namely *micro-tasks*, *projects*, and *job postings* are examined, and innovative methods are designed. Based on the results of the user studies [144, 146], the most relevant similarity aspect *required action* is examined to be used as a method to determine the similarity of tasks for task recommendation [143]. A scenario in a location-based micro-task market is analyzed in detail to provide a crowd selection

approach for task recommendation in Section 5.3. Skill extraction from employee profiles and project descriptions is provided for the recommendation of project assignments in Section 5.4. For the recommendation of job postings in a job market platform, an approach is provided, which enables the preselection of job postings enhancing the performance of state-of-the-art recommender systems [145].

The fast developing field of recommender systems provides the possibility to come up with new foundational methods for providing recommendations, e.g., based on the concepts of collaborative filtering and *matrix factorization*. While this is an interesting field, the scope of this thesis is focused on the characteristics of recommending tasks based on heterogeneous textual descriptions. Therefore, it is not a goal to develop ground-breaking recommender systems, but to identify, analyze and describe methods for recommendation, that provide meaningful results for this selected area of interest.

#### 1.4 STRUCTURE OF THE THESIS

The thesis is structured as follows. Chapter 2 provides the technical background for methods that are used as foundations within the developed methods that are provided in the later chapters. Chapter 3 provides a first contribution, where the requirements towards an ECP are gathered and used to model task assignment strategies and other aspects of ECPs in general. Chapter 4 provides another contribution, where the perspective of workers is captured in order to understand the similarity aspects of micro-tasks. Chapter 5 provides different methods of recommending tasks to workers based on the different kinds of textual descriptions provided by micro-tasks, projects, and job postings. Chapter 3, 4 and 5 are motivated by very different approaches in the literature. For a better understanding, the respective related work is provided at the beginning of each of these chapters. The thesis is concluded in Chapter 6 with a brief summary of the core contributions. Finally, an outlook on potential future work is provided.

THIS chapter provides the foundations for the thesis at hand. First, fundamental crowdsourcing principles are discussed in Section 2.1, and different task distribution platforms are described in Section 2.2. Task recommendation approaches are discussed in general and for each scenario from micro-tasks to job postings in Section 2.3. Fundamental methods of Natural Language Processing (NLP) for text processing, representation and similarity detection are discussed in Section 2.4. Machine Learning (ML) approaches for clustering and classification are introduced in Section 2.5, while corresponding evaluation metrics are discussed in Section 2.6 respectively.

## 2.1 CROWDSOURCING TASKS

Jeff Howe [74] introduced the term *crowdsourcing* in 2006 in an article for the *Wired Magazine*. It forms a portmanteau of the two terms *crowd* and *outsourcing*. In general, it describes the outsourcing of work to a more or less anonymous crowd over the Internet. In the literature, crowdsourcing is often defined very broadly and may refer to the general concept as defined by Brabham [25]: “an online, distributed problem-solving and production model that leverages the collective intelligence of online communities to serve specific organizational goals.” This definition can be understood to also include work on *Wikipedia* and *open source* projects in general.

Geiger et al. [52] follow such a general definition and provide the respective visualization (see Figure 1). Here, the *crowdsourcing organization* pursues a specific goal and uses the *crowdsourcing process* to realize it. Within the crowdsourcing process, *contributions* are sourced by distributing them to the *crowd contributors*. Aggregating the contributions provided by the crowd realizes the initial goal of the crowdsourcing organization.

However, the term *crowdsourcing* is sometimes defined in a more specific manner, as described by Yuen et al. [185]: “Crowdsourcing is outsourcing a task to a large

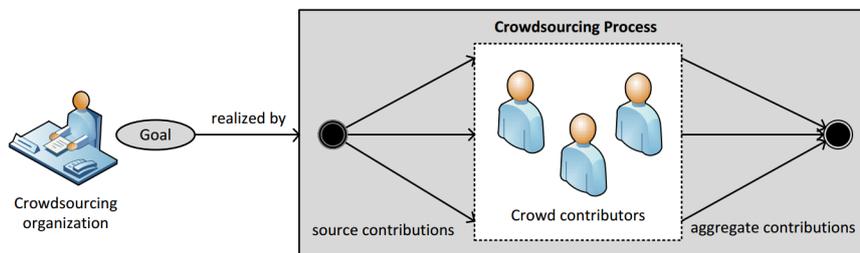


Figure 1: The general crowdsourcing process as provided by Geiger et al. [52].

group of networked people in the form of an open call to reduce the production cost.” This definition is very specific on key elements like naming a *task* as a central element, including the assignment strategy of an *open call* and focusing on the goal of *reducing cost*. This definition is more closely related to the concepts described by Howe [75], who describes an open call task assignment strategy as “self-identification of contributors.”

To distinguish different crowdsourcing processes, Geiger and Schader [51] introduce four different archetypes. They define their archetypes depending on how value is derived from contributions and how value is differentiated between contributions, which is shown in their visualization in Figure 2. Their categorization allows to distinguish between the aggregating *crowd rating* process (e.g., *IMDB*<sup>1</sup>), the collaborative *crowd creation* process (e.g., *Wikipedia*, *open source* projects), the accumulative *crowd solving* process (e.g., question answering portals) and the homogeneous *crowd processing* process.

An example of a *crowd processing* platform is often given by referring to *Amazon Mechanical Turk (MTurk)*. Platforms like *MTurk* allow publishing tasks to a huge managed community of contributors. They provide functions for publishing tasks, managing contributions, and transferring remunerations. Geiger and Schader [51] refer to such platforms as *marketplace*. Since the offered tasks can often be completed in a very short time, the tasks are described as micro-tasks. The respective platforms are also known as micro-task markets [86], which are discussed in the following.

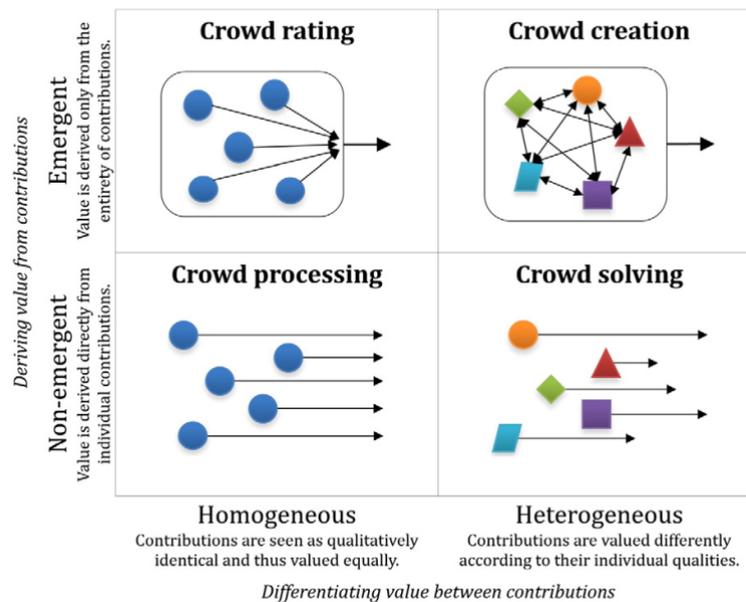


Figure 2: The four crowdsourcing archetypes, as provided by Geiger and Schader [51].

<sup>1</sup> www.imdb.com

## 2.2 DIFFERENT KINDS OF TASK MARKETS

This section introduces the different kinds of task markets. First, micro-task markets are described and analyzed. Afterward, the additional scenarios focused in this thesis are described and analyzed separately.

### 2.2.1 Micro-Task Markets

In micro-task markets such as *MTurk*, “anyone can post tasks to be completed and specify prices paid for completing them” [86]. Initially, such platforms were designed to have humans complete simple micro-tasks that are very difficult for computers to solve, relating crowdsourcing to the field of *human computation* [127]. The respective micro-tasks are also referred to as Human Intelligence Tasks (HITs). The results of the completed micro-tasks provide the training data for Machine Learning algorithms which learn to solve such tasks in the future. This methodology is still applied for tasks such as image tagging, named entity annotation or text translation [117]. From a contributor perspective, who is paid for individual micro-tasks, micro-task markets appear as non-emergent and homogeneous and therefore represent *crowd processing* platforms, following the categorization of Geiger and Schader [51]. However, it is possible to distribute micro-tasks that fall into different crowdsourcing categories, offering remunerations for adding a *rating* on Amazon or *creating* a particular *Wikipedia* article. Micro-task markets therefore follow *crowd processing* concepts but provide the possibility to distribute crowdsourced micro-tasks in general.

The structure of a micro-task market is illustrated in Figure 3. In micro-task markets, the crowdsourcing organization is referred to as the micro-task *requester*, and a contributor is referred to as the crowd *worker*. The *platform* provides the *micro-task market*, which publishes the *micro-tasks* and manages the *assignment* of workers to micro-tasks. The requester offers a micro-task on the micro-task market and therefore requests its completion, which is provided by a worker. Thereby, the worker provides his workforce and receives a remuneration, which may also be managed by the micro-task market.

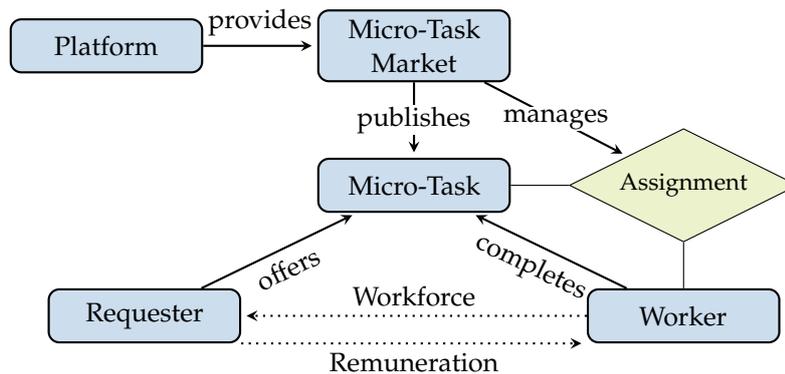


Figure 3: The general structure of a micro-task market.

### 2.2.2 *Public and Enterprise Crowdsourcing Platforms*

The term Public Crowdsourcing Platform (PCP) describes any crowdsourcing platform which publishes tasks publicly on the Internet. The term Enterprise Crowdsourcing Platform (ECP), on the other hand, describes crowdsourcing platforms which publish tasks in a more closed environment, making the tasks available to existing employees of the enterprise only [64]. Besides the abstract structure of micro-task markets described above, PCPs and ECPs provide various functions for workers and requesters. The platforms usually manage the remuneration of workers [34], provide elaborate methods to manage the different kinds of contributions [138], and allow requesters to “form user groups” [63]. A detailed analysis of the elements, components, and functions of ECPs in comparison to PCPs is provided in Chapter 3. Focusing on task recommendation, especially the functions involved in the task assignment process are concerned.

### 2.2.3 *Location-Based Crowdsourcing Platforms*

In *location-based* micro-task markets, a worker has to be present at a certain location in the physical world to perform a task [3]. Such tasks often require the worker to provide information in the form of pictures of a certain location, for example taking a picture of a given speed sign [129]. In such location-based micro-task markets, tasks are more scarce than in general micro-task markets and often require the workers to use their smartphones. Within such platforms, the location of tasks is the main factor for workers to choose tasks. Workers rather expect notifications about new tasks in their area, than actively searching on the platform [129]. Therefore, crowd selection methodologies can be applied to identify workers who should get a recommendation when a new task is published on the platform.

### 2.2.4 *Project Staffing*

Within the spectrum of tasks, ranging from micro-tasks to job postings, enterprises also face the challenge of *project staffing* [12]. In many contexts, projects provide a way to structure workforce, where usually one or more people work together in a project team to accomplish a common goal. In projects and teams, employees with different skills fill different positions that complement each other [89, 163]. One challenge is to find employees who are qualified, and therefore meet the projects’ requirements [124], to support the worker selection for project staffing.

### 2.2.5 Job Market Platforms

Various online job exchanges (e.g., *Stepstone*<sup>2</sup>), job-oriented social networks (e.g., *Xing*<sup>3</sup>) and job search engines (e.g., *kimeta*<sup>4</sup>) have been established in the World Wide Web. They aim to offer job-seekers the jobs that best suit their interests. For example, the job search engine *kimeta* finds more than 2.3 million job postings in November 2018 [85], which emphasizes the complexity of such a domain-specific search engine. On such platforms, job seekers can search for jobs from a large variety of sources. Additionally, users on these platforms can search for jobs and filter by certain criteria [147]. Whether users actually apply for jobs and whether the application is successful, usually remains unknown for the job market platform. Therefore, job market platforms can often only analyze the click behavior of users on their platform.

## 2.3 TASK RECOMMENDATION

This section introduces the fundamentals of recommender systems in Section 2.3.1 and discusses issues related to recommendations in task markets in general in Section 2.3.2. The specifics of task recommendation in micro-task markets are discussed in Section 2.3.3, while recommendation in further task markets is discussed in Section 2.3.4.

### 2.3.1 Recommender Systems

In general, recommender systems “are software tools and techniques providing suggestions for items to be of use for a user” [131]. These systems usually focus on specific kinds of items (e.g., products, music, or news) to be suggested for users to interact with accordingly (buy, listen, read) on the respective platform. Recommender systems exploit the data that is given on the platform and produced by the interactions of users and items. Items are the objects to be recommended and can come with many different features (e.g., textual descriptions) which can be used for recommendation. Users of recommender systems represent real persons that expect to find matching items faster and easier.

Interactions may provide different outcomes, depending on whether they are explicit ratings or implicit interactions. An explicit rating of a user towards an item can be provided, e.g., as a numerical value such as a five-star rating or a binary value deciding whether an item is liked or not. Implicit ratings can be produced by interactions on the platform and are often described as unary ratings, e.g., an item was observed on the platform, clicked on, or bought. This kind of interaction does not provide negative relations between user and items. There is a trade-off between these two kinds of feedback. In the case of implicit feedback, a large amount of data is available, which, however, includes some uncertainty about the user’s intentions.

---

<sup>2</sup> [www.stepstone.com](http://www.stepstone.com)

<sup>3</sup> [www.xing.com](http://www.xing.com)

<sup>4</sup> [www.kimeta.de](http://www.kimeta.de)

Implicit feedback does not clearly indicate whether a user prefers an object or not. In contrast, explicit user ratings are more scarce but provide reliable information about the user's preferences [119].

In order to provide recommendations, recommender systems try to predict the utility of an item for a user, based on the information given on the item, user, and the history of user interactions. The utility of an item  $d \in D$  for a user  $u \in U$  is modeled as a function  $R(u, d)$ , and the task of the recommender system is to compute the estimation  $\hat{R}(u, d)$  of the true function  $R$ . Based on the predicted utility of all items for a user  $\hat{R}(u, d_1), \dots, \hat{R}(u, d_N)$ , the recommender system recommends items  $d_{j_1}, \dots, d_{j_K}$  ( $K \leq N$ ) with the largest predicted utility for user  $u$  [131].

The number of recommended items  $K$  is typically a small number. Therefore, a recommender system provides for each user a list of ranked items  $(d_{j_1}, \dots, d_{j_N})$  from an ordered set of items  $\tilde{R} \subseteq D \times D$ . This recommendation set  $\tilde{R}$  for user  $u$  contains tuples of all items  $d_i, d_j \in D$ , following the constraint of order, depending on their estimated utility  $\hat{R}$ :

$$\forall (d_i, d_j) \in \tilde{R} : \hat{R}(u, d_i) \geq \hat{R}(u, d_j) \quad (1)$$

According to Ricci et al. [131], the three main types of recommender mechanisms are content-based recommender, collaborative filtering recommender, and hybrid approaches. With content-based filtering, the user profile is determined based on the user's interactions with the items, and on information about the items (content). Alternatively, a manually created user profile may exist, and a recommender compares the contents of this created user profile and item information. In contrast, collaborative filtering provides the user with a recommendation of items that other users with similar interests interacted with. Therefore, methods of collaborative filtering do not require any additional information about users and items despite their interactions.

### 2.3.2 Task Recommendation in Task Markets

As discussed before, the selection of tasks for workers is a central element of task markets in general. A recommender system supporting this selection therefore recommends tasks to workers. In recommender systems for tasks, workers are considered the users of the recommender system, and tasks are considered the items of the recommender system.

Within task markets, there is specific information a recommender system can exploit to predict the utility of tasks for workers. When workers select tasks, they provide interactions which can be leveraged by a recommender system. In task markets, the history of selected or completed tasks can be used as the history of interactions for a recommender system. This interaction may provide unary data, where a worker either submitted a task or not. Implicit ratings may also be inferred from the interactions, where a worker submitted a task, and the contribution was accepted, rejected or had to be revised for acceptance.

Task markets suffer from the problem of high *churn*, where "items come and go rapidly" [26]. In such domains, items accumulate ratings but may be irrelevant for

recommendation after a short duration. Burke and Ramezani [26] provide the domain of news items as an example, where items may become irrelevant within a few days. Task markets suffer from this problem in general, due to the dynamic nature of task markets. This means that the first interaction with a task on the platform completes the task. A completed task on the platform cannot be completed by another worker. Therefore, tasks cannot accumulate more than one interaction in such a scenario, before they become unavailable for recommendation.

Many recommenders rely on collaborative schemes, where items are recommended due to several interactions of the users. The classical collaborative approach can only recommend tasks that have been interacted with. As such tasks appear as completed in task markets, they are not available on the platform anymore, and cannot be recommended.

A recommender system for task markets has to find non-interacted and therefore available tasks on the platform, based on the completed tasks of a worker's history. Content-based recommender systems can find similarities between completed and available tasks for the workers to get recommended. Using task similarities enables the recommendation of similar tasks with respect to the worker's history, which are available on the platform. Similarities between tasks can also help to identify similar workers, where two workers did not complete the same, but similar tasks. Some approaches therefore rely on coarse task categories which represent a form of task similarity, to enable task recommendation [34, 35, 183, 185].

An alternative to the use of task categories is to use the textual description of tasks and to define measures and methods to calculate the similarity of tasks based on the textual descriptions. Becker et al. [14] show, that semantic content of tasks (task descriptions) is a decisive factor for workers to choose tasks. Instead of using the categories defined by the requesters or the platforms, similarities based on textual descriptions of tasks may allow a more detailed view on the semantic content of tasks to be exploited for recommendations. The approaches provided in this thesis follow this idea.

When focusing on textual descriptions for the recommendation of tasks, the task descriptions can be viewed as documents. These documents represent the tasks in terms of the given text. Therefore, methods for the classification and clustering of documents using ML and NLP are relevant for the approaches in this thesis. Thus, such methods are discussed in this foundations chapter in Section 2.4 and Section 2.5.

### 2.3.3 *Task Recommendation in Micro-Task Markets*

Most of the micro-task markets rely on the "self-identification of contributors" [75] and follow an "open call" approach [183]. Geiger and Schader [51] go as far as to declare that "all crowdsourcing systems are built on the same basic principle of contributor self-selection." As already motivated in the introduction, the self-selection leads to a non-optimal assignment of tasks which can be enhanced by introducing task recommendation.

Micro-task markets have the problem of the dynamic nature of task markets, where tasks become unavailable after their first interaction (completion). In many micro-task markets, so-called *campaigns* offer almost the same task to many different workers at the same time. These campaigns can be used by a collaborative recommender system; however, such campaigns are also completed in the matter of hours or days [68].

When requesters offer their micro-tasks on the micro-task market, as shown in Figure 4, they provide a textual task description, among other criteria like the offered payment, the expected working time, and more. Based on the task descriptions, a similarity between micro-tasks can be calculated in order to find matching tasks for the worker. As discussed before, the task assignment in micro-task markets is often performed by the worker who searches for tasks and selects them for completion. Instead, the micro-task market can leverage the history of the worker’s assignments, analyze the respective task descriptions, find similarities between tasks, and recommend them to the worker. Therefore, from the conceptual viewpoint of task recommendation presented in Figure 4, the micro-task market is able to *recommend* assignments for the worker, based on the *similarities* found in the given textual *descriptions* of micro-tasks.

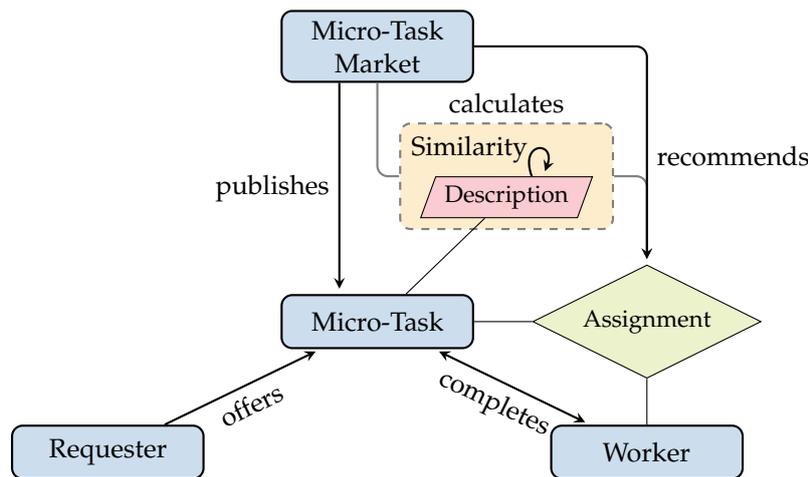


Figure 4: Task Recommendation in micro-task markets.

### 2.3.4 Task Recommendation in other Task Markets

Besides micro-task markets, this thesis also focuses on task recommendation in task markets for *projects* and *jobs*. Therefore, a generic view on task markets is provided in Figure 5. When it comes to projects or jobs, the offered tasks are not directly assigned.

In a *project staffing* scenario, employers search for employees to work on projects. Therefore, employers can analyze their project proposals and find similarities with completed projects of their employees. For project assignments, workers can be se-

lected for a project to work on, for example by matching the project requirements and skills of available employees.

The users of *job market platform* (possible future employees), select jobs of employers to apply to. The employers offer their jobs with a description, and users of the platform search for matching job postings on the job market. To optimize the search, the job market can offer recommendations for the selection of tasks to the workers.

In a more general description of task recommendation in task markets given in Figure 5, a *task provider* offers *tasks* with their *descriptions* on the *task market*. A *task contributor* searches for matching tasks or is searched to contribute on matching tasks. In either way, the task market is able to perform the function of recommendation based on the similarities of the task descriptions. Throughout the thesis, the termi-

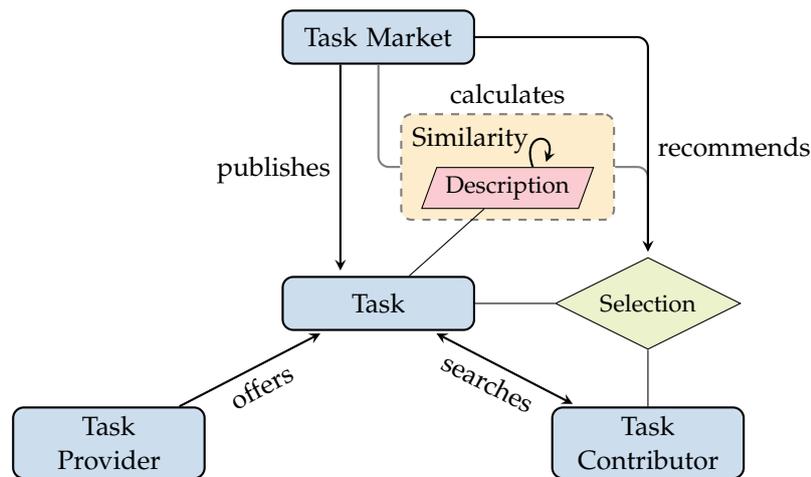


Figure 5: Task recommendation in task markets.

nology for *task provider*, *task*, *task seeker* and *task distribution platform* are adapted to the respective crowdsourcing scenario at hand. The terms *requester*, *worker*, *task* and *task descriptions* are used as generic terms. For the domains of enterprise crowdsourcing, project and job recommendation, the corresponding terms from the scenarios like *employer* and *employee* as well as *job posting* and *project description* are used. How the different focused domains relate to this general task recommendation scenario is discussed in the following.

The recommendation of tasks to workers in *location-based crowdsourcing platforms* is highly dependent on the location of task and worker. This is due to the fact, that workers expect to get a recommendation (“push” notification) when matching tasks come up in their area, instead of getting recommendations when they search on the platform (“pull strategies”) [35]. Instead of recommending tasks to workers, this scenario requires to identify workers suitable for a task to allow “job alerts” [129]. Such a recommendation scenario is often referred to as *crowd selection* [188].

In the scenario of *project staffing*, project descriptions and employee profiles are available. Following the methodology for recommendation in task markets, descriptions of completed projects can be used to find matching workers for new projects. Regarding the viewpoint of the employer, not only project selection preferences of

the employee should be taken into account for project assignments, but also the qualifications of the worker. Modeling the skills and qualification of the employees is an approach to support the worker selection and provide recommendations.

Considering an online *job market platform* as discussed before, a job seeker browses the job postings on the platform. Following the methodology for recommendation in task markets, such a job market can use the visited job postings as the history of the job-seeker in order to identify job postings for recommendation. As shown in Section 2.2.5 the number of available job postings can be very high. Therefore, the job-seeker requires support in finding matching jobs [140]. Accordingly, a recommender system can support the job-seeker, depending on his or her preferences [125].

## 2.4 NATURAL LANGUAGE PROCESSING

The goal of Natural Language Processing (NLP) is to process speech, text, and languages. Classical NLP methods for text processing extract information from given texts describing the syntactic and semantic meaning of words, resulting in Part-of-Speech tags or Named Entity Recognition. For the purpose of information retrieval, statistical methods of text mining count word occurrences in documents, which is often applied for search tasks. Machine Learning (ML) approaches for the classification and clustering of documents rely on such extracted features. Recent developments have changed the state-of-the-art NLP possibilities. Especially the field of advanced ML, applying deep neural networks [31], and providing neural word embeddings for such methods as *Word2Vec* [113] provide record-setting performances. The fundamental methods and advanced methods are introduced briefly in the following sections.

### 2.4.1 Text Processing and Annotations

In today's NLP pipelines, text processing methods are used to generate features for later applied statistical methods, classification tasks or clustering tasks [104]. This is often also referred to as text preprocessing and is broadly understood as cleaning the data of natural language especially in unstructured texts. For the subsequent steps of text processing, this often reduces the feature space by applying, e.g., tokenization and normalization.

#### 2.4.1.1 Text Tokenization and Normalization

Tokenization is the task of "segmenting running text into words" [81] and divides a text into so-called *tokens*, which usually represent single words and punctuation. In text normalization, these tokens are put into a standardized format. Usually, different tokens that carry the same semantic meaning are projected onto the same token. A conventional method of normalization is *case folding*, converting all letters to lowercase format. For example, the token 'Darmstadt' is transformed into the token 'darmstadt'. Another method often applied is the elimination of so-called *stop*

*words*. These are words that appear regularly in almost all documents, e.g., ‘the’ and ‘a’. These words have little informative value and are irrelevant for a classification based on word occurrences. A further kind of normalization is *lemmatization* where deviating grammatical forms of a word are converted into the basic form. For example, ‘programming’ and ‘programmed’ are transformed into ‘program’, which leads to a clean conversion in the form of keywords from dictionaries. In contrast to lemmatization, *stemming* leads to a grammar-independent conversion on the basis of certain rules. For example, the words ‘programming’ and ‘programmatically’ are transformed into ‘programm’. Tokens usually represent single words. It is often required to also identify meaningful *terms*. A term may represent single tokens (e.g., ‘Darmstadt’), whole words (‘TU-Darmstadt’) or multi-word expressions (‘Technische Universität Darmstadt’). Such terms are often identified using the methods of Part-of-Speech tagging, chunking and Named Entity Recognition described below.

#### 2.4.1.2 *Part-of-Speech Tagging and Chunking*

With Part-of-Speech (PoS) tags each token is assigned a part of a sentence (e.g. noun, verb, etc.) [81]. The information whether a token is a noun or a verb influences the probability of the surrounding words, e.g., nouns often follow adjectives or verbs follow nouns. This information can help to extract important information, such as people or organizations, from texts. PoS tags can be included as features in a classification process since certain sequences of tags can be typical for certain document categories [110]. By filtering for specific PoS tags, this method can also be used to reduce the feature space, by focusing, e.g., on nouns [2, 108]. Some most common tags from the Penn treebank tag set [107] are given in Table 27 in the Appendix.

Chunking describes the recognition and annotation of larger coherent multi-token structures. These *chunks* combine a string of tokens, e.g., noun or verb phrases. A noun phrase, for example, contains a word or a group of words which function as a subject in a sentence, e.g., the phrase ‘My university in Darmstadt ...’ provides a noun phrase about the subject ‘university.’ Such noun phrases are useful to identify, e.g., persons, locations or other entities in Named Entity Recognition.

#### 2.4.1.3 *Named Entity Recognition and Named Entity Normalization*

“Named entities are words or phrases which are named or categorized in a certain topic” [115]. Named Entity Recognition (NER) uses chunks to represent entities and label them with a type like person, location, date, and currency. For NER exist many different methods, from using lists of known words to employing neural networks and word embeddings [93].

After determining the type of an entity, it is also possible to assign a name to the entity. Terms that refer to the same entity may look different (e.g., ‘TUD’ and ‘TU Darmstadt’). Therefore, Named Entity Normalization (NEN) finds different representations and links them to the same entity. For task recommendation, the named

entities which are of most interest are qualifications or skills which tasks require or workers provide.

#### 2.4.2 Word and Document Representations: The Vector Space Model

Text mining is applied to transform textual information into numerical vectors for the application of data mining methods [177]. For statistical models, for example, each term found within a document may be used as a feature to represent this document. A document can therefore be represented by a list of all terms contained in this document. For comparing several documents in a corpus of documents, the combined vocabulary of terms found in the documents is used to represent a single document. Representing the document as a vector in the size of the vocabulary, the values point to whether a term appears in the document or not. This so-called Bag-of-Words (BoW) model reflects the occurrences of each word or term in a document [81]. A vector space representation of documents, for example, may comprise of a matrix where such document vectors represent one document in each a row, and each column represents a term of the vocabulary [104].

Instead of binary word occurrence, it is possible to count the Term Frequency (TF) of terms in a document. In a formal model, where  $t \in T$  is a term from the set of all terms and  $d \in D$  is a document from the set of all documents, the  $tf$  function determines the frequency of a term in a document:

$$tf(t, d) \quad \text{with} \quad tf : T \times D \rightarrow \mathbb{N} \quad (2)$$

The idea behind this is that terms that occur more often in a document are also more relevant to represent this document. Even after removing stop-words from the vocabulary, this is not necessarily true. For example, texts gathered from the website of TU Darmstadt may all contain the term 'TU Darmstadt' in very frequent manners.

To judge on the relevancy of a word for a specific document, the TF of a term in this document has to be related to the frequency a term occurs in the whole corpus of given documents. Therefore, the Document Frequency (DF) counts the number of documents in which a token appears in. The  $df$  function determines how many documents contain a particular token:

$$df(t, D) \quad \text{with} \quad df : T \times \mathcal{P}(D) \rightarrow \mathbb{N} \quad (3)$$

The Term Frequency – Inverse Document Frequency (TF-IDF) representation [154] of documents relates the TF in the document with the DF in the corpus for each term: The  $tfidf$  function relates the TF to the inverse of the DF and provides a measure for the relevancy of a certain term for a certain document.

$$tfidf(t, d, D) = tf(t, d) \cdot \log \frac{|D|}{df(t, D)} \quad (4)$$

The TF-IDF is often used for keyword extraction.

A sequence of tokens, so-called *n-grams*, is also often used as a form of vocabulary. Instead of using single tokens (*unigrams*), two tokens (*bigrams*), or three tokens (*trigrams*), can be used as features. For example, the unigram vocabulary of 'My uni in

DA' contains the tokens {'my', 'uni', 'in', 'da'}, while a bigram vocabulary contains the terms 'my uni', 'uni in', 'in da.' This allows to value the context of words appearing together. Often unigram, bigram and trigram representations are used in a combined vocabulary respectively feature space. Some methods use character *n-grams* instead of word tokens.

Using such functions, a document is converted into a high dimensional vector space with at least the size of the vocabulary of the overall document corpus. The instance vectors of the documents are very sparse, as a single document only contains a tiny fraction of possible terms. To determine the similarity of documents, many algorithms for classification and clustering can work directly on this feature space but would perform faster and better in a vector space with fewer dimensions and dense vectors. Text processing is often used to reduce this feature space as described before. Techniques like *matrix factorization* and *singular value decomposition* can also be applied to reduce the given vector space matrix to fewer dimensions [55]. The methods of matrix factorization and singular value decomposition are also applied in collaborative recommender methods to reduce the size of the user-item matrix [90], and predict the missing values for ratings in the matrix.

In these word occurrence models, the position of the word in the text is ignored, and semantic meaning might be lost. Therefore, word co-occurrence models describe how often words occur together (in the same context), following the distributional hypothesis that "words that are used and occur in the same contexts tend to purport similar meanings" [62]. Distributional semantics are often used to define the similarity of words and therefore also of texts. Alternative features for the representation of words and documents in a vector space are derived very differently using various measures for text similarity, as discussed in the next section.

### 2.4.3 Text Similarity

There exist plenty of different similarity measures and applications for text similarity [11]. Goma and Fahmy [55] provide an overview of the most common similarity measures. They categorize into string-based, corpus-based and knowledge-based similarities. String-based or also term-based text similarity measures measure the distance between two text strings, "for approximate string matching comparison" [55]. In this category fall, besides others, the *Levenshtein distance*, the *Jaccard similarity*, but also the *Euclidean similarity* and the *Cosine similarity* [55]. The *Cosine similarity* is commonly used in a vector space representation of documents. It "is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them" [55].

Corpus-based similarity measures identify text similarities in relation to the corpus of documents the text appears in. A very popular corpus-based similarity measure is the *Latent Semantic Analysis (LSA)* [94]. Based on a matrix representation of word counts per paragraph (one row per word, one column for each paragraph in the corpus), singular value decomposition is applied to reduce that feature space (number of columns), and the cosine of the angle between the resulting word vectors (rows)

defines the similarity of these words. The *Explicit Semantic Analysis (ESA)* [49] measures the similarity between two arbitrary texts by relating terms or text to *Wikipedia* articles. The terms are represented by their TF-IDF value for the respective article on *Wikipedia* which describes how good this term functions as a keyword for the article. The cosine measure between these vectors defines the similarity. This approach can also be used in cross-lingual recommendation scenarios [141].

Knowledge-based similarity measures identify the degree of similarity of words based on knowledge from semantic networks [112]. Such a semantic network is provided by *WordNet* [114], which provides six measures of semantic similarity and three measures of relatedness [120]. In *WordNet*, “nouns, verbs, adjectives, and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept” [55]. These synsets are interlinked depending on their semantic and lexical relations. Three of the semantic similarity measures are based on the path length between interlinked words. The *path* measure returns the length of the shortest path based on the hypernym/hyponym (is-a) taxonomy relation. The *lch* measure [96] relates this shortest path to the maximum path length of the hierarchy they appear in. The *wup* measure [181] finds the closest shared ancestor and relates the path length between this ancestor and the root of the hierarchy to the path length between the synsets and the root.

So-called *word embeddings* [99, 113] learn low dimensional representations of words and documents from distributional semantics applying neural networks. One popular method is the *Word2Vec* model proposed by Mikolov et al. [113]. There are two different approaches of *Word2Vec*, which learn to predict words by their context or vice versa. The *Continuous Bag of Words (CBOW)* model predicts the occurrence of a word based on its context. The *Skip-Gram* model learns to predict the context based on a given word. These models can, for example, create a vector space of semantic meaning, where vector operations traversing the feature space reflect corresponding semantic relations. For example, when executing the operation  $\text{vec}(\text{King}) - \text{vec}(\text{Man}) + \text{vec}(\text{Woman})$ , the resulting vector is closest to the  $\text{vec}(\text{Queen})$ .

Other word embeddings methods are provided by *GloVe* and *FastText*. *GloVe* [122] is a word embedding project by the Stanford University, who provide pre-trained word embeddings on huge corpora from web documents [54]. *FastText* [21] is an embedding method which is based on a “*Skip-Gram* model, where each word is represented as a bag of character *n-grams*”. For *FastText* there are also pre-trained word vectors available [47].

These word vectors can also be used to assign a vector to each document. For example, in *word vector averaging* [153], a document vector is calculated by averaging all word vectors of the corresponding document. The *Paragraph Vector* method of Le and Mikolov [95], also referred to as *Doc2Vec*, is based on the ideas of *Word2Vec*, projecting entire documents instead of just words onto a vector. The *Doc2Vec* approach provides two different methods, the *distributed memory*, and the *distributed bag-of-words* model. The *distributed memory* model of *Doc2Vec* introduces a document as additional context to predict the occurrence of a word. The *distributed bag-of-words*

model does not consider the order of words and according to Le and Mikolov [95] delivers worse results than the *distributed memory* model.

## 2.5 CLUSTERING AND CLASSIFICATION USING MACHINE LEARNING

Machine Learning (ML) applies statistical methods to recognize patterns in data and to derive decisions. ML methods include supervised approaches that learn from example data where the decisions are known (ground truth). Such approaches are used for classification, where certain items are judged to belong to a particular class or not. This can be used to decide whether a document belongs to a particular category, or if a document is of relevance for a user. Unsupervised approaches, on the other hand, find patterns and order the objects accordingly, without a given ground truth. This can be used to cluster a number of documents in respect to a similarity measure. Further methods of ML include regression, where the value of a continuous variable is predicted, and reinforcement learning, where the decisions are found on few data at first, and the algorithm adapts the decisions iteratively to maximize a certain reward function. In this thesis, mainly classification and clustering approaches are regarded.

### 2.5.1 Clustering Methods

Clustering groups items on the basis of their properties. It tries to group similar items into a so-called cluster. Dissimilar items should be located in different clusters. Items are usually represented as numerical vectors, and a certain similarity measure operating on this vector space is chosen. There are different approaches for clustering such as *k-means* and *mini-batch k-means* clustering described below.

The goal of the *k-means* algorithm is to minimize the distances between all items and their nearest center [48]. Therefore the items  $d \in D$  are to be grouped into a predefined number of  $k$  clusters. At the beginning,  $k$  random points (the “means”) are selected, and each item is assigned to the closest mean. Then, in several iterations, cluster centers (the “centroids”) are calculated from the items within the cluster. These calculated centroids become the new means for the next iteration. The complexity of the *k-means* algorithm is  $\mathcal{O}(j k |D|)$ , where  $j$  is the number of iterations.

The *mini-batch k-means* algorithm [149] is based on the *k-means* algorithm and provides an adaptation when handling large amounts of data. Therefore, a subset  $B$  (mini-batch) with  $B \subseteq D$  is randomly selected from the entire dataset in each iteration. The centroids are updated in each iteration based on the currently selected set of documents  $B$ . Therefore, the complexity is reduced from  $\mathcal{O}(j k |D|)$  to  $\mathcal{O}(j k |B|)$ .

A disadvantage of these algorithms is that the number of clusters  $k$  must be selected. One possibility is to execute clustering with different  $k$ s and determine the best value  $\phi_k$  in an evaluation. The *Silhouette coefficient* [135] provides a metric for evaluating the results of clustering. Other methods like *DBSCAN* [45] and *HDBSCAN* [27] determine the required number of clusters automatically.

### 2.5.2 Classification Methods

Classification can be based on different models described below.

K-Nearest-Neighbor (KNN) classifiers work on the vector space and a given similarity or distance measure [179]. A newly observed instance is classified based on the classification for the nearest neighbors found. Depending on the classification of the majority of the  $k$  neighbors, the new instance is classified.

Bayesian classifiers rely on modeling the probability of objects belonging to a certain class based on the Bayesian theorem [18]. Naive Bayes classifiers naively assume the independence of variables. Therefore, they do not perform as good in complex classification tasks [179].

Support Vector Machines (SVMs) [80] are supervised learning models used for classification and regression. They train a representation of a hyperplane, which divides the vector space two classes, based on the given training examples. The distance of the hyperplane towards the vectors of the classes is maximized in training. As SVMs have a high classification quality, especially in high dimensional and sparse vector spaces, they perform well in text classification problems [140].

An artificial neural network consists of several layers with connected artificial neurons. These aggregate an input signal and forward a signal based on a threshold or aggregation function to the next layer of artificial neurons [19]. The number of layers, neurons per layer and the aggregation function has to be chosen to optimize classification behavior for each focused problem [179]. Artificial neural networks show a good performance to find similarities between words with word embeddings [113]. They can also be learned to optimize a hyperplane for classification [174].

## 2.6 EVALUATION METRICS

There are different evaluation metrics to evaluate classification and recommendation approaches. In these methods, the decision found by the algorithm or Machine Learning (ML) approach is evaluated against the previously known correct decisions. Therefore, a given corpus is split into a training part, on which the algorithm is trained, and a testing part, which is used for evaluation. The evaluation metrics are based on the comparison of the actual class (positive or negative) of items and how the algorithm classified the item. If the algorithm classified an item as belonging to a class (positive), this decision can be correct (true positive (TP)) or incorrect (false positive (FP)). If the algorithm classified an item not to belong to a class (negative), this decision can be correct (true negative (TN)) or incorrect (false negative (FN)). In an ideal case, there are no falsely classified instances with FP and FN are zero. These discrete values are related against each other, and against the number of test instance  $n_{\text{test}}$ , to derive evaluation metrics as shown below.

The accuracy describes how many instances are classified correctly (true positives and true negatives), depending on the number of evaluated instances:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{n_{\text{test}}} \quad (5)$$

The *Precision* of a classifier shows how many positively classified instances (TP + FP) are actually positive (TP). The *Precision* therefore gives an impression of how high the false positive rate of the classifier is.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (6)$$

The *Recall* of a classifier shows how many actually positive instances have been classified as positive.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (7)$$

Depending on the scenario, a high *Precision* or a high *Recall* may be of interest. Focusing on a high recall means, that the classifier may produce many false positives, as long as all actually positives are found. If all items are classified as positive, the *Recall* is 1, but the *Precision* is very low. Focusing on a high *Precision* means, that rather few positives are accepted, as long as they are truly positive. If only a single instance is classified truly positive, the *Precision* is 1 but *Recall* is very low. Therefore, these metrics are often combined to define a harmonic mean between *Precision* and *Recall*, the *F1* measure:

$$\text{F1} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (8)$$

In a recommendation scenario, usually, a high *Precision* is of interested, where only a few items can be recommended to a user. Recommendation approaches usually provide a continuous ranking from preferred items to less preferred items, to allow a recommendation of only the first 5, 10, or top-k items. Therefore, the evaluation metrics are adapted, measuring the *Precision* or *Recall* of the first k ranked items, resulting in *Precision@k* and *Recall@k*.

For recommending items  $d \in D$  to a user  $u \in U$ , consider  $(d_1, d_2, \dots, d_m)$  a ranked list of items from an ordered set of predicted utility for recommendation. For the evaluation of a recommender system, a correct recommendation includes an item that has been found to be actually relevant to a user in the given ground truth. Therefore, the function *rlv* determines whether a document is relevant for a user or not:

$$\text{rlv}(u, d) \quad \text{with} \quad \text{rlv} : U \times D \rightarrow \{0, 1\} \quad (9)$$

The *Precision@k* indicates the percentage of the top-k predicted items which are relevant for a user:

$$\text{Precision}(k) = \frac{1}{k} \sum_{j=1}^k \text{rlv}(u, d_j) \quad (10)$$

To measure the *Precision@k* for a recommender system, the *Precision@k* values are averaged over all users. A disadvantage of the *Precision@k* is that if the number of

relevant items  $k_u$  for a user is less than the chosen  $k$ , it can never reach the optimal value 1. This can also lower the averaged *Precision@k* for the whole system, where only for some users  $k_u$  is less than  $k$ . To account for this problem of *Precision@k*, *R-Precision* takes for each user only the number of known relevant items  $k_u$  into account:

$$\text{R-Precision} = \frac{1}{|\mathcal{U}|} \sum_{i=1}^{|\mathcal{U}|} \frac{1}{k_u} \sum_{j=1}^{k_u} \text{rlv}(u, d_j) \quad (11)$$

In contrast to this, *Recall@k* indicates the percentage of all known relevant items  $k_u$  among the  $k$  predicted documents:

$$\text{Recall}(k) = \frac{1}{k_u} \sum_{j=1}^k \text{rlv}(u, d_j) \quad (12)$$

As described before, an evaluation of ML techniques for classification or recommendation requires a dataset which is used for training and testing. Often, a single evaluation is considered not to be representative for an evaluation. Therefore, *10-fold cross validations* are considered for the datasets. In cross validation, several evaluations are performed, using different training and test splits each. A widespread evaluation technique is the 10-fold cross validation. In such an evaluation setup, the dataset is split into ten equal parts. A first evaluation is undergone using one part as the test set and the other nine parts as the training set. This is iterated ten times until each of the ten parts has been considered as test set once. The results of the different evaluations are aggregated and presented with the corresponding variance.

When it comes to evaluating a recommendation based on the history of a user, the time dependencies in the dataset have to be considered as well. The knowledge of the system at the time of recommendation has to be considered to evaluate the system consistently. This is especially true, as a worker may develop further skills and preferences when interacting with tasks in task markets. Splitting the dataset in the fashion of a 10-fold cross validation may train the model on the basis of tasks that were performed some time after the tasks that the evaluation expects to be recommended in the test set. Therefore, recommenders are often evaluated considering timely data. Here, several iterations may be performed considering only parts of the whole dataset, first training on, e.g., the first week and evaluating on the second week, then growing the coverage of the dataset by training on the first month and evaluating on the second month, and so on. This allows to evaluate the recommender against timely patterns and events in the data and may provide insights on how much training data is required to perform a good recommendation.

CROWDSOURCING methodologies allow companies to outsource simple tasks via PCPs, as discussed in Chapter 2. It has to be considered, that not every task is suited to be externalized and for some tasks, it is not advantageous to outsource them using crowdsourcing. There are tasks for which specific knowledge is necessary, which is often derived from confidential information. Outsourcing such tasks to an anonymous crowd may produce copyright and patent concerns [166, 180]. An essential asset of enterprises is the knowledge and skills of their employees. Building knowledge and skills within the enterprise and not letting them drain to the external crowd can be an argument against the external assignment of tasks [64]. Therefore, tasks crowdsourced in micro-task markets often provide only small contributions, such as annotating images. Tasks that require very specific knowledge or sound skills, which provide a profound contribution, are rather difficult to outsource [180].

ECPs provide a solution to these problems by leveraging the workforce of the people already employed by the enterprise. Enterprise crowdsourcing enables similar methodologies as public crowdsourcing while restricting the crowd to the employees of the enterprise. This allows crowdsourcing more complex, confidential, and knowledge dependent tasks. In order to leverage the advantages of crowdsourcing methodologies, the specific requirements of enterprise crowdsourcing have to be considered (RG1). For the distribution of tasks within the enterprise, appropriate task assignment strategies have to be designed (RG2).

Therefore, this chapter provides a conceptual design for ECPs, focusing on the practical management of crowdsourcing processes in general and task assignment strategies in particular [22, 142]. Specific application scenarios of ECPs are considered, to gather the requirements for the practical management of crowdsourcing processes in the enterprise. Therefore, a categorization of scenarios considered in related work is presented, and a qualitative study with experts (n=11) from the industry is provided. Based on two representative scenarios, derived from the study results, a conceptual design of crowdsourcing processes for ECPs is designed and discussed in a user evaluation. It considers the aspects of task assignment strategies in detail.

This chapter is structured as follows. A brief overview of related work about modeling ECPs is provided in Section 3.1. A detailed analysis of the scenarios, elements, and instruments in ECPs is presented in a structured literature review in Section 3.2. Section 3.3 provides the results from the qualitative study. The individual workflows and assignment strategies of the conceptual design are presented in Section 3.4 and critically discussed in Section 3.5. The results are summarized, and an outlook on future research is given in Section 3.6.

### 3.1 RELATED WORK FOR MODELING ENTERPRISE CROWDSOURCING PLATFORMS

Modeling crowdsourcing platforms, in general, is a task that has been studied for more than a decade now. With the start of *MTurk* in November 2015 and such business models as of *iStockPhoto*<sup>1</sup> and *Threadless*<sup>2</sup> described as “crowdsourcing” by Howe [74], many researchers are interested in the dynamics and potentials of such platforms. Studying the dynamics of crowdsourcing platforms in order to establish behavioral models of workers and tasks helps to understand how crowdsourcing platforms can be optimized for different aspects. Hoßfeld et al. [72] provide a statistical analysis of a crowdsourcing platform and develop a model to estimate the activity of the users as well as other dynamics of the platform. Hirth [68] analyze crowdsourcing platforms in order to understand their processes from the given data on different platforms. Much effort was put to describe the different models of crowdsourcing in a more theoretical manner [52, 134, 148].

Brabham [25] discusses many different aspects of crowdsourcing. Besides insights into how crowdsourcing can be organized, he also describes general concepts and theories of crowdsourcing. Providing example cases of how crowdsourcing is applied, he also discusses ethical issues and challenges of the crowdsourcing approach. In his analysis on the future of crowdsourcing, Brabham predicts that “Crowdsourcing will move from a technological approach to a business service.” and emphasizes, that “As the field of crowdsourcing grows, so too does the need for an understanding of the practical management of such projects and communities.” [25]. Looking into the possibilities of crowdsourcing as an internal business service, the approaches for modeling Enterprise Crowdsourcing Platforms (ECPs) presented in this thesis gather requirements on the practical management of crowdsourcing processes and communities with a strong focus on task assignment strategies in project scenarios.

Besides the work on modeling crowdsourcing platforms presented above, there is a great corpus of literature for crowdsourcing platforms in general. While this chapter focuses on modeling ECPs, the work on PCPs provides contributions that are of high relevance. The works of Geiger et al. [52], Doan et al. [37] and Durward et al. [39] define general processes involved in crowdsourcing and required for crowdsourcing platforms in general.

Modeling crowdsourcing platforms usually relies on existing platforms as described above. In the case of ECPs, there is no publicly available data as they are restricted for internal enterprise use only. Also, before such a platform can be applied within an enterprise, it is crucial to know how such a platform can be used and what the expected benefits are for the enterprise. Therefore, elements and scenarios for ECPs have been studied as well as discussed in the following.

Zuchowski et al. [191] provide a conceptual framework for enterprise crowdsourcing which they call: “internal crowdsourcing”. They analyze and compare problems, governance, people, IT, process and outcomes of “internal crowdsourcing” against the same issues in “external crowdsourcing” (public crowdsourcing) and a tradi-

---

<sup>1</sup> <https://www.istockphoto.com>

<sup>2</sup> <https://www.threadless.com>

tional hierarchy based work. Within their ECP framework, they introduce essential elements of ECPs as can be seen in their visualization in Figure 6. They provide a general understanding of the potentials of ECPs but do not provide specific requirements about the practical management of processes or task assignment strategies.

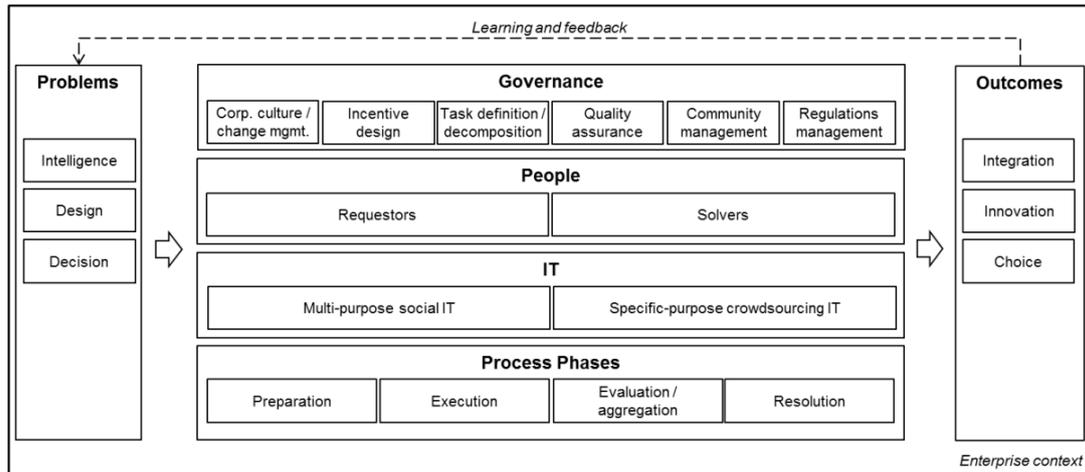


Figure 6: “Conceptual framework for internal crowdsourcing” as in [191].

Vukovic [165] discusses practical applications of enterprise crowdsourcing and provides specific examples of how ECPs can be beneficially deployed in the industry. In their works, they provide solutions for various challenges for the design of ECPs [165, 169, 173]. These results serve as the foundation for the solutions provided in this chapter. They served as a starting point and enabled the author to analyze and design aspects of ECPs in detail.

Hetmank [63] provides different approaches to describe elements, functions, and components crucial for ECPs. He provides a literature review that elaborates on the different interpretations of enterprise crowdsourcing and describes possible application domains [65]. In a scenario-based fashion, he discusses and describes the main elements and functions of enterprise crowdsourcing [64]. A focus on the practical management on crowdsourcing processes and communities is set by introducing the main elements of “user management,” “task management,” “contribution management,” and “workflow management” for his enterprise crowdsourcing framework shown in Figure 7 on Page 36. The scenario-based approach is well constructed, and he shows that his prototype covers many aspects of ECPs. However, details of processes of the different management elements are not discussed and also not directly derived from the provided scenarios. Therefore, this thesis provides an in-depth analysis of these management elements and provides scenario-based solutions for crowdsourcing processes, with a focus on task assignment strategies.

Some authors of related work focus on a particular domain to describe a framework and to provide specific solutions for task assignment as an optimization problem [71, 109]. For example, they assume a given skill ontology and provide algorithms to optimize the average skill coverage in a simulated environment. In contrast to this, the approaches presented in this thesis focus on the design aspects of

task assignment strategies in enterprise crowdsourcing to understand and provide solutions for the practical management of ECPs, their processes, and communities. Therefore, the following paragraphs discuss the impact of the previously discussed work of Hetmank and Vukovic on task assignment strategies.

Hetmank [63] describes a general model for ECPs, shown in Figure 7, where the task management component is a crucial part of the model. For this component, he recognizes, that “allocating the right task to the right person” is a critical issue within this area. He highlights the possibilities of choosing target groups and considers the skills and the availability of workers. Besides that, he does not discuss task assignment strategies in detail. Vukovic et al. [173] provide a system for a technical writing service which includes an expert discovery system as a main component. This expert discovery system is able to rank experts depending on their expertise for a task. This system provides a very specific solution for finding experts but does not discuss further task assignment strategies.

Zogaj et al. [189, 190] analyze different aspects of the application scenario of *software testing*. They provide governance processes for crowdsourcing in general but also focus on aspects such as task creation, assignment, and management processes. La Vecchia and Cisternino [92] describe three methods for the assignment of tasks. They define the methods of a “marketplace,” a “campaign,” and an “auction,” discussed in detail in the following structured literature review.

Related work that targets assignment processes is provided by Geiger and Schader [51]. They specifically target the area of task recommendation, where different approaches allow to recommend tasks based on preferences of individual workers.

To identify the requirements for ECPs in a structured manner, the approaches are based on the categorization within the literature. Task assignment strategies often depend on the type of task, i.e., the application scenario of the task. Categorizations of tasks have been provided from different viewpoints by Hetmank [65], Vukovic [165], Erickson et al. [44], Geiger and Schader [51], and Tranquillini et al. [161]. Within the literature review in Section 3.2, an additional categorization is derived, which takes the aforementioned literature into account, but enables a categorization based on scenarios of enterprise crowdsourcing found in the literature and the qualitative study.

### 3.2 STRUCTURED LITERATURE REVIEW

This literature review has been compiled using methods for searching, selecting and analyzing sources to provide the most complete and structured source identification and evaluation as possible [164, 175]. It follows the methodology of Vom Brocke et al. [164], who describe five phases for a well-structured literature review: “Definition of view scope” (1), “Conceptualization of topic” (2), “Literature search” (3), “Literature analysis and synthesis” (4), and “Research Agenda” (5).

The definition of the view scope (1) follows the taxonomy of Cooper [32]. This literature review *focuses* on applications, best practices and research results related to enterprise crowdsourcing. The *goal* is on the one hand, to identify and formulate

Table 1: Search queries used on different scientific databases.

TERM 1	TERM 2	SEARCH QUERY
E	C	(Author Keywords: "enterprise" AND "crowdsourcing") AND (Publication Title: "enterprise" AND "crowdsourcing") AND (Abstract: "enterprise" AND "crowdsourcing")
I	C	(Author Keywords: "internal" AND "crowdsourcing") AND (Publication Title: "internal" AND "crowdsourcing") AND (Abstract: "internal" AND "crowdsourcing")

a categorization for enterprise crowdsourcing applications and scenarios. On the other hand, this literature review strives to identify elements and instruments of enterprise crowdsourcing, that have not been considered for the general enterprise crowdsourcing model of Hetmank [63] and to extend this model accordingly. The literature review is *organized* according to the elements and instruments of ECPs identified in the literature, while the *perspective* is of a neutral nature. It is presented for a target *audience* of practitioners and decision-makers who need to understand the requirements of enterprise crowdsourcing before implementing such a system in their respective companies. The review *covers* eight scientific databases, chosen due to their close relation to digital information systems.

The conceptualization of the topic (2) is presented along the literature search (3) in Section 3.2.1. The main part of this literature review provides the results of the literature analysis and synthesis (4). The categorization of the literature into applications and scenarios is provided in Section 3.2.2. The discussion of elements such as actors and tasks (Section 3.2.3) as well as instruments for the management of elements (Section 3.2.4) leads to the extended enterprise crowdsourcing model provided in Section 3.2.5.

### 3.2.1 Conceptualization and Literature Search

The conceptualization (2) was focused on the topic of enterprise crowdsourcing. To specify the search terms, several sources were analyzed to understand concepts of internal crowdsourcing and enterprise crowdsourcing further. Accordingly, the following search terms have been identified: (E) enterprise, (I) internal, (C) crowdsourcing. The search was restricted to abstracts, titles, and keywords, as well as filtering for literature that is peer-reviewed. The search terms have been combined into the search queries (EC) enterprise crowdsourcing and (IC) internal crowdsourcing shown in Table 1. These queries were used for the database of the *IEEE Xplore Digital Library*. The search queries for the other databases were adapted to their corresponding search masks.

For the literature search (3), scientific databases related to the defined topic were selected. Table 2 lists the used databases with their URLs. The results covered a wide

range of publications from the most important journals and conferences on enterprise crowdsourcing topics. The search resulted in 424 sources. Removing duplicates led to a result of 315 articles as shown in Table 2.

The literature search also included further filtering steps. Articles outside of the view scope were removed when title and abstract revealed, that they are not relevant to the topic (coarse filter). This included articles about marketing, medicine, politics, and other areas without any connection to ECPs, leaving 279 references. Reviewing the remaining articles by their title and abstract towards specific inclusion and exclusion criteria left 37 relevant references (fine filter). Inclusion criteria were, e.g., 'covers chances, problems, or risks of ECPs' as well as 'focus on design, implementation or elements of ECPs.' Exclusion criteria were, e.g., 'article not peer-reviewed' and 'mentions search terms without further elaboration.' Additionally, the 37 articles were analyzed for referenced and referencing articles (forward and backward search), using the same exclusion and inclusion criteria. This resulted in a final set of 41 articles included in the following literature review.

Table 2: Queried databases and search results depending on the queries (EC) and (IC).

NAME	URL	RESULTS PER QUERY		
		EC	IC	SUM
Ebscohost	search.ebscohost.com	27	44	71
Science Direct	www.sciencedirect.com	43	26	68
Google Scholar	scholar.google.de	5	5	10
Web of Science	apps.webofknowledge.com	5	4	9
AIS Electronic	aisel.aisnet.org	39	45	84
Wiley Online	onlinelibrary.wiley.com	56	10	66
ACM Digital	dl.acm.org	28	27	55
IEEE Xplore Digital	ieeexplore.ieee.org	50	10	60
Total sum		253	149	424
Sum without duplicates from queries				315
Sum filtered by coarse filter				279
Sum filtered by fine filter				37
Sum after forward and backward search				41

### 3.2.2 Categorization of enterprise crowdsourcing Scenarios and In-Depth Analysis

Within the literature analysis and synthesis (4), this section provides a categorization of scenarios for ECPs and discusses the categories and the corresponding literature in the subsections. Different categorizations from the literature are analyzed to identify a relevant categorization. The following categorizations from related work are considered:

- Hetmank [65] divides enterprise crowdsourcing into seven areas: business process outsourcing, collective intelligence, commons-based, peer production, human-based computation, open innovation, and *open source*.
- Erickson et al. [44] divides enterprise crowdsourcing into five areas: productivity, product/service innovation, marketing/branding, knowledge capture.
- Tranquillini et al. [161] divides enterprise crowdsourcing into five areas: product design, social marketing, idea management, e-democracy, human computation.
- Vukovic [165] divides enterprise crowdsourcing into four areas: design and innovation, development and testing, marketing and sales, support competition.

The presented categorizations mainly refer to enterprise crowdsourcing concepts in general. Only some of the categories reflect specific scenarios within ECPs, e.g., *product design* and *idea management* of Erickson et al. [44] as well as the *design and innovation*, *development and testing* and *marketing and sales* categories of Vukovic [165]. Analyzing and categorizing the identified literature showed that some of the work could not be categorized into a given category or provided very imbalanced categories. Therefore, an adapted categorization into specific scenarios leads to the provided categories given below.

- *Productivity* scenarios describe the transferring of tasks from daily work activities of employees to an ECP. This range includes 12 of the 41 sources and is divided into further sub-categories:
  - *Software development* scenarios describe tasks directly related to the development of software in enterprises.
  - *Other* productivity scenarios describe tasks of employees in enterprises, which may be outsourced but do not fall into the other two categories. This includes activities related to the creation and editing of texts, presentations, etc., in an enterprise; for example, translating a text.
- *Knowledge management* scenarios describe tasks that deal with finding, creating, storing and making knowledge available in companies.
- *Idea management* scenarios include both the collection of ideas and suggestions for improvement and their specification. Ideas and suggestions can refer both to products and processes.
- *General* articles do not describe scenarios.

The distribution of the references to their corresponding category is given in Table 3, where only Hetmank [64] discusses scenarios from more than one category: *productivity* and *knowledge management*. The different categories and the scenarios of their according literature are discussed in detail in the following sections.

### 3.2.2.1 Scenarios of Productivity

*Software development* scenarios represent two thirds of the identified productivity scenarios in the literature. Therefore, they are categorized into their own sub-category, besides *other* productivity scenarios. When it comes to *software development*, especially when crowdsourcing programming tasks, there are peculiarities that must be consid-

Table 3: Categorization of the reviewed literature.

CATEGORY		SUM	REFERENCES
Productivity	Software Development	8	[40], [78], [79], [100], [138], [139], [152], [157]
	Other	4	[36], [59], [64], [156]
Knowledge Management		11	[64], [102], [165], [167], [168], [169], [170], [171], [172], [173], [178]
Idea Management		4	[84], [150], [151], [158]
General		15	[6], [9], [10], [43], [44], [65], [66], [67], [91], [92], [123], [159], [160], [166], [191]

ered when designing ECPs. Software is usually divided into different modules and these into smaller units, such as classes and methods. Therefore, it is essential that the tasks can be split into smaller subtasks, although the integration of the individual parts must also be taken into account [139].

A second essential point in software development are the skills and characteristics that are necessary to solve the tasks. This includes, for example, the mastery of a specific programming language, as well as the level of experience [79]. For a better selection of possible crowd workers, the properties and skills required for the task should be recorded as attributes and then compared with the profiles of the crowd workers. This requires well-structured crowd management.

Crowd workers also need access to the necessary tools and systems required for the completion of software development tasks. Access can either be provided with the task as a prerequisite, or is granted to the employer after the assignment decision. After the task has been carried out by a crowd worker, the required quality must be ensured, and integration into the overall project must be carried out. Quality assurance is an essential component of the system and can also be carried out automatically through tests. In addition to pure development, software testing is also classified under this category of crowdsourcing, as it is a part of the software development process.

The second sub-category of productivity scenarios consists of *other* productivity related tasks can be outsourced to ECPs. One example scenario is the translation of texts that is described by Stewart et al. [156]. Some scenarios also describe where enterprise crowdsourcing was used to determine the location of objects, to convert speech to text and to cluster and label data.

### 3.2.2.2 Scenarios of Knowledge Management

For scenarios of knowledge management, it is essentially a matter of collecting, maintaining and making available the existing knowledge within a company. In this area, Vukovic and Bartolini [166] and Vukovic and Das [167] describe the possibility of capturing this knowledge with a campaign via an ECP. To this end, the initial definition

is made of what the campaign is to cover, and individual questions are then formulated, which query the information to be collected. These individual questions are then compiled by the client into a questionnaire. Before the questions are assigned to the crowd, the experts who have the required information must be identified, and the questionnaire is then assigned to them. If the questions cannot be answered completely, the task can be delegated. After the questionnaires have been completed by the experts in the crowd, the information collected is combined and evaluated. In order to guarantee a sufficient quality of the contributions, it is necessary to establish a quality assurance process. For this task Vukovic and Das [167] describe a specific workflow, which allows the integration of quality assurance into the platform.

### 3.2.2.3 *Scenarios of Idea Management*

Scenarios of idea management involve finding and substantiating ideas for new products, improving existing processes or optimizing them. This kind of enterprise crowdsourcing usually follows the assignment strategy *campaign*, where everyone in the crowd can submit his or her ideas. For such tasks, it is essential to address the right crowd. Therefore, there exist different possibilities of opening the internal crowd to external contributors. In general, a larger crowd is considered beneficial to generate more ideas. However, there may also be reasons to restrict and limit the crowd to internal employees if necessary. This is particularly the case if trust is a necessary prerequisite for finding an idea, for example, if the information is confidential or the campaign is not to be communicated externally. However, the number of ideas submitted is severely limited due to the limited number of employees. Therefore, the definition of the crowd is the core element in idea management scenarios and must be adapted to the respective situation so that the results are satisfactory [151, 158].

### 3.2.2.4 *General Sources in Enterprise Crowdsourcing Literature*

Besides the articles that discuss scenarios of productivity, knowledge management, and idea management, the categorization in Table 3 also includes articles that do not discuss specific scenarios. These articles provide a meaningful contribution to the research area, by discussing the design of ECPs and the management of crowdsourcing projects, processes, and communities, without discussing or deriving their findings directly from application scenarios, but from rather general assumptions. These articles contribute to the following discussion about the elements of enterprise crowdsourcing.

### 3.2.3 *Elements of Enterprise Crowdsourcing*

Besides the categorization of scenarios given above, the literature analysis and synthesis (4) focuses on elements and instruments of ECPs. The literature usually describes four core elements of ECPs, which are discussed in Chapter 1 and Chapter 2. These four elements include the two types of actors (*worker* and *requester*), the *tasks*

that the actors request or work on, and the *contribution* (results) of solved tasks. These elements are described in detail in the following sections.

### 3.2.3.1 Actors on ECPs

Among the actors, it is distinguished between those who can provide tasks (*requester*) and those who perform them (*worker*) [66, 67, 165]. In some constellations, both characteristics are assigned to all actors, which means that there are no differences between the actors. Besides these main actors, there can be others, such as a platform administrator or actors who can make evaluations [166, 167, 171].

The characteristics of the actors can be assigned to individual users via roles, which leads to the fact that a user can also have several roles. In the literature [10, 36, 65, 78, 139, 161] the workers are advised to maintain a detailed profile of their skills and knowledge in order to simplify the assignment of tasks. The profile of a worker should be used to record not only their skills but also their experience level and, if necessary, their validation by the already performed tasks. At the same time, attention should also be paid to the availability of workers, as it is pointless to propose a task to a worker who is currently unavailable [10, 159]. For the requester, no detailed profile is required. However, it can be interesting for a worker to retrieve information about the requester before accepting a task. This could include the rejection frequency of tasks in the past by a requester.

### 3.2.3.2 Elements of Tasks on ECPs

A task should contain as much information as possible about the activities to be performed, and its design can depend on the scenario that is to be outsourced. In idea management, it is less necessary to have particular skills to perform a task, as in software development. All tasks have in common: a name, a brief description of the task, the client, a start and an end date [9, 10, 36]. All other attributes are optional. A selection of the identified attributes and a short description are given below.

- *Incentive type and amount*: It is necessary to provide remunerations depending on the effort and depending on the kind of incentive that motivates the workers, to encourage them for engagement on the platform. [36, 67, 161].
- *Task type*: Many workers but also recommender systems rely on given categorizations to support task assignment. Therefore, a task should be related to at least one task type. [10, 65, 78, 139, 159].
- *Duration of execution*: An estimated execution duration should be specified for the task, to represent the scope of the task for the worker. There are tasks for which the duration of execution can be precisely defined (e.g., viewing a video), whereas for most tasks the duration of execution is only an estimated value of the requester. The duration can also be to check for the availability of a worker for a task [10, 65].
- *Complexity*: The complexity of the task can be estimated in order to prevent an inexperienced worker from selecting a task too complex [9, 65, 139, 159, 161].

- *Information on subtasks*: If a task consists of smaller subtasks or is itself a subtask, this information should be provided to everyone involved [10, 65].
- *Task assignment strategy*: Different task assignment strategies may be appropriate, depending on the given task scenario [156, 161]. Task assignment strategies will be subject of further elaboration in later sections.
- *Confidentiality and visibility*: enterprise crowdsourcing is particularly suitable for tasks that cannot be published due to their confidentiality requirements. It has to be ensured that only workers with an adequate confidentiality clearing can view corresponding information [36, 65, 159].
- *Restrictions*: There can exist further restrictions that exclude specific workers from certain tasks [139, 152].
- *Necessary skills and knowledge*: These are probably one of the most important attributes of a task. Especially with increasing complexity of the tasks, these attributes become relevant to assign tasks properly. Knowledge and skills enable the worker to perform a task but can also be acquired and expanded by performing tasks [10, 36, 65, 78, 139, 161].
- *Resources*: The resources made available to the worker play a role in the completion of a task. These may include for example technical devices or software licenses [63, 152].

### 3.2.3.3 Contribution of Tasks on ECPs

The last core element is the contribution of the crowdsourcing process, which is handed back to the requester after the execution of a task. Such results can be of very different nature and strongly depend on the given scenario. In software development, it could be a part of code, whereas in idea management it can be a prototype or a description of a new product [40, 158]. The expected format of the results should be defined in the task description, i.e., in the acceptance criteria.

It is argued within the literature whether the provided incentive for a task is part of the task results. Usually, the argumentation is, that the incentive for a task plays its most important role as motivator before a worker accepts the task. Therefore, the resulting incentive is mostly defined before execution and is not regarded as a result of the crowdsourcing process.

### 3.2.4 Instruments of Enterprise Crowdsourcing

The described elements interact with each other. Various instruments of ECPs implement this interaction. Instruments for user, task and contribution management are described by Hetmank [63], who also defines a workflow management system (see Figure 7). The instruments are presented below, following the model of Hetmank [63], but refer to more detailed related work, to identify further important instruments.

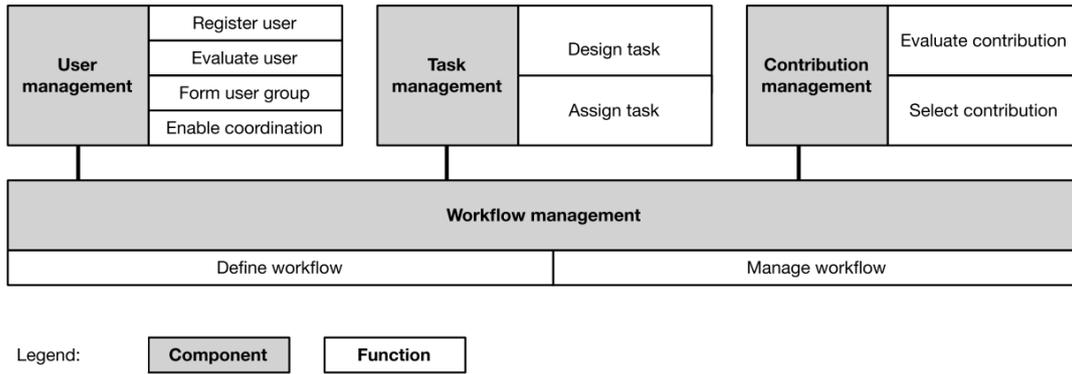


Figure 7: Core instruments of ECPs according to Hetmank [63].

### 3.2.4.1 User Management on ECPs

According to Hetmank [63], the user management must offer the following four functions: The registration and evaluation of users, as well as the formation of user groups and their coordination. During registration, users enter their data relevant to the platform, from which user profiles are generated [165]. The user is evaluated during the user evaluation and the profile adjusted if necessary. The skills and the respective level in the user profile are adapted if the evaluation of a task completed by the user requires such action. The evaluation takes place on the basis of the data from the profile and through an analysis of the work history and the currently selected tasks of the platform. In addition, a user's response speed to questions can be included in the user's evaluation [152].

The third function is the formation of user groups. This plays an important role especially for more complex tasks that cannot be solved alone or for which groups have an advantage regarding time or quality constraints. The platform must enable and promote the formation of groups. For this purpose, the literature describes systems which, from the past cooperation on the platform and data from other company-internal tools, suggest possible well interacting teams [152, 167]. Especially for tasks that are highly dependent on other tasks or require a high degree of skill diversification, the composition of the team should be taken seriously.

In addition to the composition of the team, the coordination and collaboration between the members, but also the entire community should be made possible by the platform. This can be done, for example, by integration of existing collaboration systems and by the possibility to disassemble and delegate tasks [139, 167]. Also, communication between crowd workers and clients should be facilitated in order to eliminate, for example, ambiguities regarding tasks. In addition, this part of the ECP contains the feedback function with which requesters can assess the processor of a task.

### 3.2.4.2 Task Management on ECPs

Hetmank [63] divides the task management into two sub-functions: task creation and task assignment. When the task is created, all decisions about the design of the task are made, and the task is described as precisely as possible with the help of the above attributes.

The second function is the task assignment process. The task assignment process is either centrally defined for the entire platform, or defined differently for particular tasks, depending on their attributes and requirements. The task assignment process can be defined using various task assignment strategies. The relevance of the correct choice of a task assignment strategy is justified by the strong influence on the resource expenditure and thus the efficiency of the task solution [92, 161]. Three task assignment strategies are described by Vecchia [92], with which also combinations or modifications are possible. These strategies can be seen as basic strategies, which can be adapted according to the requirements of the task.

In a *marketplace*, different tasks are openly accessible, and workers assign themselves to tasks, which supports mainly productivity scenarios. This type of assignment strategy is especially suitable for simple, small and cohesive tasks, since the effort, the result, the compensation, and the required skills must be known [92, 161]. In the literature, this assignment strategy is often used in connection with crowdsourcing in the area of productivity and knowledge management, if the scope of a task can be clearly defined [36, 78, 156, 166].

A *campaign* can be used to gather solutions for the same tasks from heterogeneous workers. This assignment strategy is particularly suitable for tasks in the creative field, for example from idea management, where diversification is necessary [36, 92]. A significant disadvantage of this assignment strategy is that it is very resource-intensive, since all not selected solutions have also tied up resources, but are ultimately worthless [161].

The strategy of an *auction* supports tasks where the requester is not able to judge the effort or the complexity. In an auction, the individual workers bid on the tasks, and the requester selects the one, who from his point of view has made the best bid [161]. The bids of the crowd workers consist not only of the profile characteristics but also of the expected compensation and the required solution time.

In addition to these three strategies, further strategies can be defined, which provide variations of the basic strategies to adapt to specific task requirements or goals of platform provider and task requester. One of these modifications is the possibility of delegating tasks to other crowd workers. This is necessary so that a crowd worker who was assigned to a task, but is not available or cannot provide a proper solution, can delegate the task to a worker who has the knowledge, skills, and resources to accomplish the task [167]. Appropriate task delegation can also be seen as a valuable contribution by a worker for the respective crowdsourcing platform [172]. Such cases occur in particular when tasks are directly assigned to workers.

Direct assignments assign tasks to specific groups of crowd workers without them having any influence on it. As a basis for this assignment, the history of the workers can be analyzed and compared to the task at hand. Also, explicitly managed skills

of the workers can be matched against explicitly stated task requirements. Balamurugan et al. [10] refer to a corresponding value as the *human performance index* of a crowd worker. However, the risk is taken that not all experts are activated on a topic since the selection system works only with incomplete information [178]. The choice of strategy for the assignment process has a considerable influence on the speed and efficiency of task execution and should be carefully chosen and adapted to the scenario at hand [92, 161].

#### 3.2.4.3 *Contribution Management on ECPs*

The core of the contribution management as defined by Hetmank [63] is to assess and communicate the quality of the results produced by the workers. To this end, functions of evaluation, integration, and selection of the results must be carried out.

Evaluation is about assessing the results submitted by the workers and giving them feedback on this basis. Also, the skill profile of workers can be adjusted on the basis of evaluation [78, 92]. For an effective evaluation and feedback mechanism, the following procedure is recommended. First, it must be specified who carries out the evaluation, the worker himself (self-assessment) or the creator of the task (external assessment). Secondly, the specificity of the evaluation must be defined. So whether it is only a simple evaluation – results are accepted or not sufficient – or whether there is a defined questionnaire for evaluation or even completely open feedback to be formulated. The third element is the time frame in which the feedback is given to the worker, i.e., immediately or delayed.

The second function is the integration. This is relevant if the value of a contribution depends on other results or the embedding in an overall system. This can either be done by the requester or assigned as a task via the platform.

The third important function of results management is the selection of the final contribution. For workflows in which more than one result is submitted, a selection process must be defined. For example, it can be done by a majority decision in which the result that was submitted most often [70] is selected. Another variation would be to evaluate through a control group. If only one result is submitted, since a task has only been processed by one person, there is the possibility that the evaluation is carried out by a group of persons. The platform can also be used for this by creating evaluation tasks that are executed by the crowd. On the basis of the group evaluation, a quality statement can be made about the submitted result. Another possibility, described by Vukovic and Das [167], has a group of experts evaluate the quality in a workshop. In addition, there are also automatic procedures for quality assessment, whereby the suitability of this depends very much on the scenario. For creative works, machine evaluation is less suitable, whereas in software development machine methods are particularly suitable [100, 138].

#### 3.2.4.4 *Workflow Management on ECPs*

The interaction flow between the elements on an ECP is implemented by Hetmank [63] using workflows that form the interface between the individual elements. There-

fore, Hetmank [63, 66] considers the workflow management system as the central tool of an ECP, as it provides the two primary mechanisms necessary for the use of the workflows. On the one hand, it enables the creation of new workflows, i.e., the definition of the concrete process of creating and processing tasks on the platform. On the other hand, it enables management functions for existing tasks, which include, for example, the optimization and adjustment of workflows. Since the elements described above, the actors, the tasks and the results, must also be managed on their own, a separate management system is given for each element (see Figure 7). The workflows determine how tasks are processed on the platform.

### 3.2.5 *Extended Enterprise Crowdsourcing Model*

The examined literature describes additional elements, instruments, and management systems, not considered in the model of Hetmank [63] (Figure 7 on page 36) since they depend very strongly on the application scenarios in their analysis. Therefore, an extended model is provided in Figure 8 and shortly described below, while a detailed discussion of the extensions is given in the following sections. This extended model provides a contribution to the definition of requirements (RG<sub>1</sub>) and provides the foundation for the later specification of task assignment strategies (RG<sub>2</sub>) in Section 3.4.

Depending on the previous discussions, the following components are added in the extended enterprise crowdsourcing model. The *community management* function is considered a significant extension to the user management. The *decomposition and composition* functions are essential for active management of tasks and is split between the task management and contribution management components. The *incentive/compensation management* component plays a critical role in enterprise crowdsourcing and provides interfaces towards all other major components.

#### 3.2.5.1 *Community Management on ECPs*

The literature describes the community management as a general requirement for the success of ECPs. This instrument enables users to exchange valuable information and support each other when working on tasks [156]. In community management, the needs of the community, in particular, are addressed and analyzed. The goal is to increase the number of participants and keep them on the platform. The feedback from a lively community also serves as a great motivational factor for many workers [167]. The community management is considered as a part of the user management in the presented extended model Figure 8.

#### 3.2.5.2 *Task Decomposition and Composition on ECPs*

The decomposition and composition of tasks provide necessary functions, in order to divide complex and extensive tasks into subtasks and to integrate their results into a combined contribution. It enables the crowd to find the decision of how to decompose and newly distribute tasks on its own. In their crowdsourcing governance

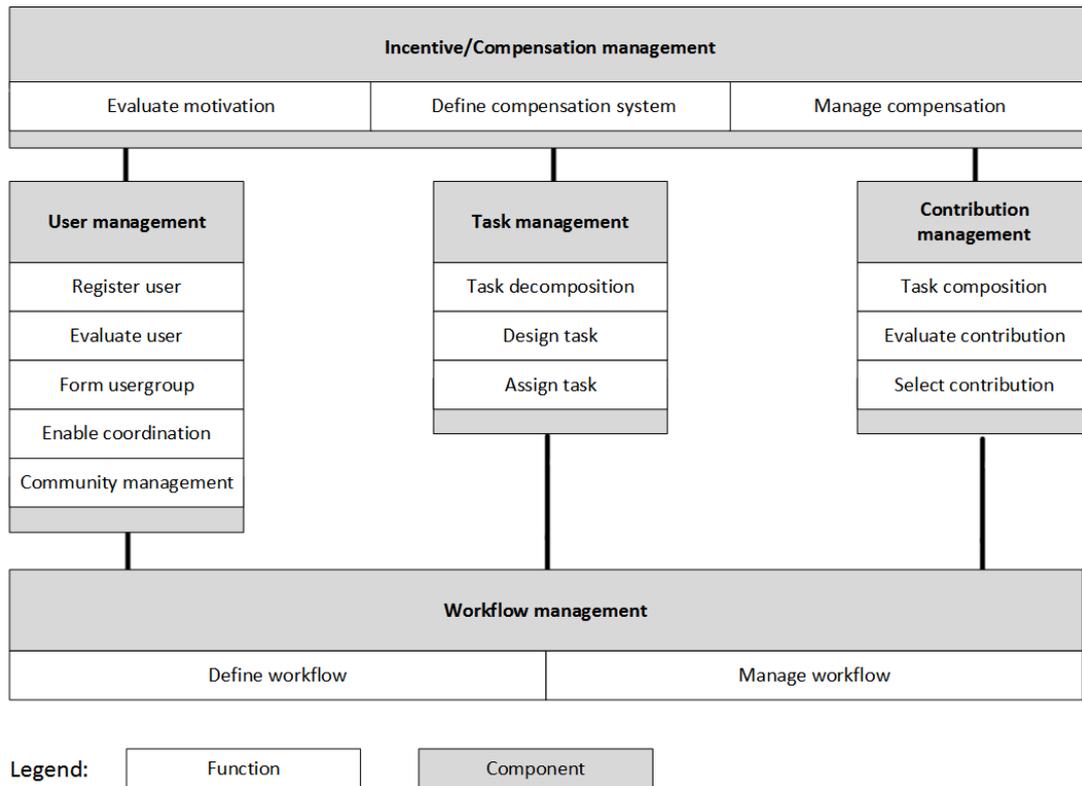


Figure 8: Extended Model of an ECP Based on the Literature Review [142].

model, Zogaj et al. [190] include this as “task modularization.” Decomposition can be done manually or by an algorithm. Their software offers the structure to perform a mechanical decomposition and composition in order to allocate the development into smaller work packages, distribute them for processing and then reassemble them. However, it is necessary to prevent that individual workers only decompose tasks (*delegation factories*), while others only solve such tasks (*delegation sinks*). This leads to unbalanced resource utilization [139].

This instrument is mainly related to the task management system, as it profoundly affects the creation and distribution of tasks when decomposing them. On the other side, the composition of tasks is mainly related to the contribution management. For a decomposed task to provide a complete and meaningful solution, the distributed parts have to be composed back together into a single contribution. Therefore, the extended model as presented in Figure 8 considers the task decomposition as a function of the task management component, while the task composition is considered as a function of the contribution management.

### 3.2.5.3 Incentive/Compensation Management on ECPs

Most of the literature argues, that the motivation of workers is a driving factor for the success of crowdsourcing platforms [51, 83]. The design of the *compensation and*

*incentive system* is a vital matter in ECP since the workers are already compensated for their working time.

In addition to compensation, the chances for personal development of a worker can be motivating for the execution of the tasks. By performing tasks, new skills can be acquired, or existing ones can be deepened that enhance the employee's profile [10, 23, 92]. Especially in task management, the motivation of the employees should be considered to determine the choice of the incentive mechanism and the amount of compensation.

There can be a wide variety of motivations that must be taken into account by the incentive system. Guy et al. [59] identify three different motivators for a crowd by analyzing a survey with users on an ECP using gamification. First, changing the context of everyday activities; second, expanding knowledge and skills; and third, as an addition to everyday activities.

Besides compensation and personal development, there are many more motivators identified in the literature, which depend on the focused scenario and are subject to individual preference. Kaufmann et al. [83] divides the motivators into intrinsic and extrinsic motivation. Enjoyment-based motivation and community-based motivation are considered intrinsic, while payment, personal development and social motivation are considered extrinsic factors.

The success of every single task, as well as the success of the ECP in general, depends on whether a motivated workforce can be established. The motivation is what drives the user to select a task and provide its contribution. Therefore, the *incentive and compensation management* is considered to be related and of great importance for the modules of user management, task management, and contribution management. The motivation of each worker has to be evaluated by the compensation management component, which therefore includes an interface towards the user management, the *evaluate motivation* function. Depending on many factors, e.g. the task assignment strategy, a suitable compensation system has to be chosen when creating or decomposing tasks. The *define compensation system* function is therefore closely related to the task management. Depending on the provided contribution for a task, the compensation management provides the compensation to the worker. Therefore, the function *manage compensations* provides the interface towards the contribution management component.

The provided extended model concludes the structured literature review. This model is used as a basis of the concept for ECPs presented later. The categorization provided and the scenarios identified serve as a basis for the expert interviews and the respective qualitative study presented in the next section.

### 3.3 QUALITATIVE STUDY

This section describes the methodology, results, and analysis of a qualitative study conducted with experts (n=11) from the industry. The following methodology description explains the selection of enterprises and experts, the creation of an interview guideline as well as the execution of interviews and their analysis. Quantitative

results are presented below in a brief manner, while the subsequent sections discuss the identification of scenarios for ECPs in more detail.

### 3.3.1 Methodology of the Qualitative Study

The methodology follows the best practices provided by Gäser et al. [53]. Before the actual study, a preliminary study was carried out with various participants of an IT-intensive enterprise, to discuss the possibilities of ECPs. On the basis of this preliminary study, it was determined that it is important to distinguish the different stakeholder groups of *decision-makers*, *users* and *developers*. Decision-makers represent individuals who decide whether and how an ECP is used within the enterprise. Developers represent individuals with experience in providing a respective service platform. Users represent individuals who would be using the platform within the enterprise, taking different roles on the platform.

Based on the insights of the preliminary study and the results of the literature review, an interview guideline (see Appendix A.1) was developed for the three different stakeholder groups. The guideline for all stakeholders of the platform first focuses on the discussion of possible application scenarios of ECPs. For each of the identified scenarios, the interview guideline stipulates among others the discussion of aspects of tasks, roles and platform functionality as well as the confidentiality, visibility and assignment strategies of tasks. Afterward, the expected outcomes for the company are discussed from different view scopes. After the discussion of workflows and scenarios, the developers discuss experiences with the implementation of such systems. The decision-makers, on the other hand, discuss the motivation for implementing ECPs in an enterprise as well as metrics for measuring the success in enterprise crowdsourcing. The guide was slightly adapted and optimized after the first interviews with the developers.

The execution of the interviews each took place during a personal meeting at the company's premises. The procedure was the same for all interviews, following the interview guideline by welcoming the experts first and start an informal discussion about enterprise crowdsourcing. Afterward, a short definition of enterprise crowdsourcing was given, the goal of the interview was presented and, an introduction to the field was provided. After obtaining the consent for recording the interview, an audio record was started, and the actual interview was executed. At the end of each interview, thanks were expressed for the cooperation, and feedback was obtained for further improvement of the interview procedure.

After the execution of the interviews, the audio records were used for a transcription of the discussion. The transcribed interviews were then analysed and coded with the support of the coding software tool *MAXQDA*<sup>3</sup>. The quantitative results of the coding are briefly discussed in the following section to identify general requirements as well as scenarios in ECPs. Specific insights from the interviews are discussed in detail in Section 3.4 afterward.

---

<sup>3</sup> <http://www.maxqda.de/>

In the study, eleven experts from five different enterprises discussed aspects of ECPs. All five enterprises (or business units) have a focus on information technology (IT). One of the companies was chosen to focus on specific scenarios. Therefore, seven experts from this company were interviewed. Information about the companies is provided in Table 4, while information about the experts is given in Table 5.

Table 4: The enterprises of the interviewed experts.

	SECTOR (BUSINESS UNIT)	EMPLOYEES	AGE IN YEARS	REVENUE IN EURO	INTERVIEWED EXPERTS
A	Transport / Logistics (BU with IT focus)	>300,000	>80	~40 Billion	1 User
B	Software and Hardware	>300,000	>80	~80 Billion	1 User
C	Publisher (BU for IT Services)	>10,000	>50	~2 Billion	1 Decision-Maker 6 Users
D	IT-Services	11-50	>10	unspecified	1 Developer
E	IT-Services	11-50	>3	unspecified	1 Developer

Table 5: Overview of interviewed experts.

EXPERT	COMPANY	GROUP	POSITION
1	A	User	Product Owner IT
2	B	User	IT Product Manager
3	C	User	UI-Designer
4	C	User	UI-Designer
5	C	User	Social Media Manager
6	C	User	Data Scientist
7	C	User	Head of Product Management IT
8	C	User	Head of Product Management IT
9	C	Decision-Maker	Executive
10	D	Developer	Head of Development
11	E	Developer	Head of Development

### 3.3.2 Quantitative Results of the Study

A total of eleven interviews were recorded, transcribed and coded. The coding resulted in 485 marked relevant passages, which are classified into several coding levels. The general categories are provided in Table 6. On the first coding level, eight main categories are found, while each topic is subdivided into further categories (coding level 2). The most mentions are found within the category *elements*, which

includes previously discussed elements of ECPs like tasks and worker but also components of ECPs like the incentive system and task management (*assignment strategy*). The details behind the statements are discussed in the design concept in Section 3.4. A detailed discussion of *scenarios* is provided in the following.

Table 6: Overview on the interview results with the number of mentions per coding level.

LEVEL 1	MENTIONS	SUM	LEVEL 2	MENTIONS
Elements	5	258	Task	130
			Incentive system	51
			Assignment strategy	38
			Crowd worker	31
			Community	4
			Quality assurance	4
Scenarios	0	51	Productivity	35
			Knowledge Management	14
			Idea Management	2
Collaboration	1	46	Ticketing	17
			Communication	16
			Wiki	3
Handling of data	7	30	Visibility of information	27
			Goals	3
Expectations	25	16	Team work	3
			Bridging departments	13
Roles	0	7	Worker (3)	
			Administrator (3)	
			Manager BI (1)	
Preconditions & Concerns	21	-		
Metrics for evaluation	20	-		

### 3.3.3 Scenario Identification and Description

Within the expert interviews, various scenarios were discussed which are categorized into the coding scheme given in Table 7. *Productivity* scenarios with 35 mentions were discussed the most while *knowledge management* scenarios were discussed 14 times and *idea management* two times.

When it comes to *idea management*, it appears to be less interesting for enterprise crowdsourcing, as the crowd is limited to the employees, which contradicts the prac-

Table 7: The interview results given with the number of mentions per scenario.

LEVEL 2	MENTIONS	SUM	LEVEL 3	MENTIONS
Idea management	2	-		
Knowledge management	2	12	Research	7
			Reporting / Analysis	5
Productivity	0	35	Supporting office tasks	6
			Label and Cluster	5
			Software development	3
			Testing	15
			Other	6

tice of gathering ideas from an open crowd [151]. However, one user discussed a scenario, where proposals for a new logo could be collected via an ECPs. For *knowledge management* scenarios, research tasks, such as collecting information about new technologies and analyzing data, were mostly mentioned. From the analysis of the different statements, these scenarios are subdivided into the categories *research* and *reporting/analysis*. Following the discussions of scenarios in the expert interviews shows, that enterprise crowdsourcing seems to be very suitable for *productivity* scenarios. Therefore, these are analyzed in detail and subdivided into several different kinds of productivity scenarios.

*Supporting office activities* scenarios describe principal or secondary activities of enterprise employees and can be completed using office tools available in the enterprise. One example is the writing of texts for an Internet presence. The experts also mentioned *editing photos*, *creating Excel macros* and *editing presentation slides*. *Label and cluster* scenarios describe tasks concerning the clustering and labeling of data. The results are often used to train Machine Learning algorithms. One user described a project, where this is already in place using a PCP. Scenarios describing *testing* include all tasks are concerned with the testing of program code. Here, the advantages of testing user interfaces with the help of an ECP were discussed in particular. In this context, a user of Company C drew attention to an attempt to perform crowd-testing within the Company introducing contests and prizes to motivate employees. *Software development* scenarios describe tasks that involve the development and creation of program code but excludes testing, which is handled separately. Currently, IT departments suffer from the lack of employees in Germany. The ideas of outsourcing task related to software development or testing within a company might be related to the hopes, that ECPs could relieve the IT department.

Scenarios in the category *other activities* include everything that belongs to the productivity scenarios but does not fall into the categories mentioned above. One example of such an activity is the organization of corporate events, which was mentioned by the decision-maker. Other scenarios also include organizing the activation

of IT tools and taking photographs to verify the current conditions or location of particular objects.

From the qualitative study, a high number of scenarios in the area of productivity are identified. These form the basis for the differentiation of task assignment strategies and corresponding workflows. In the following, two contrary examples are described, which are provided to discuss general concepts for task assignment strategies and workflows in ECPs.

Scenarios of the sub-category *software testing* and *support of office tasks* are mentioned most often throughout the interviews. Therefore, the specific scenarios *testing a graphical user interface* and *create and edit presentation slides* are chosen as examples. In addition, they show contrary requirements regarding the assignment strategy, confidentiality, and incentive mechanisms. These requirements are derived from the detailed transcribed and coded interview results.

The example *testing a graphical user interface* describes a certain task from the *testing* area within the *productivity* category, and serves as a representative task for many similar activities that are important for ECPs in IT companies. Testing a web-based Graphical User Interface (GUI) for example can be easily shared with skilled testers by providing a simple link and collecting written feedback. On the other hand, such a task may require the workers to come to a special laboratory for testing purpose, depending on the data that is supposed to be gathered. *Testing a graphical user interface* is a task that can be directly assigned to every worker of the enterprise, without further constraints in regards of *qualifications* or *confidentiality* level and may even be shared with partners or the public crowd. In some specific cases, GUI testing can be restricted by confidentiality issues, which requires an appropriate community management. Therefore, this example scenario provides motivation and requirements for the task assignment strategy of *direct assignment* presented in the next section.

In some cases, it can be advantageous to require specific attributes or preconditions of the worker, such as the experience in using IT systems or even physical restraints (e.g., color blindness). For the motivation of workers, it was found, that testing a graphical user interface can be perceived as a positive diversion from the day to day office work. As testing is part of the software development process, which is usually undergone iteratively, tasks like this come up in regular intervals. It might be of value, to build a community of testers throughout the enterprise, who are queried regularly for specific testing tasks. As such a task does not support any learning or advancement of the worker, other ways of *motivation* have to be found. Incentivizing the worker with a lottery or recognition in the community can be options. This allows accounting for the necessary financial support without limiting the number of contributions when a direct reward would be promised.

The example *creating and editing presentation slides* describes a task from the *supporting office tasks* area within the *productivity* category. In this example, an employee of the company shares information, e.g., incomplete or outdated presentation slides, to be updated for an upcoming presentation. In this scenario, the task has to be done by a single individual. In this case, constraints towards the *qualifications* or *confidentiality* level of the worker have to be considered.

Creating and editing presentation slides is a task that can be perceived as a burdening additional workload to workers. In such a case, the *motivation* plays an important role to ensure the quality of the provided work. Here, the affinity of the worker towards the content is of importance, and intrinsically motivated workers have to be found, that like to work on the given topic or with the required tool. Such a task could also be used to improve the workers' capabilities to use specific office tools or to introduce them to certain information and processes within the company. Such a task cannot be assigned using a *direct assignment* of workers, but rather requires workers to apply for such a task and allow for review, rejection, and acceptance of applications. Therefore, this example motivates and provides requirements for the *open application* strategy provided in the next section.

From the empirical study, several scenarios relevant in the IT sector are identified and described. Two specific scenarios from the *productivity* category are chosen to describe two example tasks that focus on different aspects of task assignment strategies. From the identified requirements, task assignment strategies are provided in the following. Following the results of the qualitative study, the following task assignment strategies are derived from more than the two described scenario examples and take further identified requirements into account.

### 3.4 DESIGN ASPECTS OF ENTERPRISE CROWDSOURCING PLATFORMS

In this section, the conceptual design of an ECP with the focus on task assignment strategies is presented. Section 3.4.1 provides the basic structure, discussing briefly the relationships of basic elements on the ECP derived from the literature and the qualitative study. Section 3.4.2 focuses on the design of assignment strategies and presents the corresponding workflows for the management of the respective crowdsourcing processes motivated by the identified scenarios.

#### 3.4.1 Basic Structure of Enterprise Crowdsourcing Platforms

Figure 9 shows schematically how the elements of an ECP are structured. At the center of an ECP, there are the *tasks* and the *actors*, as well as the *contribution* towards the result of the crowdsourcing process. Depending on the interaction of the actor on the task, an actor can either be the *worker* of a task or the *requester* of a task. In a managed ECP, it could be necessary to define a *manager*, who needs to review created tasks before they are published on the platform. A user of an ECP can act in different ways on different tasks and is therefore seen as an actor, who can perform different actions, depending on his view on the task. Actors have three main attributes attached that describe their profile: the qualifications, the roles, and the motivators.

The *qualifications* describe the skills, knowledge, and characteristics of the respective actor and can be matched against the requirements of a task. The qualifications can describe characteristics such as age, department, confidentiality level, etc., which can be integrated from existing systems or queried during registration. Skills and knowledge, on the other hand, are more difficult to grasp and are subject to a con-

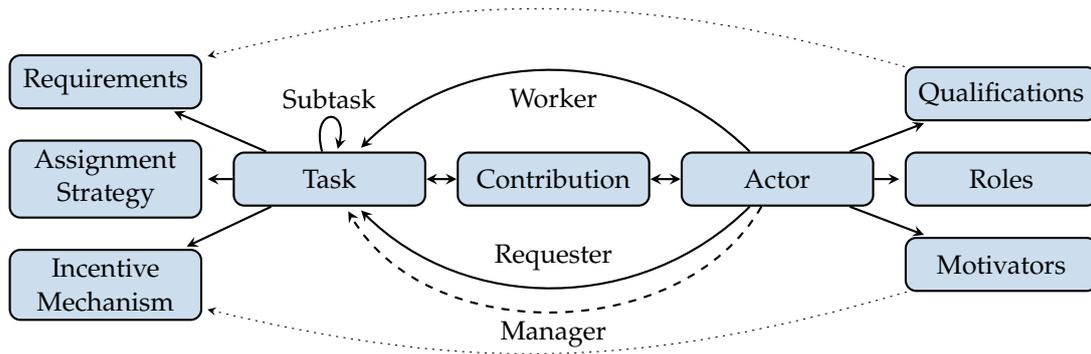


Figure 9: Basic structure of elements on ECPs.

stant change as a result of experience at work and in everyday life. They may be recorded at the time of registration, while a distinction should be made between non-verified data and verified data, e.g., via certificates. In addition, the actor's activity on the ECP results in changes in skills and knowledge. Executed tasks and corresponding feedback can serve to prove the skills and the level at which they are mastered.

Besides the qualifications, an actor's profile consists of *motivators* describing the motives of the user on the platform. Depending on those motivators, the incentive mechanism of a task can be adapted. In addition, at least one *role* is assigned to each actor, which defines the general rights of the users on the platform. Such roles include the rights to work on tasks, request tasks, manage tasks or provide administration rights for certain parts of the platform.

A task can be broken down into further *subtasks* as shown in Figure 9. In order to perform the composition of results from a decomposed task, this relationship has to be managed carefully. A task contains information about the specific *requirements* that are necessary to fulfill the task. In addition, each task adheres to exactly one *assignment strategy* with which it is offered to the crowd. The *incentive mechanism* can be dependent on the task and also adapted to the motivators of its target workers.

The core task of the platform is to provide the actors with the tasks that are appropriate for them. For this purpose, various filter and matching processes are used, which are described in Section 3.4.2. These reduce the crowd to the actors who are suitable for a task, based on the qualifications that match the requirements of the task but also based on the motivators of an actor. However, it is necessary not to restrict possible assignments too much, as one main motivator in ECPs is to gather further skills [83], also described in one of the interviews:

"... you want to expand your skills in some way [...] you take on tasks that perhaps go beyond your own qualifications, to get this competency into your profile."

User, Company C, Head of Product Management IT SBU 2

### 3.4.2 Design of Task Assignment Strategies and Workflows

The scenarios identified and the information collected serve as a basis for the distinction of task assignment strategies and for the design of corresponding workflows. An assignment strategy describes general conditions and goals in which a task is offered to possible workers. An assignment workflow describes different steps and dependencies to assign a task to one or many workers. Different scenarios demand identical assignment strategies, and therefore, workflows can be used in different scenarios depending on the required assignment strategy or application.

This section describes the two different assignment strategies *direct assignment* and *open application*, which differ from the *self-selection* assignment strategy used widespread in micro-task markets and the corresponding workflows. From the coded interview results, task assignment strategies were often found to include a *crowd selection* workflow. As this is a shared process between the different task assignment strategies, crowd selection is described as an independent process since this workflow is identified as part of different assignment strategies. Therefore, the crowd selection workflow is presented first, while the descriptions of the *direct assignment* and *open application* strategies follows afterward.

#### 3.4.2.1 The Crowd Selection Workflow

In enterprise crowdsourcing, the set of workers is equal to all employees. In particular cases, it is extended by members of partners or clients. Previous sections discussed the relevance of qualification, confidentiality, and motivation in enterprise crowdsourcing. In many cases, the general set of all workers is restricted to assignable workers. This means, before an assignment of a task to one or many workers can be executed, the possible crowd has to be filtered. This step of selecting workers is called crowd selection. Selecting assignable workers from the crowd is described as a three-step process, which is visualized in Figure 10.

- o. At first, all workers are judged to be relevant for selection.
1. In a first filtering step, the qualifications of the workers are matched with the requirements of the task. To not restrict the selection too much, a task can include must-have and nice-to-have requirements. The result of the first step is a list of workers that are capable of solving the task without further restrictions.
2. A second filter takes the availability and the confidentiality level of the workers into account. This optional step returns a list of workers that are capable of, available for, and have the clearance for solving the task.
3. The third and last filter is concerned with the motivation of the selected worker. The target is to identify workers that are not only capable of working on the task but have a particular interest in solving the task. This filtering step leads to the final set of chosen workers.

Though the last two filtering steps are optional, they are considered a crucial element in order to maintain a sustainable platform, as they provide driving factors for the acceptance of the platform.

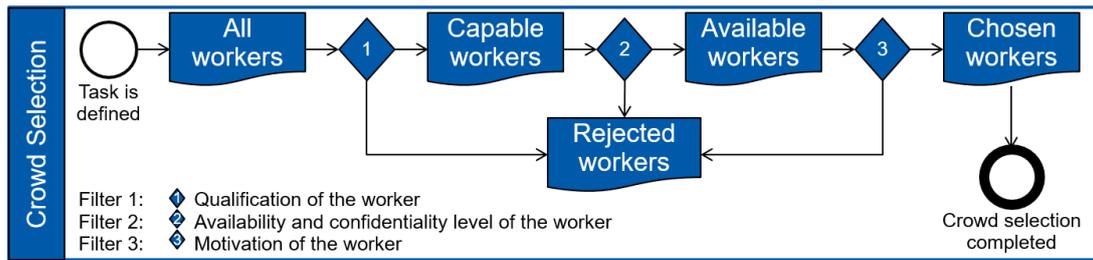


Figure 10: Crowd selection process.

### 3.4.2.2 Assignment Workflow for the Direct Assignment of Workers

The first assignment strategy, which is related to the scenario of GUI tests is called *direct assignment* (see Figure 11). The goal of the *direct assignment* strategy is to reach many users without caring too much about single contributions. Therefore, the corresponding workflow assigns the tasks to the selected crowd (see Section 3.4.2.1) and mainly focuses on the concern to reach a critical number of contributions. As long as this number can still be reached, there is nothing to do but wait for the contributions. When this number cannot be reached anymore, the requester needs to take action in order to get the task done (i.e., adapt the crowd selection or allow the system to publish the task also externally). The selected crowd is also applicable to form a user group in order to provide the tests to the same crowd a second time. This can also benefit the efforts of building proper community management.

The workflow for this strategy does not require any application of workers. After creating the tasks and filtering the crowd, the selected workers are directly assigned to the task. That means they will find a new task assigned to them on the platform for execution.

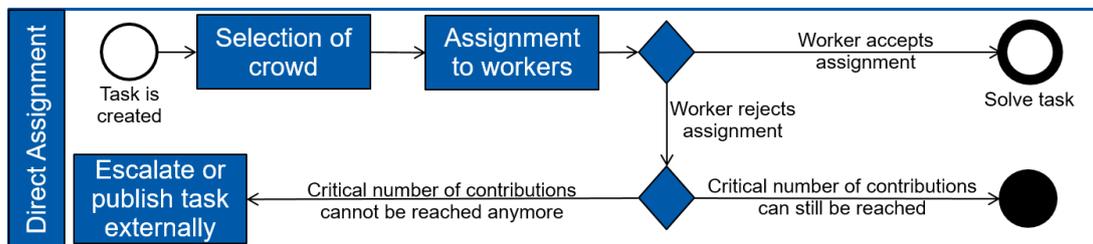


Figure 11: Direct assignment workflow to reach a broad crowd.

### 3.4.2.3 Assignment Workflow for Open Applications of Workers

The second assignment strategy *open application*, is related to the scenario of creating presentation slides, described before. The goal of the *open application* strategy is to find a single qualified worker. The scenario requires an iterative approach for reviewing applications and republishing the task (see Figure 12). After creating the task and filtering the crowd, workers can apply for the task. The requester is directly involved in the selection process, by reviewing the workers that applied for the task.

The selected worker submits the solution, and the requester either accepts the solution or reopens the task with additional feedback for the worker. This process can be repeated until the expected result is reached. If there are no applications or every application has been rejected, the requester needs to lower the requirement and constraints of the task or find other means to solve it.

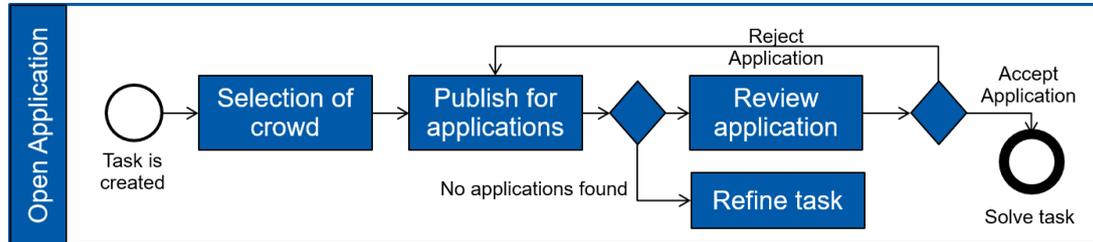


Figure 12: Open application workflow to assign a specific worker.

### 3.5 USER EVALUATION OF SELECTED WORKFLOWS

The task assignment strategies and the designed corresponding workflows are distinguished based on the qualitative study performed. Additional aspects and requirements for the design of information systems are often perceived and uttered by users only after they gain practical experiences in the use of the system. Therefore, a prototypical platform was implemented that realized the issues and especially the task assignment workflows presented above. It has been used to evaluate the workflows by means of a small user evaluation. The goal of the evaluation was to realize additional aspects and requirements related to the assignment strategies and workflows which have not been mentioned in the study. Two individual users that have been part of the initial study, followed specific guidelines in order to recreate the scenarios and example tasks provided above and discuss possible shortcomings of the prototype. Therefore, sample tasks and artificial workers were provided in order to simulate a running system. Each of the users went through both of the example tasks from the viewpoints of workers and requesters.

The figures provide a brief insight into the implemented platform, which was available through a browser in different environments (e.g., desktop, tablet, or smartphone). Figure 13 shows the view of a requester who can manage tasks that can be listed in different states (at the bottom of the page). Here, it is possible to review

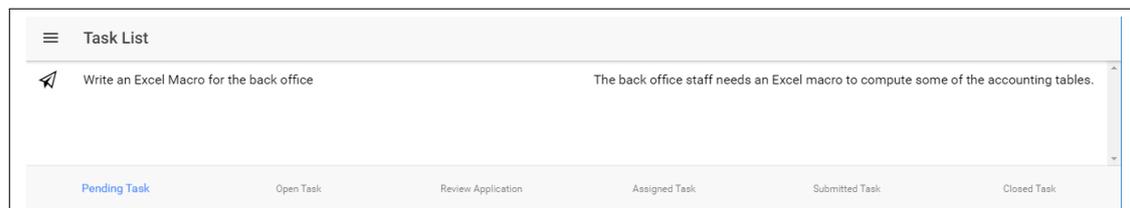


Figure 13: Platform interface with example task (menu is closed).

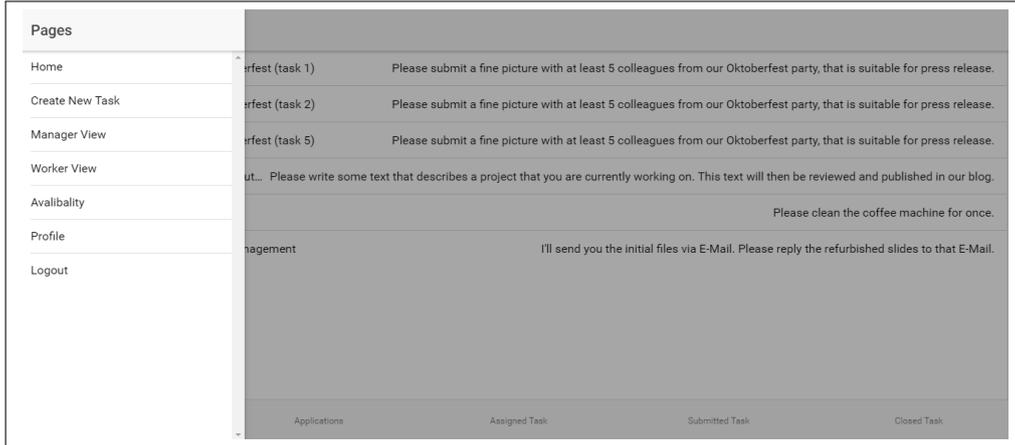


Figure 14: Platform interface with example tasks (menu is opened).

pending or open tasks, review applications to tasks (as required in the *open application* workflow), and also to review assigned, submitted or closed tasks. Figure 14 shows the view of a worker who has several possible tasks displayed. The menu is opened in this screenshot (left margin). Within the menu, the user can switch from his current view to the *manager view* or the *worker view*. In the menu, there is also the possibility to create new tasks, or change the profile information as well as changing availability settings.

Regarding the *crowd selection* workflow, it was found, that it is crucial to have a manageable system for defining and assigning qualifications correctly. Without a very well defined taxonomy of qualifications, the selection of required qualifications is a difficult task when creating a new task. The qualification of a worker could be derived and ranked from their task history. However, the required qualification for a task should be creatable on the fly when creating tasks in the system. On the other hand, it is also possible to introduce a system, that derives requirements from a textual description and match workers respectively. Another mentioned aspect is, that the process of filtering the crowd should be provided as transparent as possible to the task requester. A system is suggested, that provides, e.g., a worker count of the currently filtered crowd, in order to let the task requester know what kind of consequences specific requirements have for the visibility of the task. One can imagine a system, where the requester is able to explore the selected crowd, by comparing user profiles. However, privacy concerns of the employees must be considered in such a scenario. Another important feature of the crowd selection process is to save a filtered crowd to enable proper community management especially when it comes to testing-scenarios.

For the *direct assignment* workflow, the evaluators mentioned that for the worker, the assignment of tasks should be made as transparent as possible. The knowledge of the reason for an assignment might affect the motivation of the worker to accept or reject the task in a positive, as well as in a negative manner.

Within the scope of the *open application* workflow, the evaluation showed, that it is particularly well designed for tasks that require a single contributor. It is easily possible to use an open application based workflow on tasks with several contributors; however, the review process for the requester provides much overhead in such a case. Within the *open application* workflow, it is also possible to define a manager for a created task when, e.g., the decision on who should work on the task is not done by the task requester himself but rather by a supervisor.

One critical review noted that workers that are excluded from certain tasks in either workflow have no knowledge of this exclusion and are not able to develop their profile for future inclusion, which might be beneficial in an enterprise-wide scope. This is a design decision, which is particularly necessary when considering the confidentiality of tasks. However, it is possible to include less strict visibility configurations, that provides solutions for low confidential tasks.

### 3.6 CONCLUSION

The goal of this chapter was to identify requirements towards task recommendation systems (RG<sub>1</sub>) and to design task assignment strategies (RG<sub>2</sub>) in ECPs.

Against this background, a structured literature review and a qualitative study were conducted, which highlight scenarios and requirements for the design of task assignment strategies in ECPs. The provided task assignment strategies for *direct assignment* and *open application*, as well as the corresponding workflows, serve as extension on the conceptual model of ECPs and therefore extend the existing strategies in the literature. The results are presented by highlighting conceptual decisions on the design process of an ECP while focusing on approaches related to new concepts. This allows enterprises to use the presented methods and workflows to apply them in their own design of an ECP directly. The motivation and background of these methods are clearly stated and critically discussed. The user evaluation performed on two workflows highlights some limitations of the work and displays new aspects which should be considered in future work.



GOALS of this thesis, as described in Chapter 1, are the definition of requirements towards task recommendation (RG<sub>1</sub>) and the design of assignment strategies (RG<sub>2</sub>). Micro-task markets and especially PCPs rely on the task self-selection of workers as discussed in Chapter 2. Different criteria could be used for recommending tasks to workers in public micro-task markets. On the one hand, factual attributes of tasks could be used as recommendation criteria, such as payment and required time, which are often already provided as assignment filters for the workers on the platform.

On the other hand, semantic attributes of tasks could be leveraged, providing detailed information about the task to be performed. As the provided filters often reflect the perspective of the platform provider and requesters, a challenge for task recommendation and respective assignment strategies is the worker's perspective towards task recommendation. Therefore, this chapter describes two user studies conducted on a public micro-task market, which analyze the demands of the workers towards task recommendation. The first user study provides insights into what kinds of recommendation criteria are relevant for the workers [146]. The second user study focuses on semantic attributes, analyzing how similarities of tasks are perceived from the workers perspective [118, 144].

This chapter is structured as follows. The next section highlights related scientific work. Afterward, in Section 4.2, the demands of workers towards task recommendation are discussed. Section 4.3 presents the most relevant similarity aspects of tasks as perceived by the workers on a PCP.

#### 4.1 RELATED WORK FOR USER STUDIES IN PUBLIC CROWDSOURCING PLATFORMS

Different user studies analyze the behavior and opinions of workers in crowdsourcing platforms. This section provides an introduction into the methodologies and results of related work on user studies in crowdsourcing platforms.

Some studies focus on the motivations of the workers why to engage in crowdsourcing and why they choose certain tasks over others. Brabham [24] identifies several different motivators for workers to engage in crowdsourcing. Kaufmann et al. [83] focus on micro-task markets and analyze extrinsic and intrinsic motivators for task choice preference. Other studies provide surveys based on directly asking the workers for their opinions.

Chilton et al. [30] analyze how workers search for tasks on the public micro-task market *MTurk*. Therefore, they analyze the provided search and sort filters such as

*reward (largest), time (newest), and title (a-z)*. Providing the results from 257 unique workers, they find that the workers focus on newest hits and “look mostly at the first page of the most recently posted tasks.”

Schulze et al. [148] investigate the properties of tasks and how they influence the task selection preferences considering the different demographic backgrounds of the worker. They analyze properties such as the length of task descriptions, high reward per hour, high reputation of requester, and more. Their findings provide insights on differences between American and Indian workers, and show that many workers like “short and simple tasks, while others prefer comprehensible task descriptions.”

Yuen et al. [184] provide a worker study where the workers are directly asked about their selection criteria. Based on the answers of 100 workers on *MTurk* they find that the *amount of pay* is the most relevant selection criterion followed by the *task nature, level of difficulty* and *duration*.

The user studies provided so far focus on task selection and task choice preference. None of the user studies introduce a recommender scenario for the workers to consider. Therefore, the presented studies in this thesis introduce the idea of recommendation and provide insights about preferred task recommendation criteria. Additionally, from the insights of the first study, the second user study analyzes the details of how workers perceive similarities of tasks and provides a list of the most preferred similarity aspects of tasks.

## 4.2 WORKERS PREFERENCES ON TASK RECOMMENDATION

This section provides the methodology and the results of a user study with the votes of 130 workers towards their preferred selection criteria of tasks on the platform. The qualitative and quantitative survey among crowd workers focuses on task selection and recommendation and takes into account the demographic characteristics of the workers. The provided results support the hypothesis that the preferences of workers are inhomogeneous and that criteria relevant for selection are not available for filtering the tasks in the current platforms.

### 4.2.1 Methodology

The study focuses on the question which characteristics of tasks are most relevant to workers with respect to the recommendation of tasks. The survey first introduces the idea of task recommendation to the workers. Then the workers are supposed to select their most important recommendation criteria. After selecting recommendation criteria, the workers are required to arrange all recommendation criteria according to importance. The recommendation criteria include six standard measurements, like payment, as well as additional criteria that consider the similarity of tasks. Six factual criteria are considered, namely the *most money*, *least time*, *payment per time*, *time to rate*, *best-rated task* and *best-rated requester*. Additionally, three criteria considering the similarity are provided: *similar task*, *different task*, *similar worker*. The *time to rate* describes the time a worker is required to wait until the work on the task is verified

and the money transferred. The *similar/different task* criteria describe a task which is similar or different to the last completed task of the worker. The *similar worker* recommendation criterion describes tasks that have been completed by workers which are found to be similar to the worker receiving the recommendation. The task was made available on the PCP *Microworkers*. The survey features a structured questionnaire

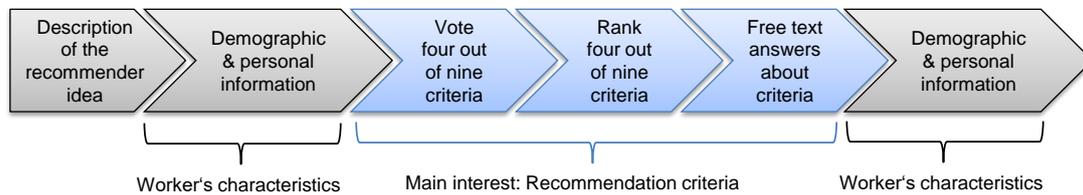


Figure 15: Survey design: five sections [146].

including six sections, shown in Figure 15. Within the questionnaire, the workers are provided with a description of the idea of task recommendation and have to answer questions about characteristic personal information considered later. The main part has the workers vote for the four most important criteria from a set of nine and asks them to order the whole set according to importance afterward, also giving them the opportunity to express further opinions given a free text field. The last part asks for more or different personal characteristics of the worker, including *consistency questions* [73] to filter out spammers.

The survey was executed on the PCP *Microworkers* in April and May 2015, leading to 130 submissions that are taken into consideration for data analysis. The analysis focuses on the characteristics: region, gender, experience, average payment, and activity, to find differences between the workers.

#### 4.2.2 Results

A single survey submission is filtered by comparing the answers of voting (step three) and ranking (step four), applying the methods of the consistency questions. Only such votes that are consistent in voting and ranking are considered for the data analysis. The votes for each of the criteria are weighted by the rank it was voted on (1 – 4), where a higher rank corresponds to a higher weight. For each of the criteria, the weights are summed up and divided by the overall sum of weighted votes. This *average weighted ranking (awr)* gives a value between 0 and 1 for each of the criteria, where a higher *awr* hints towards higher valued recommendation criteria. The overall votes for the criteria and the corresponding *awr* is given in Table 8.

Analyzing the results shows, that the workers unsurprisingly value the criteria *most money* and *highest payment per time* the most. However, the criterion *similar tasks* comes in third place being more valuable to workers than criteria like *time to rate* or *best-rated task*. This shows that there is a requirement for identifying similar tasks in order to be able to recommend them to workers on such platforms. Additionally, *different task*, is the least valued of the criteria, showing that workers want to be provided with similar tasks.

Table 8: Overall votes and *awr*.

CRITERIA	AWR	RANK			
		1	2	3	4
most money	0.242	37	17	13	12
payment per time	0.169	19	20	9	12
similar task	0.147	18	12	11	14
least time	0.124	6	19	17	7
time to rate	0.120	5	16	21	8
best-rated	0.068	5	7	8	10
similar worker	0.051	4	7	4	5
best-rated requester	0.043	6	1	4	7
different task	0.035	2	3	6	5

Table 9: Number of submissions and counted votes per region.

REGION	COUNTRIES	SUBMISSIONS	VOTES
Asia	BD, NP, PH	37	104
EU - West	FR, DE, ES, IE, IT, NL, PT, SE	45	133
Western	US, UK, CA, AU	48	140

While the analysis dividing the workers by gender, experience, payment or activity did not result in significant insights, analyzing the different regions shows significant differences between workers. The regions are chosen as defined by the platform *Microworkers*, where a task can be created to be published in a specific region only. The submissions were divided into the different regions *Asia*, *EU-West* and *Western* as seen in Table 9. Figure 16 shows the differences between the regions for each of the criteria. It can be seen, that the results of the *EU-West* region almost mirror the overall results. However, the opinions of the workers from the *Asia* region and those from the *Western* region differ significantly, especially when it comes to the recommendation criteria of *similar tasks*. In the *Asia* region, this is the least valued recommendation criteria, while in the *Western* region this is the most valued recommendation criteria. Accordingly, *different task* is ranked sixth out of nine in *Asia* compared to the last rank in *EU* and *Western* regions.

#### 4.2.3 Conclusion

This section provides the results of a survey undergone with workers on the PCP *Microworkers*. Analyzing the votes of 130 workers towards their preferred recommendation criteria, it showed that overall, workers value the criteria of *time* and *money* most. However, criteria that are not so strongly defined as the similarity of tasks

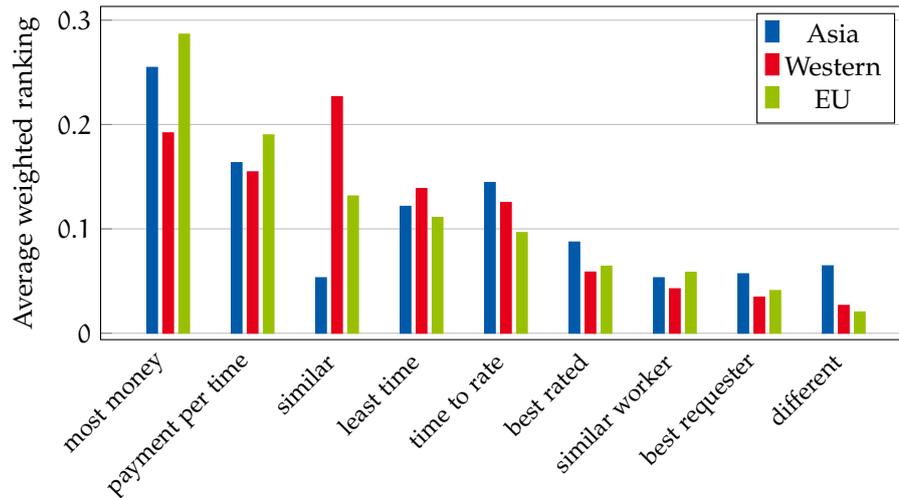


Figure 16: Results with respect to workers' region [146].

are also valued high. Especially the *similar task* criteria, which is found to be most valuable for workers from the *Western* region and appears to be least valuable for workers from the *Asia* region provides interesting recommendation criteria and has to be analyzed in a more detailed study.

#### 4.3 SIMILARITY OF TASKS FROM THE WORKER'S PERSPECTIVE

Section 4.2 showed that the *similar task* is a relevant recommendation criterion for workers. How the similarity of tasks is defined remains unclear. Therefore, how workers view and perceive the similarity of tasks is the focus of the user study presented in the following. The survey is designed describing several different similarity aspects of tasks and has workers decide on how useful each of the aspects is in order to decide on the similarity of two tasks. To determine the differences between regions the survey gathers the opinions of 100 workers from five different world regions each. The results provide a strong claim towards task recommendation and certain similarity aspects based on the 500 submissions in total.

##### 4.3.1 Methodology

The survey comprises of four main parts shown in Figure 17. The first part explains the motivation of the survey and introduces the field of task recommendation and similarity aspects. The second part asks the workers to provide some details about the demographics as well as the experience and activity of the worker. This data is compared with data given on the platform and used as *consistency question* [73] to identify spammers. The third part of the survey provides the main questionnaire. The main part presents 14 questions examining the worker's opinion about 14 similarity aspects of tasks. For each of the similarity aspects, the workers are provided with a

five point Likert scale to judge how useful the similarity aspect is. One example is given in Figure 18. Additionally, a further test is introduced to identify spammers. Therefore, a 15th nonsensical test question is introduced to the 14 original questions, which states “Task B is completely jabberwocky<sup>1</sup> to Task A”, providing the instruction in the description to answer this question with ‘very useful’. This simple test question allowed to filter about 47% of all survey submissions, which have not been answered with care and the instruction to answer ‘very useful’ was not followed. In most cases, a wrongly answered test question in the main part correlated with the results from the consistency questions.

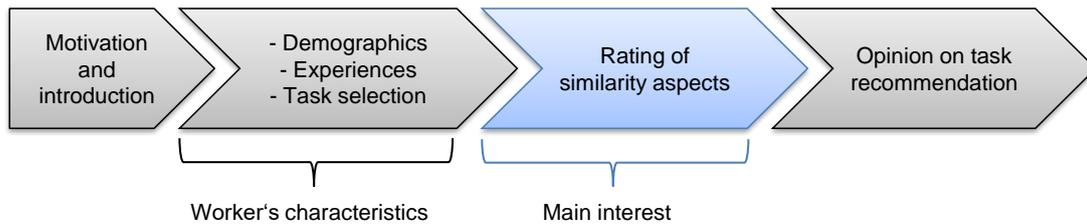


Figure 17: Four parts of the survey.

The 14 similarity aspects include different kinds of aspects. The first five similarity aspects represent semantic similarity measures that could be derived from the task descriptions and are therefore referred to as *semantic*. These are included to determine whether semantic similarity measures are required by the workers. These semantic similarity aspects include the questions whether two tasks can be described as similar when they come from the same *domain*, require the same *action*, have the same *complexity*, feature similar *comprehensibility* or serve the same *purpose*. Six of the similarity aspects cover *basic criteria*, where similar tasks are described to offer the same *payment*, require the same amount of *time* for completion, have the same *payment/time* ratio, take the same *time to rate*, have the same *success rate* or have the same *number of open tasks* in their campaigns. The last three attributes describe *employer attributes*, where two tasks are published from task requesters with the same *employer experience*, who come from the same *employer country* or can be categorized into the same *employer type* such as commercial or scientific.

**Task B comes from the same domain as Task A.**

not useful at all      very useful

(Domains are for example: fashion, social networks, psychology, smartphone apps, videos.)

Figure 18: One example question from the main part of the survey [144].

The survey was published on the PCP *Microworkers* in November and December 2015. It was published in five different regions as shown in Table 10 which also shows the spam rate within the regions as well as the number of valid submissions used for data analysis per country of origin of workers.

<sup>1</sup> The word “jabberwocky” represents the nonsensical nature of the question [28, 182]

Table 10: Submissions per region.

REGION	SPAM RATE	RESIDENCE COUNTRY (SUBMISSIONS)
Asia, South	63%	BD(77), IN(13), LK(6), NP(3), PK(1)
Asia, South East	52%	ID(34), MY(28), PH(27), VN(6), SG(3), TH(2)
English speaking	35%	US(62), UK(21), CA(13), AU(4)
Europe, East	35%	RS(34), RO(12), MK(12), BA(11), BG(10), HR(7), PL(4), LT(3), AU(2), TR(2), SI(1), HU(1), CZ(1)
Europe, West	42%	IT(28), BE(19), FR(16), PT(14), ES(9), DE(6), FI(3), CH(2), IE(1), DK(1), AT(1)

#### 4.3.2 Results

The results section first provides the results of the last part of the questionnaire, whether or not the workers want task recommendation. Then, the overall results are discussed. The different results per region are presented briefly afterward.

##### 4.3.2.1 Necessity of Task Recommendation

The workers answered the two questions “Do you think it is easy to find tasks that are interesting and enjoyable to work on?” and “Would you like to receive task recommendations on the platform?”. They had the chance to vote ‘yes’, ‘no’ and to give no statement ‘ns’. 61.2% of the overall workers answered for the first questions that it is easy to find interesting tasks, 33.0% found it not to be easy, while 5.8% gave no statement. Figure 19(a) shows the different results per region. There is quite a difference between the regions, where 78% of workers from the *Asia, South* region voted for ‘yes’, while only 42% voted ‘yes’ in the *Europe, West* region. The answers to the second question, whether task recommendation is wanted on the platform is answered with ‘yes’ by 74.6% of all workers. Again the answers differ per region, however, at least 65% of the workers in any region state a demand for task recommendation. These results provide a strong motivation to further support workers on the platforms by using task recommendations.

##### 4.3.2.2 Overall Results on Worker's Preference for Task Recommendation

The overview of the results given in Figure 20 shows for each similarity aspect all votes on the Likert scale. According to this survey, the aspect *required action* is rated the highest and thus represents the most important similarity aspect for workers. The ratings are weighted from ‘0’ for ‘not useful’ to ‘4’ for ‘very useful’, to compute an overall average value given in Table 11. All aspects are rated with a mean value above 2.0, and thus fundamentally positive, except for the *employer country* (1.82). Though the average values show preference tendencies, Figure 20 provides a more detailed understanding of the voting. For example, the averages of *comprehensibility* and *domain* differ only insignificantly with 2.97 and 2.87 respectively. However,

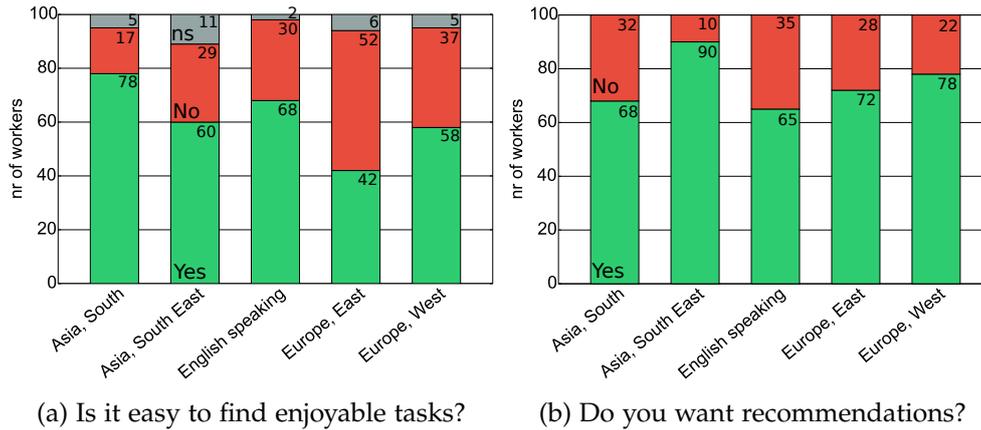


Figure 19: Necessity on task recommendation [144].

Figure 20 shows, that *comprehensibility* gained this through more votes for 'very useful' and less neutral votes.

For each region and similarity aspect, the average value and the rank in the corresponding region are given in Table 11. The similarity aspect *required action* with an average rating of 3.41 is clearly voted to the first rank. The similarity aspect *required action* holds this first rank in all regions. To quickly find the five best-rated aspects per region in the table, they are printed in bold.

As explained earlier, the similarity aspects are divided into different kinds of aspects: *semantic*, *basic criteria* and *employer attributes*. In the overall results, the five *semantic* similarity aspects occupy the first five ranks. After the *semantic* similarity aspects follow the *basic criteria* and then the *employer attributes*, with only one exception (*nr. of open tasks*). This proves the hypothesis that semantic similarity aspects, which could be derived from the task descriptions, are relevant for the workers on the platform.

#### 4.3.2.3 Results Depending on Regions

Table 11 shows the differences between the regions in detail. Some significant differences (rejection of the null hypothesis that the samples come from the same population with at least  $p < 0.05$ ) are discussed below. The *Asia, South* region differs most from the other regions in its voting behavior. Their workers voted the similarity aspects *success rate*, *employer type* and *employer experience* on the ranks two, three and five, while these aspects can be found in the overall ranking only on ranks eight, eleven and twelve. Also, the aspects of *open tasks*, *employer experience*, *employer type*, and *employer country* follow this pattern of higher valuation in *Asia, South*.

Furthermore, workers from the region *Asia, South East* rate the similarity aspect *same domain* significantly higher than workers from *English-speaking* countries or *Europe, West*. In the Asian regions, the *purpose* of the task is a more important similarity aspect for the workers, than for workers from the region of *Europe, West*. This also applies to the aspects *payment* and *time to rate*. The aspect of *time* is less interesting

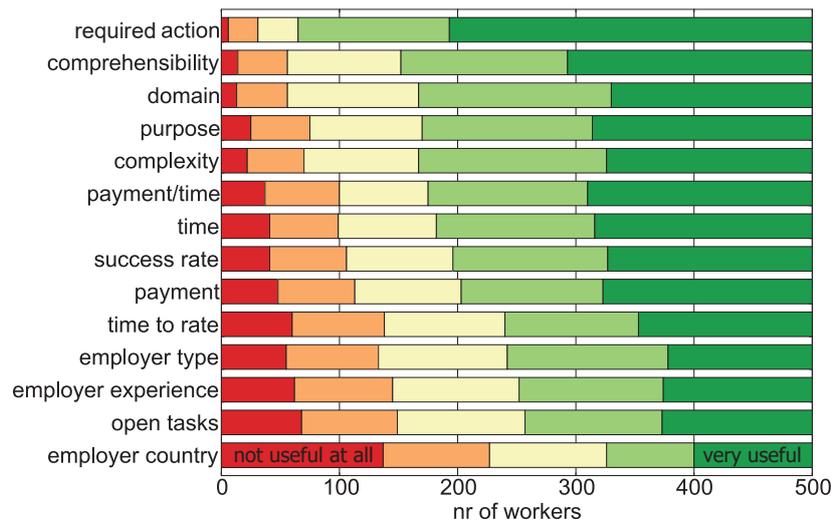


Figure 20: The similarity aspects judged on a Likert scale from “not useful at all” to “very useful” by 500 workers (ordered by average rating) [144].

for workers from *Europe, West* than for workers from the Asian and English-speaking countries. It is remarkable that for the aspects *required action*, *complexity* and *payment/time* no significant differences can be found between the regions.

#### 4.3.3 Conclusion

The main results, shown before, provide several valuable insights. On the one hand, the demand for task recommendation is clearly stated by workers from all five investigated regions. On the other hand, the most valuable similarity aspects as perceived by the workers are identified. The study shows that the similarity aspect *required action* is valued the most throughout the whole community of workers. In general, the semantic similarity aspects which could be derived from the textual descriptions are voted the highest, followed by basic criteria and employer attributes. The exact values for the similarity aspects vary between the regions; however, the four *semantic* similarity aspects of *required action*, *comprehensibility*, *domain* and *purpose* can be found within the top seven (out of 14) similarity aspects in every region. This is quite surprising regarding the fact that platforms often only provide filters concerned with *basic criteria* like *required time* or *payment* of tasks.

Table 11: Similarity aspects with average rating and rank.

SIMILARITY ASPECT	AVERAGE RATING AND (RANK) BY REGION											
	OVERALL		ASIA S.		ASIA S. E.		ENGLISH SP.		EUROPE E.		EUROPE W.	
required action	3.41	(1)	3.25	(1)	3.43	(1)	3.52	(1)	3.36	(1)	3.49	(1)
comprehensibility	2.97	(2)	2.96	(6)	3.16	(2)	3.07	(2)	2.79	(4)	2.87	(2)
domain	2.87	(3)	2.92	(7)	3.13	(3)	2.73	(7)	2.87	(2)	2.69	(4)
purpose	2.83	(4)	2.99	(4)	3.05	(5)	2.84	(4)	2.74	(5)	2.54	(6)
complexity	2.83	(5)	2.74	(13)	2.97	(6)	2.89	(3)	2.81	(3)	2.74	(3)
payment/time	2.76	(6)	2.83	(11)	2.83	(10)	2.79	(5)	2.73	(6)	2.60	(5)
required time	2.72	(7)	2.87	(8)	2.97	(6)	2.78	(6)	2.62	(8)	2.38	(7)
success rate	2.66	(8)	3.10	(2)	3.13	(3)	2.40	(9)	2.47	(9)	2.20	(8)
payment	2.63	(9)	2.84	(10)	2.88	(8)	2.53	(8)	2.71	(7)	2.17	(9)
time to rate	2.42	(10)	2.81	(12)	2.83	(10)	2.26	(10)	2.08	(13)	2.11	(10)
empl. type	2.38	(11)	3.02	(3)	2.70	(13)	2.16	(11)	2.22	(10)	1.82	(11)
empl. experience	2.33	(12)	2.97	(5)	2.84	(9)	1.92	(13)	2.20	(12)	1.74	(12)
nr. of open tasks	2.31	(13)	2.87	(8)	2.73	(12)	2.06	(12)	2.22	(10)	1.65	(13)
empl. country	1.82	(14)	2.60	(14)	2.37	(14)	1.38	(14)	1.41	(14)	1.34	(14)

THIS chapter covers the works on task recommendation discussed and motivated in the introduction. Covering the broad spectrum of tasks as shown in Figure 21, this chapter provides contributions for three different kinds of tasks: micro-tasks, projects, and job postings. One challenge dealt throughout the presented approaches is derived from the dynamic nature of tasks. The high churn of tasks requires the application of content-based recommender systems and the definition of appropriate methods to determine similarity. The following sections provide task recommendation approaches based on textual descriptions towards RG3. Thereby, similarity-based approaches are presented for providing task recommendation, crowd selection, skill extraction and the preselection of tasks, as shown in Figure 21.

For the area of micro-tasks, two different scenarios are focused. At first, the similarity aspect *required action*, considered highly relevant by workers (compare Section 4.3), is used as a basis for designing a method to determine task similarity. This approach evaluates verb phrase similarities in order to rank and recommend tasks for workers. Secondly, an approach is presented which provides a selection of workers for task recommendation in location-based micro-task markets. This crowd selection approach is used to investigate, whether similarities in task descriptions can improve the recommendation of collaborative filtering approaches.

Related to projects, a scenario from the industry is considered to improve skill extraction approaches in this scenario based on the distributional similarities of words. For the assignment of projects to workers, a small dataset is given providing project descriptions in worker profiles as well as project proposals stating required skills. Applying a simple recommendation scheme for matching required skills and provided skills allows evaluating the performance of the presented skill extraction approach.

For job recommendations, the interactions of users with job postings are analyzed. Therefore, this scenario relies on the click data of users on a job market platform. Projecting users of the platform into the vector space representation of the job posting's

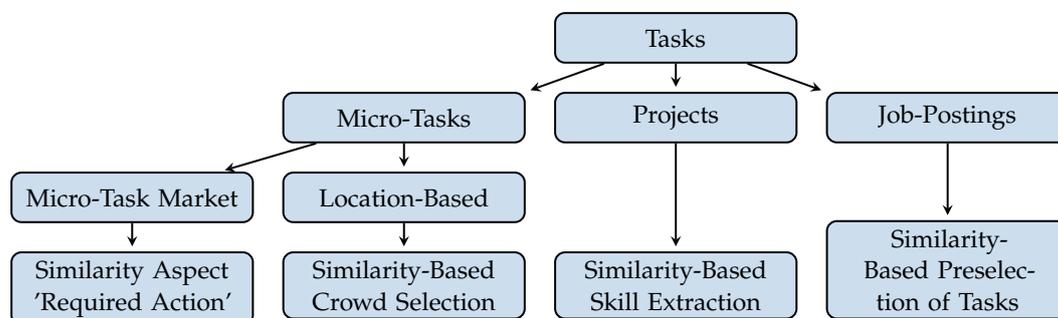


Figure 21: The spectrum of tasks from job-postings to micro-tasks.

documents enables a preselection of job postings derived from their textual similarities. The cluster-based preselection improves the recommendation performance and reduces the runtime of applicable recommender approaches.

## 5.1 RELATED WORK FOR TASK RECOMMENDATION

Recommender systems have been studied for many applications and domains. A general overview of recommender systems is provided by Bobadilla et al. [20], who discuss the foundations, problems, and evaluation of recommender systems. Ricci et al. [131] gather in their work many case studies and practical recommender approaches. They consider recommendations in domains like technology-enhanced learning Manouselis et al. [105]. Erdt et al. [42] provide an extensive study on the evaluation of such systems. Another domain focuses on active learning as provided by Rubens et al. [136] and Schnitzer et al. [147].

The problem of *churn* is described by Burke and Ramezani [26]: “A high churn domain is one in which items come and go rapidly,” who mention news items as an example. They also describe that “Items that have been around for some time may accumulate ratings, but by the time they do, they may no longer be relevant.” This shows, that in domains considered to have high churn, items still are expected to get a number of ratings. As described before, the dynamic nature of task markets provides an extreme form of the churn problem, where items are no longer relevant, as soon as they gather a single interaction.

### 5.1.1 Recommendation of Micro-Tasks and Task Classification

Geiger and Schader [51] provide an overview of the different approaches regarding personalized task recommendation in crowdsourcing systems. They also identify the high churn problem for crowd processing systems as a challenge for collaborative filtering approaches. They claim, that “an efficient personalized task recommendation should therefore make (additional) use of content-based techniques.” Accordingly, task recommendation approaches have always considered the contents of the tasks.

One approach to task recommendation has been proposed by Yuen et al. [185] by developing a task recommendation framework, which considers the task attributes, the worker performance and the history of the worker’s completed tasks. For their task properties, they include factual features, such as the reward and allotted time, but also semantic features such as title, category, and keywords. Yuen et al. [185] describe their “TaskRec” approach to be a collaborative recommender system. Including the information about the categories of the tasks, they come up with a worker-task-preference matrix and a worker-category-preference matrix. Intersecting the matrices and applying probabilistic matrix factorization, they identify recommendable tasks based on the given categories. This means they reduce the knowledge about tasks to given categories and compute a rating for each category. From the task history of the user, they infer a five-star rating system. They assign 5 stars if a worker completed a task successfully, 4 stars if a worker submitted a task but the result was

rejected, 3 for completing a task without getting accepted or rejected, 2 for selecting a task but not completing it, 1 for browsing the task details but not selecting it, and 0 for never clicking on the task. The evaluation is performed on a dataset derived from the NAACL 2010 workshop [117].

Another approach which is focused on the history of a single worker is proposed by Geiger [50]. He also takes categories (tags or keywords) provided by the platform into account and introduces features of the employer into his preference estimation model for recommendation. Geiger [50] performs an online evaluation using a meta-recommender spanning several different micro-task markets on PCPs.

Assadi et al. [8] approach the field of task recommendations from a requester point of view. Maximizing the number of assigned workers using a fixed budget for heterogeneous tasks, they support the decision on which worker to assign to which task. This is based on optimizing task assignment in a scenario where workers place bids of their accepted payment for a task.

An also requester focused approach is provided by Mavridis et al. [109] who apply a taxonomy based skill modeling approach to optimize task assignment quality. They provide very promising results, where they are able to increase the number of correct answers significantly. However, their approach requires very specific knowledge about the requirements of the tasks and the skills of the workers and optimizes the task assignment in a system-wide fashion. The results can be used to identify requirements of tasks and qualifications of workers, which provides the basis Mavridis et al. [109] presume.

One approach provided for task recommendation in crowdsourcing systems is described by Ambati et al. [4], who also evaluate their work on a subset of the NAACL 2010 corpus [117]. They use a content-based approach including the task description by applying a simple BoW scheme. The proposed scheme computes the similarity as the overlap in the vocabularies of different tasks. Evaluating the ranking of the BoW approach against a classification based recommender trained on data about reward, timestamp and number of associated tasks, they report, that the similarity-based recommender performs best. The two methods proposed and compared by Ambati et al. [4] use a classification approach as well as an approach based on semantic similarities, where the former is outperformed by the latter in their offline evaluation. Their classification approach mainly relies on factual features. Their semantic approach uses a simple BoW technique to compute similarities in the vocabulary of task descriptions and rely on a binary task preference model.

Arora et al. [7] present a classification approach for questions which have been posted to the question and answering system *StackOverflow*<sup>1</sup>. They utilize *n-gram* counts for a Naive Bayes classifier to classify questions as *good* or *bad*, depending on their scores provided by the community.

In a very similar domain to micro-tasks, Schnitzer et al. [147] and Schmidt et al. [140] use a TF-IDF based approach and an ensemble classifier in order to classify job offers to improve the results of a job search engine.

---

<sup>1</sup> [www.stackoverflow.com](http://www.stackoverflow.com)

### 5.1.2 Crowd Selection

Difallah et al. [35] provide the “Pick-A-Crowd” framework, which introduces a “push methodology” in contrast to the self-selection principle (“pull strategies”). They aim at a high quality of answers in crowdsourcing systems and claim to carefully select the worker to perform a given task, based on the social network activities of workers. From tasks descriptions and other task information such as the payment of the task, the system assigns tasks to selected workers from the crowd. They propose three different models, a category-based model, an expert profiling assignment model, and a semantic-based assignment model, which all rely on interactions of workers with the social network *Facebook*<sup>2</sup>.

The recommender system “CrowdRex” provided by Mao et al. [106] calculates task preferences from the worker’s history. In a scenario where developers are searched for a crowdsourced software development task, they apply the results of code challenge competitions to select matching workers.

Basak [13] provides the “BruteForce” framework, which considers three different user profiling methods. Based on these user profiles, KNN approaches are applied in order to provide a recommendation. Within this framework, tasks can be recommended to workers (pull-based), and workers can be recommended to tasks (push-based).

A framework for declarative crowd member selection “December” is proposed by Amsterdamer et al. [5]. They provide a query language to select workers from the crowd based on different attributes assigned to the worker. This Member-QL query language considers a semantic similarity evaluator, which relies on given taxonomies and the respective fact-sets for the workers.

### 5.1.3 Project Staffing and Skill Extraction

In the approach of Barreto et al. [12], the problem of *project staffing* is modeled as a constraint satisfaction problem with multiple constraints. Within their approach “characteristics of the project activities, the available human resources, and constraints established by the software development organization” are taken into account.

Biesalski and Abecker [16] design a system for project staffing within their platform for personal development. They describe an ontology-based system to calculate similarities for the matching of skills and profiles.

Mavridis et al. [109] describe an approach for task assignment, which may also be relevant for project assignment. They use hierarchical skills combined in an ontology which provides skill similarities to optimize the task assignment. In their approach, tasks require a single or a few skills only. However, most projects require a number of tasks and a number of very different skills. Also, the creation of an ontology-based skill-set is a research area on its own.

---

<sup>2</sup> <http://facebook.com>

The current approaches on project staffing require already existing skill information and knowledge about required skills for project execution. The presented approach focuses on skill extraction and therefore provides the information that is necessary to apply the discussed approaches. However, to focus on the skill extraction in detail, only a simple recommendation scheme is applied for the evaluation. Most skill extraction problems are considered as a problem of NER, where specific elements of a text are labeled to represent a particular skill. There exist NER systems that perform well on location, names and other entities. An overview of NER systems for the German language is given by Benikova et al. [15], which includes, e.g. systems training neural networks on large annotated corpora of text. Other approaches use character-based word representations that perform well compared to lexical approaches using gazetteers [93]. A very well known domain for NER is the medical domain, where treatments, illnesses, and chemical terms have to be recognized and identified. Therefore, NER is often combined with NEN, and some approaches join them into one method [97]. Another recent approach provided by Quimbaya et al. [126] combine different dictionaries into a single NER system.

Skill extraction is a very specific sub-task of NER, and there are basically two different methods provided. Implicit skill extraction uses texts and artifacts, where a certain skill is not mentioned explicitly. Explicit skill extraction derives the skills directly from the text and identifies textual representations. Explicit skill extraction systems either generate a skill taxonomy or apply ontology enrichment. The applied method is often dependent on the dataset, which is provided for training and evaluating the skill extraction system. For example, an annotated corpus of text allows to evaluating against explicitly mentioned skills.

A word occurrence based skill extraction system for extracting implicit skills is provided by Rodrigues et al. [132, 133]. They perform their approach of competence mining on a corpus of scientific publications. At first, text processing methods like tokenization, stemming and stop-word removal are applied. Then they extract keywords based on the term frequency within scientific papers and interpret these keywords as competences of the authors. These skills are understood as the competence to help writing an article about the specific topics reflected by the keywords.

Applying the similarities between work artefacts and *Wikipedia* articles, Kivimäki et al. [88] provide implicit skills for employees. When a certain artifact is matched to a *Wikipedia* article, the category structure of *Wikipedia* is traversed to discover skills. Data from LinkedIn is applied to identify *Wikipedia* articles that reflect skills.

Within explicit skill extraction, Harb et al. [61] analyze company websites for an ontology enrichment approach. This approach relies on an existing ontology, which “describes the subject domain accurately” [61]. Explicit association rules are learned from the data, which allows introducing certain skills to the ontology at hand.

A system for explicit skill recognition and normalization, called *SKILL*, is described by Zhao et al. [187] and Javed et al. [77]. They generate a taxonomy of skills by lexically detecting skill terms from a given list of skills and normalizing them.

#### 5.1.4 Recommendation of Job Postings

Tripathi et al. [162] present an overview of different job recommendation procedures. In innovative platforms on the Internet, manifold information about users and objects is available. Platforms such as *Xing* contain demographic data, skills, and contacts of the user. The job offers can be available as unstructured texts, but also contain much more details, such as requirements, keywords or locations [101]. The information available, or the extraction and processing of it, has a decisive influence on the quality of the recommendation. There are many different methods for calculating recommendations based on a wide variety of information.

In the publication of Leksin and Ostapets [98] an element-based collaborative procedure on the professional, social network *Xing* is described. To calculate the object similarities, different measures such as *Cosine similarity*, *Jaccard* and *Pearson* similarity are described. In addition, they present a further approach using matrix factorization. They report that collaborative approaches are not suitable for large amounts of data.

Guo et al. [57] report that most job postings, unlike other traditional objects, are such objects. Another disadvantage is sparseness since most of the entries of a user-item matrix are empty.

Guo et al. [58] present different text-based procedures on the job market *Career-Builder*<sup>3</sup>. First, the feature vectors of the job postings are created using BoW and on the other hand on the basis of entities, such as persons, companies, and locations. Another approach uses the tags of the users with which the documents were marked (e.g., customer service, teaching or finance). The recommendations are calculated based on the distributed KNN [186].

In the work of Huang [76], a content-based recommendation system for job postings on the *Xing* platform is developed. The *Doc2Vec* method of Le and Mikolov [95], discussed in Section 2.4.3, is used to project the documents into the feature space. The document vectors are trained on the basis of 3,000,000 job postings from *Xing*. This is followed by a preselection of job postings for each user using Elasticsearch, which are then ranked with the averaged KNN of Musto et al. [116] discussed in Section 2.5.2.

The content-based method of Poch et al. [125] is used to find suitable jobs on the Job Talent platform. The user first creates a profile, which also contains free text. Based on the profile, potentially suitable positions are presented to the job seeker. For this purpose, properties for users and locations are extracted using a BoW scheme. These are then grouped with *k-means* or Latent Dirichlet allocation and assigned to users with an SVM.

In addition to collaborative and content-based approaches, there are also hybrid methods. For example, Lu et al. [103] present a hybrid recommendation system for a job search website from Switzerland. The aim is to recommend job offers to job seekers on the one hand, but also to suggest potential workers to employers on the other. Job seekers and employers interact to provide different information regarding job postings and other users. There is also further information on the similarity

---

<sup>3</sup> www.careerbuilder.com

of people and job postings on the basis of CVs and job descriptions. A weighted graph is constructed from this information, where the different entities (job seekers, employers, and job) are connected by edges (visited, liked, similar, ...). Then the entities with the 3A ranking algorithm [41] can be ranked for each user.

Zhang and Cheng [186] present an ensemble method from a content-based and a collaborative approach. The content-based procedure creates feature vectors as well as Huang [76] using *Doc2Vec*, but only on the basis of titles and tags. The recommendations are then made using the distributed KNN algorithm. The collaborative approach uses singular value decomposition to determine latent user and document properties, from which the empty cells of a user item matrix can be filled with predictions. To combine the two methods, the 100 most relevant job postings are first determined for a user using the content-based approach. These are then rearranged using the collaborative procedure.

## 5.2 DETERMINING MICRO-TASK SIMILARITY FOR TASK RECOMMENDATION IN MICRO-TASK MARKETS

A categorization of the tasks, as some platforms provide it, helps the workers in their decision which tasks to choose for completion. The classification into given categories allows workers to filter their tasks before selection. Such categories or tags are usually provided by the task requesters. Most of the task recommendation approaches rely on the categorization given on the platform.

To show, that an automatic task categorization is feasible, task descriptions of micro-tasks are used to apply text classification techniques. Evaluating different feature sets and classifiers, this section shows that the classification given on the platform can be reproduced with sufficient accuracy. The classification serves the purpose to show that the very short descriptions of micro-tasks can be used to understand their nature in general. This allows to conclude, that the used classification methods can classify micro-tasks into given logical categories. On a platform that does not follow the strict category assignment as *Microworkers*, such a classification can already be used to support a worker's decision by recommending tasks from the same category that the worker has successfully submitted tasks in.

However, the semantic aspects, that were found to be relevant for task recommendation from the worker's perspective cannot be regarded by the approach of task categorization. Recommenders have the potential to support the workers in their decision and provide tasks according to the interests of the workers. Such systems use methods to calculate the similarity between workers or tasks in order to apply collaborative or content-based filtering. Preceding studies [144, 146] presented in Chapter 4 showed that important similarity aspects for the workers are not only factual aspects like the amount of the reward, but mainly semantic aspects. Among the five most highly rated similarity aspects only semantic aspects were found: the required action, the comprehensibility, the domain, the purpose and the complexity of the task. Therefore, an approach is proposed to calculate similarities with respect to the most valued similarity aspect *required action*, which introduces a method to calculate the similarity between two tasks using a continuous measure [118, 143].

Therefore, features that specifically target the semantic similarity aspect *required action* are defined and discussed. The similarity is calculated by considering the verb phrases found in the description of the micro-tasks and computing an overall similarity value between two tasks taking the similarity values of words from *WordNet*[114] into account. Six differently designed approaches are applied and compared in the evaluation. To evaluate these approaches a gold standard is created manually, describing the similarity regarding the *required action* between ten different tasks which yields 90 similarity values. Additionally, the method to calculate the similarity is used to cluster the tasks, and the resulting clusters are evaluated qualitatively.

With the insights gathered from the classification approach, a method is proposed to cluster micro-tasks accordingly. With this approach content-based or collaborative recommender systems would be able to recommend micro-tasks with respect to *required action* as similarity aspect.

This section is structured as follows. Section 5.2.1 describes the classification approach and its evaluation and highlights the most interesting insights. The central part in Section 5.2.2 describes how the similarity with respect to *required action* is calculated and how similarities between tasks can be derived for recommendation. Section 5.2.2 describes the evaluation of this method, which is done quantitatively, using a manually created gold standard and qualitatively, analyzing generated clusters. Section 5.2.3 concludes the chapter and provides further ideas for research approaches.

### 5.2.1 Classification of Micro-Tasks

A categorization of tasks, as it is provided by some platforms, helps the workers in their decision which tasks to choose. Such categories and tags are usually provided by the employer, which is not necessarily reliable. As shown in Section 5.1, most of the introduced task recommendation approaches rely on a given categorization. Therefore, an automatic classification alone can also be seen as a valuable contribution to the field of task recommendation in crowdsourcing systems. As shown in Section 5.1, in related work either simple BoW schemes are applied, or the focus was different from the domain of micro-tasks.

This section shows, by applying and evaluating different feature sets and classifiers, that it is possible to reproduce the classification given on the platform with sufficient accuracy. The most accurate setup for classification is identified, by extracting four different feature sets and evaluating six different classifiers on every combination of the feature sets. Table 13 shows how unbalanced the tasks are distributed across the different categories. It was shown, that class imbalance leads to a higher tendency for a classifier to predict the larger categories [176]. This is also observed in the provided experiments. However, over- or undersampling of the dataset is not applied in order to prevent overfitting and to use all available training data.

#### 5.2.1.1 Features

The four different feature sets considered for the classification represent different perspectives on the tasks and are shown in Table 12.

*Factual features* include factual information of the tasks, such as *payment* and *time to finish*. While most of the features in this group are numerical, the task characteristic *employer* is a nominal attribute. The *country* attribute is a string attribute, which is transformed into multiple binary features indicating whether a country was explicitly included or excluded.

*Content features* represent the vocabulary used in the textual attributes of a task. Tasks in the same category are likely to exhibit similar vocabulary and should be recognizable by the appearance of specific keywords. Before classification the textual attributes are transformed into word occurrence vectors. The dimensionality and sparsity is reduced with several text processing steps, i.e. case folding and stopword removal. Binary word occurrence, TF-IDF measure, stemming and bi/tri-grams were tested in addition to word counts.

Table 12: Feature sets [143].

FEATURE SET	FEATURES
Factual	Payment, time to rate, time to finish, positions, payment per minute, Employer, countries
Content	<i>N</i> -gram (uni-/bi-/tri-gram) word occurrence; unigram TF-IDF
Structural	Word count, no. of bullet points, avg. words per sentence, Avg. commas per sentence, avg. chars per word, avg. paragraph length, Avg. line length, readability (Gunning Fog Index [56]), lexical diversity [33]
Semantic	URL hosts, named entities, sentiment

*Structural features* capture the writing style and structural information of the task. The structure may be similar in tasks of the same category as a similar sequence of steps is required. The writing style may vary in different categories as they target different groups of workers. The readability is calculated using an *open source* Python module<sup>4</sup>.

*Semantic features* are introduced to extract semantic information like topics and sentiment from the tasks. *SentiWordNet* [46] is used as an external knowledge resource for obtaining a numerical sentiment score. URL hosts, as well as named entities, are extracted as string attributes and transformed to word occurrence vectors before classification. A named entity chunker and a PoS tagger from the Natural Language Tool Kit (NLTK) [17] are used for Named Entity Recognition.

#### 5.2.1.2 Classification

Six different classification algorithms, which are implemented in the *WEKA* [60] framework are used: Random Forest, KNN (*IBk*), Naive Bayes, a rule-based classifier (*JRip*), a decision tree (*J48*), and a Support Vector Machine (*SMO*). For the classification, tasks of the *Other* category are excluded from the dataset, as this category consists of many incoherent tasks, where employers did not find a matching category for their task, which leaves a corpus of 1466 tasks.

#### 5.2.1.3 Dataset for Evaluation

In order to evaluate the different approaches for classification, a manually categorized corpus of micro-tasks is required. On the micro-task market platform *Microworkers*, employers are obliged to assign one of the predefined categories provided by the platform. This provides a ready to use gold standard for the evaluation of the classification methods. Other platforms also enable their employers to categorize and tag their tasks, but only a few of them are comparably strict. Therefore, the dataset was gathered from the *Microworkers* platform, between October and December 2015.

<sup>4</sup> <https://github.com/mmautner/readability>, last accessed on October 28, 2018

The HTML structure is parsed to extract available tasks and their attributes. Extracted attributes are: the ID, description, proof, title, employer, payment, category, time to rate, time to finish, success rate, number of jobs available/done and countries the task is available in. The HTML tags are kept within the textual attributes of the tasks to preserve structural information. The categories for the tasks, provided by the platform, are used as gold standard for the classification step. The distribution of how many tasks belong to which category can be found in Table 13. The table also serves as an overview over the used dataset containing 1602 tasks overall. The categories are of very different sizes, which reflects the distribution on the platform.

Table 13: Category distribution in the dataset.

ID	CATEGORY	TASKS	%
1	Search, Click, and Engage	597	37.27
2	Bookmark a page	182	11.36
3	Sign up	162	10.11
4	Forums	154	9.61
5	Other	136	8.49
6	Google	83	5.18
7	Facebook	69	4.31
8	Mobile Applications	46	2.87
9	Youtube/Vimeo//Dailymotion/Vevo	40	2.50
10	Comment on Other Blogs	32	2.00
11	Promotion	27	1.69
12	Blog/Website Owners	17	1.06
13	Twitter	14	0.87
14	Write an honest review	10	0.62
15	Download, Install	8	0.50
16	YahooAnswers/Answerebag/Quora/Wikians	7	0.44
17	Surveys	6	0.37
18	Leads	5	0.31
19	Testing	4	0.25
20	Instagram	3	0.19
Sum		1602	100.00

To understand what kind of tasks the different categories represent, an analysis of the most varying task attributes between the given categories is provided in the following. Analyzing the tasks from the different categories, it is observed that the averaged task attributes vary per category. This is also observed by Hirth et al. [69]. The category *Testing* has the highest average payment per task (\$7.96) and also the most time-consuming tasks (54.5 minutes). Tasks in the category *Forums* are similarly time-consuming but have lower average payment (\$0.60 per task) resulting in the

Table 14: Weighted average  $F1$  scores for different classifiers and feature sets.

FEATURE SET	RANDOM FOREST	JRIP	SMO
Factual	0.86	0.82	0.73
Structural	0.81	0.74	0.54
Semantic	0.83	0.75	0.84
Content ( $n$ -grams)	<b>0.92</b>	<b>0.92</b>	0.91
Content (TF-IDF)	<b>0.92</b>	<b>0.92</b>	<b>0.94</b>

lowest average payment per minute (\$0.01). The category *Leads* offers the highest average payment per minute (\$0.28) and the category *Twitter* offers the least time-consuming tasks (less than 3 minutes). The average payment in the collected dataset is \$0.31 per task. The average time to finish is about ten minutes. This means an average wage of \$0.03 per minute is observed.

The task description contains URLs, numbers, special characters, and HTML tags. Some conventional text processing steps are applied and the HTML structure is stripped from the text and used as additional attributes. The text processing steps are implemented with regular expressions. These steps include:

1. URLs are replaced with 'dst'. The word occurrence vector will contain the respective information of how many URLs are found within the text.
2. A list of URL hosts is added as a new attribute.
3. HTML tags are removed from the text.
4. The HTML tag sequence is added as a new attribute. This is used to derive structural features later.
5. standard representations are used for common variations of words. For example, 'you'll' is transformed to 'you will'.
6. The following number types are replaced by a textual representation, to reduce dimensionality and sparsity:
  - a) pure numbers: 'num'
  - b) percentages: 'perc'
  - c) numbers which are followed by a currency symbol: 'cur'

#### 5.2.1.4 Evaluation

For each classifier trained on each feature set, a 10-fold stratified cross validation is executed. From the evaluated classifiers, the three best-performing (JRip, SMO and Random Forest) are analyzed in detail. The results of weighted average  $F1$  score are given in Table 14. The content feature set using the TF-IDF approach achieves the best results for classifying task categories over all classifiers. The SMO classifier using the content feature set, obtains the highest  $F1$  score of 0.94. However, the two other classifiers outperform the SMO classifier, when they are trained on the factual or structural feature set. The most stable classifier is found to be the Random Forest classifier with  $F1$  scores of above 0.8 for each of the four feature sets.

Table 15: Weighted average  $F1$  scores for SMO and combinations of feature sets.

FEATURE SET	+FACTUAL	+STRUCTURAL	+SEMANTIC
Factual	-	0.82 (+0.09)	0.86 (+0.13)
Structural	0.82 (+0.28)	-	0.85 (+0.31)
Semantic	0.86 (+0.02)	0.85 (+0.01)	-

In order to analyze their combined performance, the combinations of different feature sets are also evaluated. Combining the content feature set with any other feature set yields no improvement. For the remaining feature sets, there are improvements when adding features from another feature set (see Table 15). For example, the  $F1$  score improves by 0.28 when factual features are added to structural features. Combining all three feature sets results in small to no further improvements. The results for all different feature set combinations and classifiers can be found in Table 29.

The  $F1$  scores per class are also included in the evaluation. As expected, there is a clear descent of the  $F1$  score for less supported categories, shown in Figure 22. However, the  $F1$  score is above 0.7 for all categories that contain at least 10 tasks. Training a classifier on a class with less than 10 positive examples expectedly performs not as good. For categories with at least 83 positive examples in the corpus, an  $F1$  score greater than 0.94 is achieved.

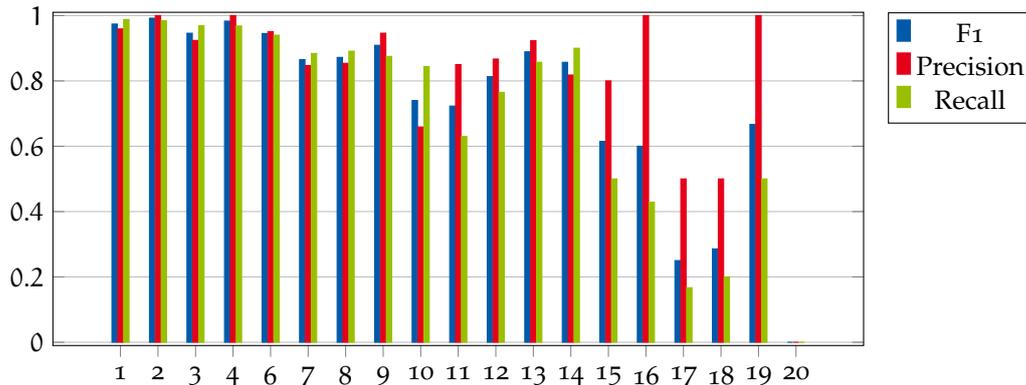


Figure 22:  $F1$  score per category for the content feature set and SMO classifier. Categories are sorted decreasing in size. Category *Other* (ID=5) is not considered here. Categories with ID 15 and greater have less than 10 positive examples.

This evaluation showed, that it is generally possible to reproduce given task categories from the task descriptions and that a classification of micro-tasks is feasible. The SMO classifier provided the best result, while content features represent the best-performing feature set. As long as classes are represented by at least 10 positive examples, an  $F1$  score of at least 0.7 can be reached using a corpus of 1466 tasks.

### 5.2.2 *Micro-Task Similarities Regarding the Required Action*

In order to rank tasks, regarding their semantic similarity to a given task, continuous measures are required instead of classifiers for binary class membership. The results of the classification in Section 5.2.1 show that it is possible to judge from the task description about information of the task such as the category assigned by the employer. The insights about how certain features can be applied for the classification task can help to propose an approach for calculating task similarities based on such aspects.

As discussed in Section 5.1.1, most recommender systems for tasks in micro-task markets rely on keywords and tags provided by the task requester. Determining task similarity with respect to the similarity aspect of *required action* provides a worker-centered approach which takes the whole task description into account. This allows the introduction of more fine-grained features compared to simple BoW approaches.

In this section, certain features are discussed and proposed, that are specific for the aspect of *required action*. From those features, a method for determining the similarity is designed that focuses on the aspects of the action of the task. In Section 5.2.2 the implementation of this method is evaluated quantitatively and qualitatively.

#### 5.2.2.1 *URL Hosts for Required Action*

The categories defined on the platform (Table 13) involve different web platforms such as Google or Facebook where such tasks have to be performed on. The action that has to be performed for a task is often directly depending on the involved platform. In Section 5.2.1.1 it is described how the URL hosts are extracted from the task descriptions for the semantic dataset. The URL hosts are expected to represent the platforms that are involved in the action to solve the task. Therefore, the URL hosts that can be found within the task description are considered as valuable information to measure how similar two tasks are in their required action.

#### 5.2.2.2 *Verb Phrase Similarity for Required Action*

The best-performing feature sets for the classification in Section 5.2.1.1 are the content feature sets. In this evaluation, The TF-IDF approach performs only slightly better than *n-gram* word occurrence features. Using this insight, a specific implementation is designed that concentrates on *n-grams* representing the verb phrases found within the task descriptions. In general, verbs are supposed to represent actions within any sentence. In a task description, they are expected to reflect the actions that are required to solve the task. As actions are also defined by the object on which they are performed on, complete verb phrases are considered for the calculation of the similarity. Word similarities provided by *WordNet* are applied to compute the similarity between verb phrases, which is later described in detail. At first, the extraction of verb phrases from the task descriptions is described.

To recognize verb phrases, the task description is tokenized, and the tokens are annotated with their PoS tags using NLTK libraries. To recognize verb phrases in the

specific dataset, containing many short and bullet point formatted sentences, two simple regular expressions are applied:

$$\begin{aligned} &< \text{VBP|VB} >< \text{TO|IN} >? < \text{DT} >? < \text{VBN|JJ} >? < \text{NN|NNS} > + \\ &< . >< \text{NNP} >< \text{TO|IN} >? < \text{DT} >? < \text{VBN|JJ} >? < \text{NN|NNS} > + \end{aligned}$$

In the Penn Treebank PoS tagset [137] used by the NLTK tagger, VB, VBP and VBN are forms of verbs, NN, NNS and NNP are forms of nouns, TO is the word ‘to’, IN a preposition or conjunction, DT a determiner and JJ an adjective. The regular expression is designed to match verb phrases like ‘enter the given keyword’, ‘scroll to the bottom’ and ‘click the blue link’. The second expression was added as it was observed that the NLTK tagger often tags verbs at the beginning of sentences as proper nouns. Instead of implementing an adapted PoS tagger, the presented solution was found to recognize more verb phrases correctly and still produce few false positives. This grammar is used to evaluate different word similarity measures from *WordNet*. Improving on the described grammar, more complex regular expressions were added, considering verbs connected with conjunctions, superlative, and comparative forms of adjectives and proper nouns. This complex grammar recognizes more verb phrases in general, but also produces more false positives. This extension is called *complex grammar* and compared in the evaluation with the *simple grammar* described before.

Two task descriptions, which feature same verb phrases, are assumed to be similar regarding their required action. However, many verb phrases bear similar meaning, even though the used vocabulary is not exactly the same. Instead of only using a verb phrase occurrence, word similarities from *WordNet* are applied as suggested in discussed related work [1, 29]. To be able to compute the mutual similarity between all tasks in the dataset, tasks are represented as n-dimensional attribute vectors, where n is the number of all verb phrases that have been detected in the dataset (corpus verb phrases  $vc$ ). The single attributes in a vector representing a task indicate the maximal similarity between the respective corpus verb phrase  $vc$  and all verb phrases within the task’s description (instance verb phrases  $vi$ ). This means that a vector representation of a task contains attributes of value 1 for verb phrases contained in its description and of values between 0 and 1 for all verb phrases contained in the dataset but not in the task itself. The final mutual task similarities are computed using *Cosine similarity* between the vectors. This is described in detail with examples in the following paragraphs.

To determine the similarity between two verb phrases, the following procedure is applied, taking the word similarities for the nouns and verbs from *WordNet*. For each noun in a corpus verb phrase  $vc$ , the maximal similarity ( $sim_{WN}$ ) to any noun in  $vi$  is determined, and their sum is computed to find the similarity value  $sim_{nouns}$  (15) between two verb phrases. The same is done with verbs for obtaining the verb similarity value  $sim_{verbs}$  (16). The similarity between the two verb phrases  $vi$  and  $vc$  is determined by averaging the noun and verb similarities, which provides the verb phrase similarity  $sim_{VP}$  (17):

$$N_{vc/vi} = \{\text{nouns in } vc/vi\} \quad (13)$$

$$V_{vc/vi} = \{\text{verbs in } vc/vi\} \quad (14)$$

$$\text{sim}_{\text{nouns}} = \sum_{n \in N_{vc}} \max_{n' \in N_{vi}} \text{sim}_{WN}(n, n') \quad (15)$$

$$\text{sim}_{\text{verbs}} = \sum_{v \in V_{vc}} \max_{v' \in V_{vi}} \text{sim}_{WN}(v, v') \quad (16)$$

$$\text{sim}_{VP}(vc, vi) = \frac{\text{sim}_{\text{nouns}} + \text{sim}_{\text{verbs}}}{|N_{vc}| + |V_{vc}|} \quad (17)$$

For each verb phrase pair  $(vc_i, vi_j)$ , this procedure is executed. The verb phrase  $vc_i$  is from the set of corpus verb phrases, while the verb phrase  $vi_j$  is an instance verb phrases which was extracted from considered task  $t$ . For each verb phrase in the corpus, the task  $t$  receives a similarity value. This value is set to the maximal similarity found between the corpus verb phrase  $vc_i$  and any instance verb phrase  $vi_j$  in task  $t$ :

$$t[vc_i] = \max_{vi_j} \text{sim}_{VP}(vc_i, vi_j) \quad (18)$$

*WordNet* provides three measures of similarity between words operating on the path lengths of a hypernym/hyponym relation hierarchy (*lch*, *wup*, *path*) [120]. The different similarity measures *lch*, *wup*, and *path* are compared in the evaluation. To compute the distances between two tasks using this method to determine the similarity with regards to the *required action*, the *Cosine similarity* distance is applied.

### 5.2.2.3 Dataset and Similarity Gold Standard

To evaluate the proposed method, a small manually labeled gold standard was created. This gold standard defines task similarities regarding the aspect of *required action*. For a set of ten tasks  $T_1, \dots, T_{10}$ , each of the tasks is assigned to one human annotator as the so-called *base task*. The annotator then decided for each of the other tasks in the set (*non-base tasks*) how similar it is to the base task. As a result, the gold standard contains one ranking  $R_1, \dots, R_{10}$  per annotator. In each of the rankings  $R_i$ , the task  $T_i$  represents the base task. All other tasks are supposed to be ranked in descending similarity compared to the base task. An exemplary ranking can be found in Figure 23, where the base task is put in the first column, and the remaining tasks are supposed to be placed to the right, descending in similarity. Twice as many columns than there are tasks to rank (20 including the column for the base task) are offered, to give the annotators the possibility to quantify similarities between tasks by leaving gaps. This allowed the annotators to group tasks together, where a definite decision on which task is more similar was hard to find. The 20 columns also

allowed the annotators to fill the columns without grouping tasks together, leaving a gap between each entered task. A numerical value is computed, which represents the distance  $d_g(T_i, T_j)$  between the base task  $T_i$  and any other non-base task  $T_j$  in the gold standard. To actually force annotators to define those gaps in the rankings, it is

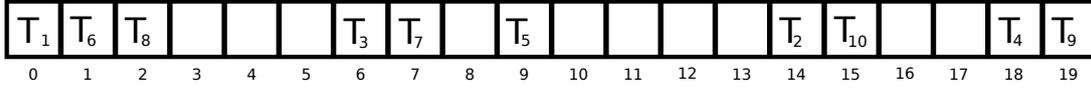


Figure 23: Example gold standard rating scheme [118].

demanding that the base task is filled in the first column, and that the last column has to be filled with a (least similar) task, too. Therefore, for a ranking  $R_i$ , the following equation holds:

$$d_g(T_i, T_i) = 0 \text{ and } \max_{j \in [1, 10]} d_g(T_i, T_j) = 19 \quad (19)$$

#### 5.2.2.4 Quantitative Evaluation

The proposed method for determining task similarity with respect to the aspect of *required action* was used to produce rankings for every base task that can be compared to the gold standard rankings. The distance values obtained by the similarity method  $d_c(T_i, T_j)$  range from 0 to 1. These values are scaled to the numerical range given by the gold standard distance values:

$$d'_c(T_i, T_j) = 19 \cdot d_c(T_i, T_j) \quad (20)$$

The comparison of calculated and gold standard rankings was conducted using the Spearman rank correlation and mean absolute error. The three different measures from *WordNet* (*lch*, *wup*, *path*) are evaluated as discussed earlier. For the best-performing *WordNet* similarity method ( $WN_{path}$ ), different feature setups are evaluated, by leaving out the URL host feature and using the more complex grammar as described in Section 5.2.1.3. The results can be found in Table 16 and Table 17 respectively.

Regarding the Spearman rank correlation (Table 16), the configuration using the *WordNet path* similarity measure outperforms configurations using *wup* and *lch* measure with a median correlation of 0.46. Removing URL hosts from the feature set does not change the median correlation but results in a higher maximal correlation of 0.7. This maximal correlation is only exceeded by using the complex grammar. However, for this configuration, the minimal correlation is very low (0.09), and the median is worse. Therefore, the configuration using *WordNet's path* similarity measure and the simple grammar for verb phrase recognition including URL host features are considered to perform best in terms of the correlation.

In terms of mean absolute error (Table 17), configurations with simple grammar and URL hosts using *WordNet's wup* and *lch* measure result in lower errors than the ( $WN_{path}$ ) measure (median of 5.89 and 5.29 in contrast to 6.53, respectively).

Table 16: Spearman rank correlations ( $r_s$ ) for *required action*.

MEASURE	$r_s$		
	MEDIAN	MAX	MIN
$WN_{path}$	0.46	0.70	0.25
$WN_{wup}$	0.30	0.62	0.09
$WN_{Ich}$	0.29	0.50	-0.07
$WN_{path}$ , no URL hosts	0.46	0.59	0.27
$WN_{path}$ , complex grammar	0.43	0.83	0.09

Table 17: Mean absolute error (mae) for *required action*.

MEASURE	mae		
	MEDIAN	MAX	MIN
$WN_{path}$	6.53	8.13	5.59
$WN_{wup}$	5.89	7.82	4.70
$WN_{Ich}$	5.29	7.40	4.68
$WN_{path}$ , no URL hosts	6.17	8.69	5.33
$WN_{path}$ , complex grammar	5.82	7.28	3.92

Taking the maximum and minimum values for the mean absolute error into account, the ( $WN_{path}$ ) measure, using the complex grammar, provides the lowest values for both.

For the evaluation, ranking similar tasks is considered more important than determining the exact distance. Hence, correlation is the more important measure here. Overall, the *path* similarity measure of *WordNet* together with the simple grammar and the URL host features are found to be the best-performing configuration. This configuration produces the most correlations above 0.5 and is comparatively stable for the rankings in Table 16. The complete evaluation results can be found in Table 30 in the Appendix.

#### 5.2.2.5 Qualitative Evaluation

The qualitative evaluation provides a clustering of the given tasks. In a recommendation scenario, a clustering is not required, but it is given here to provide further insights on how the provided method performs on the given corpus. For this qualitative analysis, the 1602 tasks in the collected dataset are clustered, applying the *DBSCAN* clustering algorithm [45, 121]. The parameters for *DBSCAN* are optimized by comparing the resulting clusters to the gold standard. *DBSCAN* requires the parameter *eps*, which defines the radius of the considered *neighborhood* of a point *p*. A particular point *p* is considered a *core point* if at least *minPts* points are within the neighborhood of this particular point *p*. Several parameter combinations of *eps* and

Table 18: Word frequencies in selected *required action* clusters. The word frequency is given in brackets. Most interpretable words are highlighted in bold face.

RANK	A <sub>1</sub>	A <sub>6</sub>	A <sub>11</sub>
	735 tasks 54429 words	36 tasks 1367 words	19 tasks 691 words
1	dst (2045)	<b>email</b> (81)	<b>app</b> (58)
2	profile (1959)	go (63)	<b>review</b> (46)
3	<b>google</b> (1488)	select (60)	write (28)
4	task (1036)	<b>sign</b> (54)	honest (22)
5	url (851)	<b>address</b> (50)	<b>install</b> (19)
6	<b>keyword</b> (839)	<b>name</b> (40)	<b>feedback</b> (18)
7	least (818)	already (37)	dst (18)
8	make (739)	dst (36)	note (17)
9	go (729)	paid (35)	<b>download</b> (17)
10	<b>click</b> (674)	optional (29)	least (17)

minPts were evaluated using the gold standard, to determine the best-performing clustering setup. The parameter selection resulted in  $\text{eps} = 2$  and  $\text{minPts} = 10$  as the best-performing setup. With ten tasks to be ranked and twenty positions open, ten columns remain empty. Especially as the annotators were forced to put the base task in the leftmost column and the task which is supposed to be least similar in the rightmost column, these empty spaces are found between the inserted tasks. Tasks with more empty columns between each other are considered less similar towards each other. Therefore, the ranking was considered to provide the following constraints for clustering. First, the base task and directly neighboring tasks are supposed to be in the same cluster. Second, tasks which are not directly neighboring the base task should not be in the same cluster as the base task. The further away a task is placed in relation to the base task, with regards to empty cells, the more important is that they do not appear in the same cluster.

The setup, using the selected parameters  $\text{eps} = 2$  and  $\text{minPts} = 10$ , produced 14 clusters with 270 outliers for the method with respect to *required action*. The cluster sizes vary between 735 and 12 tasks. To get an idea of what kind of tasks are contained in the clusters, the word frequencies within the clusters are analyzed. Table 18 shows the ten most frequent words of three exemplary clusters resulting from the method. The cluster A<sub>1</sub> is interpreted to mainly consist of tasks requiring the user to search and click. Cluster A<sub>6</sub> contains tasks requiring sign-in actions and cluster A<sub>11</sub> seems to consist of tasks requiring writing app reviews. Not all produced clusters are that easily interpretable using word frequencies; however, all clusters are clearly distinguishable. The word frequencies for all clusters can be found in Table 31 and Table 32 in the Appendix.

Table 19: Category distribution for selected clusters regarding *required action*.

CATEGORY	A <sub>1</sub>	A <sub>6</sub>	A <sub>11</sub>
Blog/Website Owners	-	-	0.11
Facebook	0.08	-	-
Google	0.10	-	-
Mobile Applications	0.02	-	0.89
Other	0.16	-	-
Promotion	-	0.03	-
Search, Click, Engage	0.47	-	-
Sign up	0.09	0.94	-
Youtube/Vimeo/...	0.04	-	-
Various	0.04	0.03	-

The goal of the clustering is not to reproduce the categorization of *Microworkers*. However, it is assumed that some categories also represent actions (e.g., Sign Up, Search, Click and Engage). Therefore, the category distribution in the obtained clusters is analyzed regarding *required action*. Figure 19 shows the results for the three selected clusters. The cluster  $A_6$  and  $A_{11}$  mainly consist of tasks that belong to one category (*Sign up* and *Mobile applications* respectively). Finding compliance with single categories from *Microworkers*, which reflect single actions well (as it is the case for *Sign up*), can be an indicator that the proposed method is able to detect similarities regarding the required action in tasks. On the other hand, cluster  $A_1$  consists of tasks from many different categories, which shows that similarities are identified, that go beyond the given categories. In the case of  $A_1$ , the cluster appears to contain search tasks in general.

#### 5.2.2.6 Discussion of Recommendation Scenarios

The presented method to determine task similarity with respect to the aspect *required action* is designed to enable recommendations in micro-task markets based on the semantic similarities of tasks. A recommendation in a micro-task market in such a scenario can be based on the content of tasks that a worker submitted before. The tasks successfully done by the worker, combined with the provided method, allows finding tasks that are similar in action, which can then be recommended to the worker. In a collaborative filtering scenario, a task that was identified to be suitable for the worker, but is not available on the platform due to the high churn in micro-task markets, can be replaced by a still available task which is similar in regards to the *required action*. When it comes to recommending tasks in micro-task markets, a semantic similarity is not the only factor to consider. Factual aspects such as payment and required completion time are very important to the worker, as discussed in Section 4.2. Contextual aspects, such as the currently used device, available time or the number of recently done tasks should be considered too, as well as ratings by

worker and employer towards certain submissions. However, the method provided to calculate the semantic similarity with regard to the *required action* is considered important, as it is chosen as the most valued similarity aspect by the workers [144].

Further semantic aspects, such as those also valued as important factors for recommendation, can provide the base for further methods to calculate the similarity of tasks. Being able to measure the similarity of two micro-tasks, with regard to the given semantic aspects, enables the creation of recommender systems which takes sophisticated preferences of each worker into account. A recommender system that requires reliable automatic classification of micro-tasks as well as the capability of finding semantically related micro-tasks in an unsupervised manner provides value to the workers and might also improve the quality of work for the task requesters.

### 5.2.3 Conclusion

This section provides an approach on how to measure semantic task similarity based on the proposed similarity aspect *required action*. At first, it is shown that the content of task descriptions can be used to classify micro-tasks into known categories. The evaluation of the classification shows that a classification is feasible using the proposed setup. From this basis, a method to determine the similarity of tasks with respect to the similarity aspect *required action* for micro-tasks is proposed, which was found to be the most important similarity aspect to workers on the platform *Microworkers*. Different approaches for the calculation of the similarity are quantitatively evaluated against a manually gathered gold standard and analyze the best setup qualitatively by providing insights into the generated clusters within the given corpus. The quantitative analysis showed that the results of the method to calculate similarity can correlate with the result of human annotators. The qualitative analysis shows that clusters generated from the similarity measure can be compared where the category from the platform reflects a specific action. It is also shown that the similarity aspect *required action* can provide similarities beyond the given categories. Therefore, the calculation method defined for the *required action* can be effectively used to find tasks that require a similar action.

However, the provided similarity aspect reflects only a small portion of aspects that have to be considered when recommending micro-tasks which are subject to future work. Also, the quantitative evaluation is conducted on a small manually gathered gold standard to determine the similarities of tasks. This evaluation can be extended by providing the results to the workers on a micro-task market to gather more meaningful results. For further development of the proposed method, further features and methods can be considered such as provided by the fields of topic modeling and word embeddings.

### 5.3 CROWD SELECTION FOR RECOMMENDATION IN LOCATION-BASED CROWD-SOURCING

As described in Section 2.2.3, location-based micro-task markets require the workers to be present at a physical location to perform a task, and crowd selection strategies are needed to enable a “push” to relevant workers when a new matching task comes up in their area. For example, a task may request a picture of a speed sign at a specific location to validate the current speed limit on this road [129].

As in micro-task markets, the selection of workers for tasks can be supported by recommendation systems. To answer the question, whether textual task similarities can improve the recommendation in location-based crowdsourcing platforms (RG<sub>3</sub>), this section presents an approach for crowd selection based on textual similarities of tasks from the history of the workers [128]. In this approach, task similarities are used to provide a clustering of tasks, enabling a selection of workers who worked on tasks from the same cluster. The presented approach is evaluated against a matrix factorization approach, which is based on a given categorization provided by the platform (task-types). This matrix factorization approach is therefore applied as a baseline and evaluated on the given dataset. The results show that the presented approach can significantly improve the results of recommendation in location-based crowdsourcing platforms.

#### 5.3.1 Methodology for Crowd Selection based on Textual Similarities

The method follows several steps to filter workers to whom a task should be recommended to, as shown in Figure 24. The first criterion for recommending tasks in a location-based crowdsourcing platform is the availability of a worker at the location of the task. Therefore, the similarity-based approaches and the baseline take the location of tasks and workers into account. All tasks completed in the platform are analyzed for semantic similarities and clustered accordingly. When a (new) task is introduced, for which workers have to be identified to recommend the task to, the following filter steps are performed. At first, the location of the task is examined, and only such workers are considered for recommendation, which have completed tasks in areas near that location. In a second step, the task is sorted into one of the previously computed clusters, based on the similarities in the task description. Now, only such workers are considered, which have completed tasks from within this cluster. Therefore, this step provides a crowd selection based on the similarity of task descriptions. In a last step, the workers are ranked by a matrix factorization approach to provide recommendations.

The baseline approach basically implements the same steps, leaving out the filtering for textual similarities. The details of the crowd selection approach are presented below. The following description of the method is visualized in Figure 25.

For a previously unknown task and its corresponding description document  $\tilde{d}_u$  the users  $\tilde{u} \in \tilde{U}$  can be recommended. Whether a user has completed a document  $\tilde{d} \in \tilde{D}$  successfully, is defined by the ratings of a user  $\tilde{L} \subseteq \tilde{D} \times \mathbb{R}$ . The presented

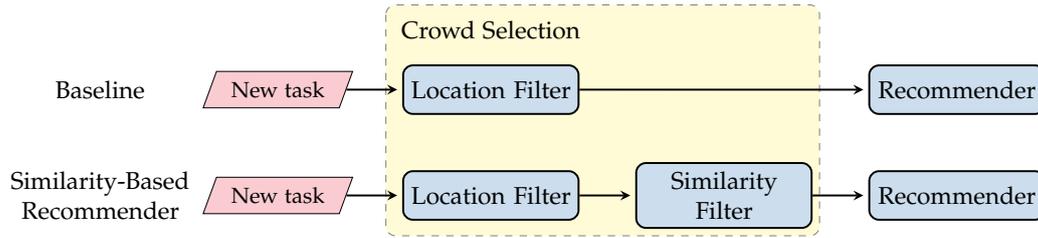


Figure 24: Overview on the applied process for crowd selection.

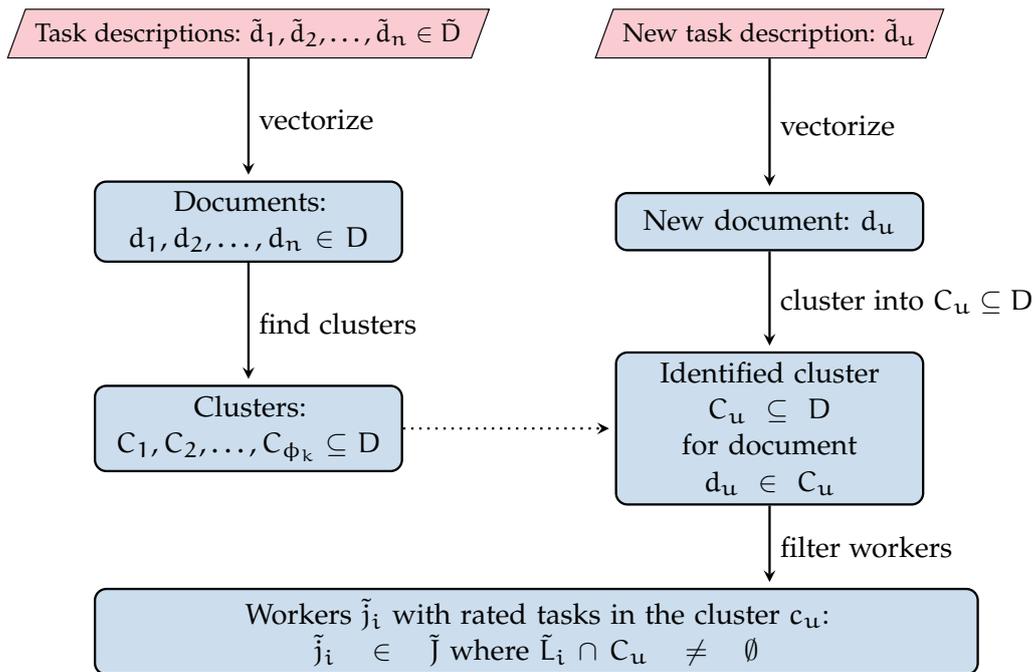


Figure 25: Overview on the applied method for the similarity-based crowd selection.

method applies an additional filter to select a crowd according to the task similarities. To achieve this, the documents  $\tilde{d} \in \tilde{D}$  are projected into a vector space representation  $d \in D \subseteq \mathbb{R}^{\phi_f}$  with  $\phi_f$  feature dimensions, based on their similarities. Afterward, the documents  $d \in D$  are grouped into  $\phi_k$  clusters  $C_1, C_2, \dots, C_{\phi_k}$  using an unsupervised clustering mechanism. The previously unknown document  $\tilde{d}_u$  can now be projected into the same vector space and clustered accordingly into an already existing cluster  $C_u$ . Now, such users  $\tilde{j} \in \tilde{J}$  can be identified, which have completed a task from this cluster  $C_u$  ( $\tilde{j}_i \in \tilde{J}$  where  $\tilde{L}_i \cap C_u \neq \emptyset$ ). Only such workers  $\tilde{j}$  in the crowd  $\tilde{J}$  are considered for further recommendation.

### 5.3.2 Evaluation of Crowd Selection and Task Recommendation

To apply this method, several decisions about algorithms and mechanisms applied have to be made. At first, a meaningful location-based filter has to be defined. Secondly, the representation based on document similarities has to be chosen. Thirdly, a suitable clustering method has to be identified. Additionally, the recommender algorithm has to be chosen, which only considers the selected crowd for recommendation. To decide these aspects, different analysis are performed based on a given dataset.

#### 5.3.2.1 Description of the Dataset

The dataset for the evaluation of the recommendation consists of one year of data on a location-based crowdsourcing platform, spanning from July 2016 to July 2017. During this period, around 90,000 micro-tasks entries are found. The data for analyzing the activity of workers is collected from 2012 to 2017. During this period around 2700 different task types can be identified and 350,000 workers are registered on the platform. Task types combine tasks that are of the same category but have to be performed at different times and different locations. Within the group of workers who performed 20-50 tasks, the median of tasks from different task types is 9. This shows that workers tend to repeat tasks from the same task type. The tasks within the dataset provide a description of the task as well as the location where the task has to be performed at. The data about the workers provide which tasks they have completed, including whether the results were accepted, rejected or disputed. There is no location given for a worker but the locations that can be inferred from the locations of completed tasks. The textual descriptions are extracted from the tasks and tokenized for further processing.

#### 5.3.2.2 Location-Based Filter

The dataset does not provide location data of workers, neither current location nor location of home or work address. Therefore, a location-based filter has to be created from the locations given by the tasks the workers submitted. From the data, the distance between two tasks performed by a worker can be computed, and the average or maximum distance between two distinct tasks can be determined. This can be

interpreted as the distance workers are willing to travel. For this evaluation, the mean distance between two performed tasks of all workers is considered. That means a worker is selected for a task when the worker previously completed a task at a location within the range of this mean.

### 5.3.2.3 Document Similarity Measures and Clustering

It is necessary to find an appropriate method to project the documents into a vector space representation. Based on earlier discussions, the methods of *Doc2Vec*, *FastText* and TF-IDF are evaluated comparatively. Also, a suitable unsupervised clustering algorithm has to be identified. Here, *DBSCAN* and *HDBSCAN* are considered as well as *k-means* as discussed in Section 2.5.1. The *k-means* approach is evaluated with a number of clusters  $\phi_k$  between 10 and 200.

Though *DBSCAN*, *HDBSCAN*, and *k-means* performed comparably good, the best-performing setups for this dataset used *k-means*. They are presented in Table 20. The performances of these setups do not deviate significantly. However, the best-performing setup (using the *FastText* approach with *k-means* and  $\phi_k = 110$ ) is considered for further evaluation.

Table 20: Best-performing setups of document similarity and clustering methods

VECTOR REPRESENTATION	NUMBER OF CLUSTERS	F1	R-PRECISION
<i>FastText</i>	110	<b>0.056</b>	<b>9.57%</b>
<i>FastText</i>	170	0.053	<b>9.57%</b>
<i>Doc2Vec</i>	60	0.05	8.96%
<i>Doc2Vec</i>	170	0.054	8.54%
TF-IDF	10	0.055	9.06%
TF-IDF	20	0.0555	9.31%

### 5.3.2.4 Evaluation of Crowd Selection for Recommendation

The basis for the calculation of the recommendation are the user’s ratings on the tasks (compare Section 2.3). Yuen et al. [185] apply a scheme, where the interactions of workers and the feedback of requesters are used to generate a five-star rating system, which many of the modern recommendation systems rely on. Such ratings usually apply the lowest rating to a non-attempted task, higher ratings to attempted but rejected tasks, and the highest ratings to successfully attempted tasks. Following this approach, the user ratings given in Table 21 are generated from the dataset. In the given dataset, there are five states: Verified means a task was completed, the results were accepted, and the worker received the remuneration. Verified-dispute means that correct results were provided only after the first results had been denied. The denied-dispute state describes a situation, where results were denied even after several iterations with the worker. The denied state is used, where the results were re-

Table 21: User ratings derived from task attempt state of workers.

TASK RESPONSE	APPLIED RATING
verified	5
verified-dispute	4
denied	3
denied dispute	2
no attempt	1

jected, and the worker did not attempt the task a second time. Based on these ratings the collaborative filtering method of matrix factorization is applied using singular value decomposition [90]. This returns a ranking of workers for recommendation.

For the evaluation, a recommendation is considered correct, if a recommended worker has a rating of 4 or 5 for the given task. For evaluation of  $Precision@k$ , the first  $k$  workers from the computed ranking are considered. For some task types, the number of positive examples in the test set  $m$  may be less than  $k$ . Therefore, the  $R$ -Precision is used additionally, which considers a different  $k = m$  for each evaluated task. The evaluation on the time series data was cross-validated using a 10-fold time series split.

The evaluation results for  $Precision@5$  are given in Figure 26(a), and the evaluation results for  $R$ -Precision are given in Figure 26(b). Overall, the similarity approach improves the results compared to the baseline in terms of  $Precision$ . The performance increase of the similarity approach from  $Precision@5$  to  $R$ -Precision indicates that this approach performs especially good for smaller  $k$ . That means it is more likely to recall the correct workers in evaluation scenarios, where only a few positive examples are given. Accordingly, the similarity approach performs better for the  $Recall@5$  measure, which can be seen in Figure 26(c). As can be seen, the approach integrating the similarity performs significantly better than the baseline approach. Table 22 presents significant differences between baseline and similarity approach. In this context, significance is defined as rejecting the null hypothesis that there is no difference in recommendation performance when the concept of *task similarity* is introduced, with at least  $p < 0.005$ . The low  $p$ -values indicate that the use of *task similarity* for crowd selection can improve the results of task recommendation in location-based micro-task markets.

Table 22: The  $p$ -values: for baseline and task similarity approaches.

EVALUATION METRIC	P VALUES
$Precision@5$	$p < 0.0007$
$Recall@5$	$p < 0.0007$
$R$ -Precision	$p < 0.0002$

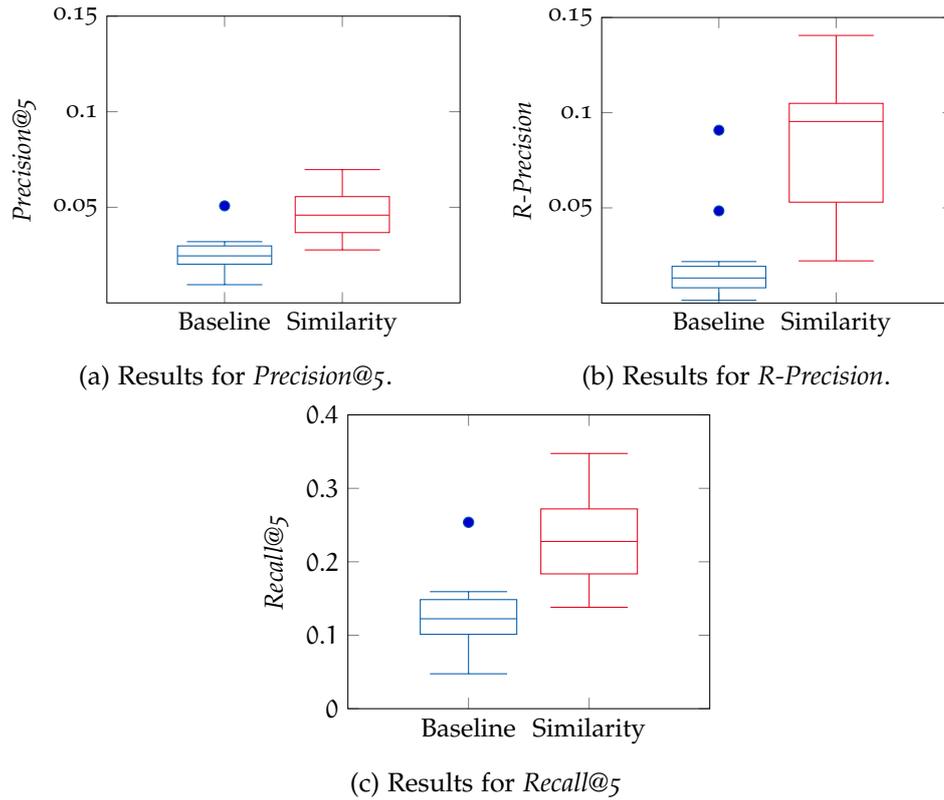


Figure 26: Evaluation results for  $Precision@5$ ,  $R-Precision$ , and  $Recall@5$ .

### 5.3.3 Conclusion

This section proposed a new method for crowd selection based on task similarities for the recommendation of tasks to workers in location-based crowdsourcing. The presented crowd selection approach takes the similarities of task descriptions into account to filter workers for recommendation. The baseline approach applied a collaborative filtering scheme based on the given task types on the platform. Introducing task similarities to determine suitable workers for recommendation, improved the performance compared to the baseline approach significantly. This shows that also in location-based micro-task markets, a recommendation can be supported by considering similarities found within the textual task descriptions.

#### 5.4 SKILL EXTRACTION FOR PROJECT ASSIGNMENT RECOMMENDATION

This section presents an approach for recommendation of employees, which can be assigned to a project based on skills [155]. Enterprises often assign their employees towards specific projects depending on the project requirements and the qualifications of the employees. Well executed project assignments can optimize product quality as well as motivation and productivity of employees [12, 83]. Within enterprises, information about employee qualifications, capabilities or skills, and information about project requirements are often provided in unstructured textual descriptions. Therefore, it is necessary to extract the given information on project requirements and employee skills into a robust representation. This section describes a system for recommending project assignments in a real-world scenario based on approaches for skill term extraction from unstructured textual descriptions.

The scenario is motivated by an industry partner, which provides the dataset for skill extraction and project assignments. Within the dataset, there are 270 projects and 227 employee profiles provided. This dataset offers several challenges: It includes the textual descriptions of *project proposals* and *employee profiles*. The *project proposals* include project descriptions which state *required skills*. The *employee profiles* include *project descriptions* which state some of the *possessed skills*, as well as an explicit list of *possessed skills*. The list of *possessed skills* in an *employee profile*, does not necessarily include all skills stated in the *project descriptions* of that *employee profile*, and vice versa. Also, the project assignment is decided on additional factors such as availability and other criteria, apparent to the decision-makers, but not given in the dataset of textual descriptions.

The actual assignment of employees to projects is known, which is based on the decision of specialized employees, who assess such project assignments in their day-to-day work. So for each project, there is a certain number of employees assigned. This information is used to evaluate the project assignment based on skill extraction later on. As there is mostly only a single employee assigned to a project, *Precision* and *Recall* do not provide appropriate evaluation measures. Therefore, a custom evaluation measure *matches at k* will be defined and used for the evaluation.

In Section 5.1 the methods from related work applied towards the methodology have been described. The proposed method is presented in Section 5.4.1 The setup of the evaluation, the dataset and the results are explained in Section 5.4.2, followed by a conclusion in Section 5.4.3.

##### 5.4.1 Methodology of Skill Extraction and Project Assignments

This section provides the methodology for skill extraction as well as for the recommendation of project assignments based on skill matching. An overview of the whole system is given in Figure 27. As input, the system retrieves *projects* and *employee profiles*. Projects are provided with *project assignments* and the textual description of the *project proposal*. *Employee profiles* provide a textual *skill list* as well as textual *project descriptions*. From the joined *skill lists* of all *employee profiles*, a *skill term list* is generated,

which includes every *skill term* found within these lists. The *skill term extraction* is provided with the *skill term list* for training. The *skill term list generation* also generates the *profile skill terms* which serve as an evaluation standard for the *extracted profile skill terms* generated by the *skill term extraction*. The main component is the *skill term extraction*, while the *skill matching* component is kept simple. The *skill term extraction* takes *project proposals* and *project descriptions* as input to extract skill terms. The *extracted project skills* and the *extracted profile skills* are then matched for the recommendation of *project assignments*, which are evaluated against the given project assignments in the projects. The following sections describe the *skill term list generation*, *skill term extraction* and *project assignments* step by step.

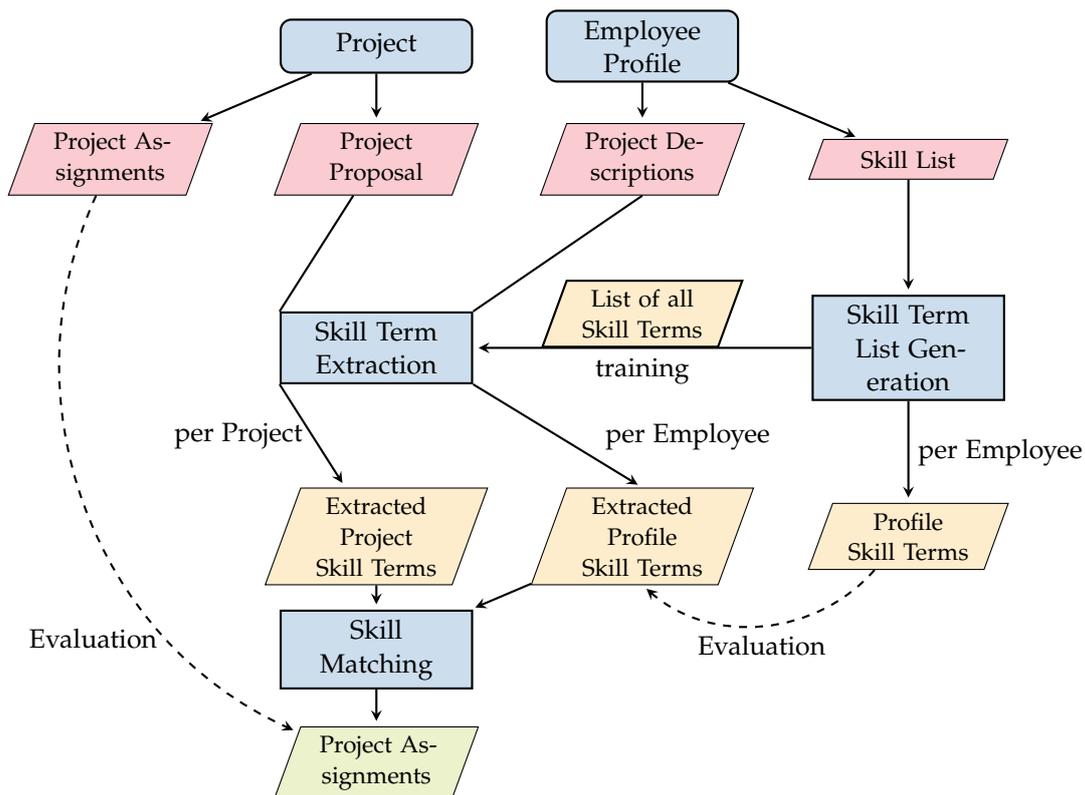


Figure 27: Overview of the process of skill extraction for project assignments

#### 5.4.1.1 Skill Term List Generation

The *skill term list generation* takes the *skill lists* from the employee profiles and generates an overall *skill term list*, as well as *profile skill terms* per employee. To achieve this, some text processing is necessary to generate a consistent list of skills from the texts. Therefore, all terms are converted to lowercase and special characters (-, \*, <sup>TM</sup>, ...) are removed to reduce the variation of terms. Different kinds of conjunctions are detected and converted. For example 'unix shell scripts (bash, csh); ms project; ms-visio; html/xml' is converted into 'unix shell scripts, bash,

csh, ms project, ms-visio, html, xml'. The generated *skill term list* serves as example data for the *skill term extraction* to learn from.

Most of the project proposals are in German, and only some are in English. However, both often include language-independent skill descriptions like 'Microsoft Excel.' The skills in the dataset are sometimes described in German, but sometimes also given in English. Skills might be given in English while the sentence structure is German ('routers und firewalls') or similar skills are given in different languages ('Unix shell Skripte' and 'unix shell scripts'). The skill term list generation compiled a list of all skill terms from the employee profiles. This list of skill terms includes 3718 unique skill terms.

#### 5.4.1.2 Skill Term Extraction

The *skill term extraction* extracts skill terms from the textual descriptions of projects (project proposals) and project descriptions from employee profiles. The known *profile skill terms* from the *skill term list generation* are used to evaluate the performance of the *skill term extraction* in regards to the project descriptions from employee profiles. The skill term extraction uses a distributional representation of tokens to be classified to either a *skill term* or not.

The word representation, as well as the classifier selection, is subject to evaluation. The overall *skill term list* is used as positive training examples for the classification. The project descriptions serve as negative examples, after removing tokens from the *skill term list*.

A number of relevant classifiers for NER from WEKA [60] are evaluated (i.e. Naive Bayes, Maximum Entropy, Conditional Random Fields, Neural Networks and SVMs) [15]. Throughout the classifiers, the SVM classifier produced the best results and is used to provide the evaluation of different distributional representations. As the provided corpus is not large enough to learn a meaningful distributional representation, the following pre-trained embeddings corpora are used. The *FastText* [21] models are trained with a 300 dimensional *Skip-Gram* model with default parameters, obtained from the *FastText* website [47]. The *GloVe* [122] models are trained with 300 dimensions as obtained from the *GloVe* website [54].

- *FastTextDE*: *FastText* pre-trained on German *Wikipedia*.
- *FastTextEN*: *FastText* pre-trained on English *Wikipedia*.
- *GloVeCommonCrawl*: *GloVe* (840B tokens, 2.2M vocab, cased).
- *GloVeWikiGigaword*: *GloVe* (6B tokens, 400K vocab, uncased).

At first, the performance of classifying single tokens to be part of a skill term (skill token) is evaluated. The results of the evaluation of correctly classified tokens to be a token of a skill term are provided in Table 23. It can be seen, that the German *FastText* approach *FastTextDE* performs best in regards to *Precision* and *F1* score, while the English *FastText* approach performs best in terms of *Recall*. This is probably due to covering the most vocabulary entries with a total vocabulary size of 27332 words.

After recognizing skill tokens, the skill term extraction concatenates sequences of identified skill tokens to skill terms. The results of the evaluation of correctly classified skill terms are provided in Table 24. Overall, the English *FastText* classifier

Table 23: Results of skill token classification.

EMBEDDING	COVERED	COVERED	PRECISION	RECALL	F1
	SKILL TOKENS	VOCABULARY			
FastTextDE	<b>2,306</b>	<b>12,425</b>	<b>0.756</b>	0.814	<b>0.784</b>
FastTextEN	2,250	6,943	0.661	<b>0.888</b>	0.757
GloVeCommonCrawl	2,245	6,161	0.698	0.758	0.727
GloVeWikiGigaword	1,943	4,289	0.676	0.617	0.645

extracts more skills from the employee profiles as well as from the project proposals. However, the German *FastText* classifier extracts significantly more correct skill terms and performs best in *Precision*, *Recall*, and *F1*. Therefore, the German *FastText* classifier is used in further evaluation steps.

Table 24: Results of skill term classification.

EMBEDDING	EXTRACTED		TRUE	PRE-	RECALL	F1
	SKILLS	FROM				
	PROFILES	PROJECTS	TIVES			
FastTextDE	19,819	5,415	<b>2,150</b>	<b>0.106</b>	<b>0.223</b>	<b>0.135</b>
FastTextEN	<b>19,909</b>	<b>6,427</b>	620	0.029	0.067	0.038
GloVeCommonCrawl	17,825	5,917	608	0.031	0.065	0.038
GloVeWikiGigaword	15,420	5,139	578	0.034	0.063	0.041

#### 5.4.1.3 Project Assignment

For the assignment of employees to projects, a score for each worker/project combination is computed based on the extracted skills in employee profile and project proposal. The skills are given without weights in the project proposals and without quantifier in the employee profiles. Therefore, only the coverage of skills provided by an employee for the skills required by a project is used to compute this score. The set of skills required by a project  $S_p$  and the set of skills provided by an employee  $S_e$  are given. The skill coverage is defined to reflect the number of skills that are in  $S_p$  and  $S_e$ , related to the number of required skills by the project:

$$\text{coverage}(S_p, S_e) = \frac{|S_p \cap S_e|}{|S_p|} \quad (21)$$

## 5.4.2 Evaluation of Skill Extraction for Project Assignment Recommendation

### 5.4.2.1 Dataset and Evaluation metrics

For 52 of the 270 projects in the dataset, project assignments of employee profiles are given. There can be a number of 1 up to 6 employee profiles assigned to certain projects. 40 of the employee profiles are assigned to more than one project. Most projects are assigned with only a single employee profile. Overall, there are 87 individual assignments given.

In this dataset, which provides only a single assignment for many of the projects, the metrics *Precision@k* and *Recall@k* do not provide many insights. For each of the projects, the recommendation provides an ordered list of employee profiles. If the single correct employee is not given as the first recommendation, but for example, as the third recommendation, an increasing  $k$  allows that a correct recommendation (true positive) is included at some point. However, increasing  $k$  also increases the number of false positives until the position of the true positive for a project is reached. Therefore, *Precision@k* and *Recall@k* provide a high variance between different  $k$ . In such a scenario, a more detailed measure helps to evaluate the performance, instead of the continuous measures as provided by *Precision@k* and *Recall@k*. Therefore, a custom measure *matches at k* is used for evaluation, where each match within the dataset can be tracked when increasing  $k$ . *Matches at k* provides the number of true positive assignments in the whole dataset, while for each of the projects the first  $k$  recommendations are considered. This metric provides only increasing results for increasing  $k$ . The highest possible result for a  $k = 1$  in this dataset is 52, the number of projects with given assignment. The highest possible result of 87 correct assignments can only be found for  $k \geq 6$ , the highest number of assignments for a single project.

### 5.4.2.2 Evaluation of Combined Skill Extraction and Project Assignments

In the following, the recommendation of project assignments based on the skill extraction system applying the German *FastText* word embeddings to represent distributional similarities and an SVM classifier for skill term extraction is evaluated. It is evaluated against a baseline that matches skills from the skill term list in a lexical manner. Three different baselines were considered. Baseline 1 performs exact character based string matching against the skill term list. Baseline 2 tokenizes the text and extracts tokens that are found in the skill term list. Baseline 3 matches all strings from the skill term list using regular expressions. Table 25 shows the results of *matches at k* for the three baselines as well as results for the proposed methodology.

Comparing the *matches at k* for each of the evaluated methods shows, that the proposed method appears to perform best. Though baseline 2 provides the best result for *matches at 1*, the proposed method performs already best for *matches at 2* and stays best until *matches at 7*. Performing well for small values of  $k$  shows, that a good skill extraction helps to place the best matching workers on top of the ranked list for each project. After *matches at 7*, the proposed method again performs best compared to all other approaches. There seems to be no difference around *matches at 13* between

the approaches. This could be due to the small dataset of given project assignments, where at some point certain matching employees appear in the ranked lists, but no true positive match is found, as these employees might have been unavailable for the project, even if they would have been a good match. However, from *matches at 15*, the proposed method again outperforms other approaches.

Table 25: Project staffing matches at k

Values for k:	MATCHES AT K																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Baseline 1:	1	2	2	3	3	4	6	6	8	10	15	16	16	17	17	17	17	19	20	21
Baseline 2:	4	4	6	6	8	8	10	11	14	15	16	<b>18</b>	<b>19</b>	<b>20</b>	20	20	21	22	22	24
Baseline 3:	2	3	4	7	8	<b>11</b>	<b>12</b>	13	15	16	16	<b>18</b>	<b>19</b>	<b>20</b>	21	21	22	23	24	25
Proposed Method:	3	<b>6</b>	7	8	<b>9</b>	<b>11</b>	11	<b>14</b>	<b>16</b>	<b>17</b>	<b>18</b>	<b>18</b>	<b>19</b>	<b>20</b>	<b>22</b>	<b>24</b>	<b>26</b>	<b>26</b>	<b>27</b>	<b>28</b>

### 5.4.3 Conclusion

This section introduced methods for skill extraction from task descriptions to provide a skill-based project staffing approach. Approaches for skill term extraction are developed, implemented and evaluated on a real world dataset. Comparing approaches for lexical skill matching against approaches using distributional similarities for skill extraction showed, that even on a very small dataset a recommendation can be improved.

## 5.5 PRESELECTION OF DOCUMENTS FOR THE RECOMMENDATION OF JOB POSTINGS

In this section, a personalized recommendation system for job postings on a job market platform is introduced [130, 145].

Job postings are provided by a job market platform, where employers offer descriptions of currently open jobs, and potential employees select jobs to apply to (compare Section 2.2.5). A respective recommender can leverage the browsing behavior (click data) of viewed job postings per user (compare Section 2.3.4). The presented scenario provides a dataset of approximately two million job postings and the click data of about 300,000 users from a major German search engine.

Three major challenges for the identification of recommendations need to be highlighted. First, there is only implicit feedback from users in the form of click data, which contains some uncertainties. For this reason, it is more difficult to determine the interests of users. Second, the title and description of the job postings are available in raw format, whereby the properties of the job postings must be obtained on the basis of unstructured texts. Thirdly, the handling of very large amounts of data is required. The dataset contains approximately two million job postings. Therefore, scalability must be taken into account during the design.

In contrast to micro-tasks on micro-task markets, job postings on a job market platform are not completed or assigned directly on the platform. Thus, the job postings are available on the platform for some time from several days to several months. Considering the use of collaborative recommender, Melville et al. [111] report that if no user ratings are given for more than 99,5% of items, the quality of recommendation of a collaborative approach drops drastically. For the described dataset this value is at 99.9981%. Therefore, collaborative approaches are not considered.

As discussed in Section 5.1.4, Huang [76] provide a content-based recommender for job postings on the *Xing* platform using a *Doc2Vec* approach combined with an averaged KNN recommendation method. This approach is used as a baseline and referred to as *AVGKNN-D2V* in the following. To account for improved precision and scalability, a preselection approach is presented in this section, which reduces the number of job postings to consider for recommendation for each user. The respective approach based on the *AVGKNN-D2V* baseline incorporating a positive preselection of documents is referred to as *PRS-AVGKNN-D2V*. Additionally, the presented approach can be used to find negative training samples, which enables further classification methods. Therefore, an approach applying a neural network to learn a hyperplane for recommendation, as proposed by Vuurens et al. [174] for the domain of movie recommendation, is adapted to the scenario of recommending job postings. This approach, relying on the negative preselection of documents, is referred to as *PRS-HP-D2V* in the following.

Guo et al. [58] describe a recommender for the *CareerBuilder* platform, using a *BoW* approach combined with a distributed KNN. This approach serves as an additional baseline (*DSTKNN-BOW*, as it provides the best results in terms of *Precision@k* but is

unattractive due to scalability reasons, which is discussed in detail in the following section.

Zhang and Cheng [186] present an ensemble method of a content-based and a collaborative approach for recommendations on the *Xing* platform. In this approach, the *Doc2Vec* model is trained only on titles and tags and the distributed KNN algorithm is used for recommendation. Therefore, this baseline approach is referred to as DSTKNN-D2V.

This section is structured as follows. In Section 5.5.1, the method of the preselection is described. Section 5.5.2 provides the evaluation of the method, and Section 5.5.3 summarizes the results and present a conclusion.

### 5.5.1 Methodology of Preselection and Job Posting Recommendation

This section explains how the preselection of the documents is done and how it helps to calculate a personalized recommendation for job postings. At first, an overview of the process is given, explaining the structure and the components of the method. Afterward, the details of the system are presented, focusing on the preselection of documents and describing the approaches for recommendation.

An overview of the overall process of the method is given in Figure 28. The job postings are given as their corresponding textual descriptions in the set of documents  $\tilde{D} \in \tilde{D}$ . The set of users  $\tilde{u} \in \tilde{U}$  is given together with a corresponding set of ratings  $\tilde{L}$  for each user, which accounts for the number of clicks on certain documents  $\tilde{L} \subseteq \tilde{D} \times \mathbb{R}$ . In a first step, the textual descriptions in the documents are tokenized. A document  $\tilde{d} \in \tilde{D}$  is therefore represented as a sequence of tokens  $\tilde{t} \in \tilde{T}$ . All tokens of the individual documents are marked with their PoS tags from the set  $\tilde{P}$ . This is followed by the cleaning of the documents, consisting of case folding, removal of stop words and PoS filtering. The resulting documents  $\tilde{D}$  flow into the *Doc2Vec* model described in Section 2.4.3, which projects the text-based objects into a real feature space  $D \subseteq \mathbb{R}^{\phi_f}$  with  $\phi_f$  feature dimensions.

The feature set  $D$  can then be used, in a second step, to create the user profile. First, all documents are grouped into  $\phi_k$  clusters using the *mini-batch k-means* algorithm introduced in Section 2.5.1. The resulting clusters  $C_1, C_2, \dots, C_{\phi_k}$  are used for the preselection together with the ratings of the users. The goal is to identify clusters for each of the users according to their preferences. Therefore, for each user, a positive preselection of documents  $\tilde{S}^+ \subseteq \tilde{D}$  is generated as well as a negative preselection of documents  $\tilde{S}^- \subseteq \tilde{D}$ .  $\tilde{S}^+$  contains job postings that may be of interest to the user.  $\tilde{S}^-$ , on the other hand, contains job postings that are of least interest for the user. Documents within the positive subset  $\tilde{S}^+$  are used for a recommendation using the PRS-AVGKNN-D2V approach. Documents within the negative subset  $\tilde{S}^-$  are used as additional negative training samples for the PRS-HP-D2V approach. Therefore, these classification approaches are trained to calculate the user profiles  $u \in U$ . The PRS-AVGKNN-D2V approach projects the user  $\tilde{u}$  into the same vector space as the documents  $u \in U \subseteq \mathbb{R}^{\phi_f}$  using the averaged KNN method. The PRS-HP-D2V ap-

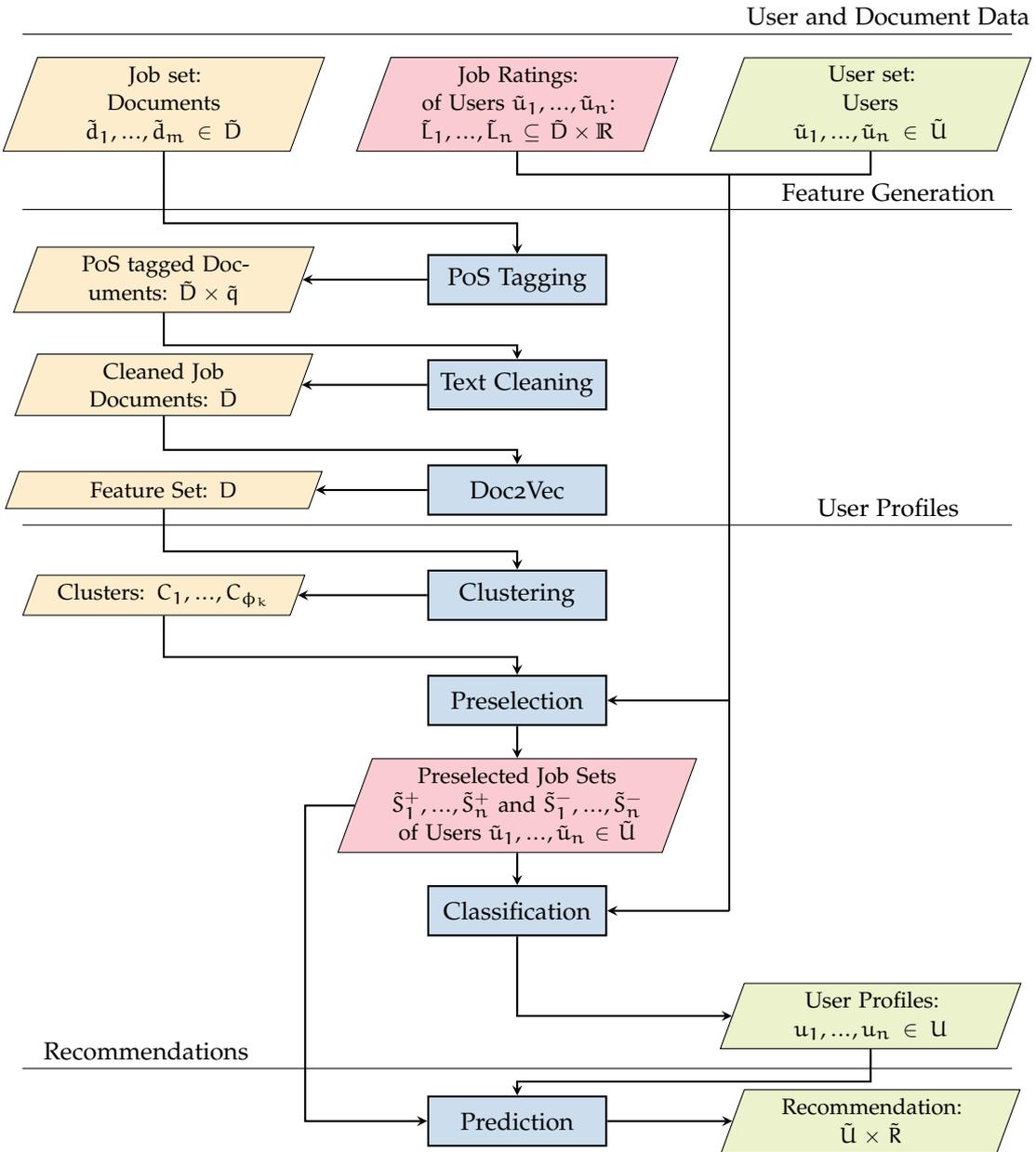


Figure 28: Structure and process of a method from the theoretical point of view.

proach learns a hyperplane within the vector space of the documents for each user  $\tilde{u}$ , which is represented by the normalized orthogonal vector  $u$  of the hyperplane.

For the final prediction of which documents to recommend from the test set in the evaluation, the respective representations of a user  $u$  and a document  $d$  are used to calculate their corresponding distance (respectively similarity) in the vector space. This allows providing the ordered set of recommendations  $\tilde{R}$  for each user, which ranks the documents  $d$  according to their distance to the user profile  $u$ . This distance or similarity depends on the choice of the similarity measure  $\text{sim} : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$  (e.g., *Cosine similarity* discussed in Section 2.4.3). Therefore, the ordered recommendation set  $\tilde{R}$  contains tuples of all items  $\tilde{d}_i, \tilde{d}_j \in \tilde{S}^+$  following the constraint of order, depending on the calculated similarity of their projected vectors  $d_i, d_j \in D$  (compare Section 2.3.1):

$$\forall(\tilde{d}_i, \tilde{d}_j) \in \tilde{R} : \text{sim}(u, d_i) \geq \text{sim}(u, d_j)$$

Each user can now be recommended the  $k$  most highly ranked documents respectively job postings.

#### 5.5.1.1 User and Document Data

To calculate the recommendations of documents  $\tilde{d}_1, \dots, \tilde{d}_m \in \tilde{D}$  to users  $\tilde{u}_1, \dots, \tilde{u}_n \in \tilde{U}$ , the user's ratings  $\tilde{L}_1, \dots, \tilde{L}_n \subseteq \tilde{D} \times \mathbb{R}$  for certain documents are required.  $\tilde{u}_i \in \tilde{U}$  represents a single user with an ID and his ratings  $\tilde{L}_i$ , provided as the frequency of user clicks on the respective job postings. It is assumed that the user prefers frequently clicked job postings to less frequently clicked job postings. However, in order not to prioritize a job posting too much, the following feedback weighting function is defined, where  $x$  is the frequency of clicks:

$$\text{fbw}(x) = \frac{x}{1 + |x|} .$$

This sigmoid feedback function is depicted in Figure 29. It can be seen that the function increases fast from 0 to 3 clicks, but almost reaches a plateau for a further increasing number of clicks. This ensures, for example, that two clicks compared to one click are weighted higher, but three clicks compared to four clicks does not provide a big difference in weighting.

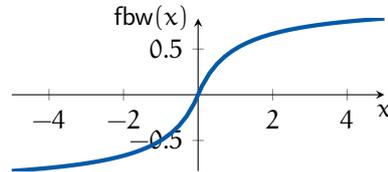


Figure 29: Sigmoid user feedback weighting function.

A job posting exists as a text-based job description with a title. The title and description are concatenated and then tokenized. Thus, a document  $\tilde{d}$  is displayed as a sequence of tokens  $\tilde{t} \in \tilde{T}$ .

### 5.5.1.2 Generation of Document Vectors

Based on the set of documents  $\tilde{D}$  the feature vectors are calculated in three substeps.

In a first step, each token  $\tilde{t} \in \tilde{T}$  of a document is assigned a PoS tag  $\tilde{p} \in \tilde{P}$ . The PoS tagger returns a sequence of PoS tags  $\tilde{p}_1, \dots, \tilde{p}_n \in \tilde{Q}$  for each document with the length  $n$ . This relates each document with its particular sequence of PoS tags ( $\tilde{D} \times \tilde{Q}$ ). The reason to use PoS tags is that the quality of features is to be improved by filtering certain PoS tags. In the article of Masuyama and Nakagawa [108] different PoS tags for text classification are compared, and better results are achieved by filtering nouns. Nouns include, for example, persons, places, professions, tools or materials. Therefore, a job description can be defined quite clearly, on the basis of nouns. Verbs include words related to an action, such as ‘backen’, ‘zeichnen’, and ‘programmieren’. Therefore, verbs can be used to describe a specific profession clearly. The question arises, whether one of the word classes is also sufficient for extracting decisive properties. Therefore, the following three PoS filters are integrated into the method. The meanings of the tags can be taken from Table 28.

N filter: NN | NE

VV filter: VVFIN | VVIMP | VVINFIN | VVIZU | VVPP

NVV filter: NN | NE | VVFIN | VVIMP | VVINFIN | VVIZU | VVPP

The *N-filter* eliminates all words in the document, except normal nouns and proper names. The *VV-Filter* filters the text on all full verbs. Auxiliary verbs such as ‘wird’ and ‘ist’ and modal verbs such as ‘können’ and ‘wollen’ are not considered, because their linguistic information is limited with respect to describing a profession. Additionally, a combination of the first two filters is provided as the *NVV-filter*, extracting both nouns and full verbs.

In a second step, further text processing steps are carried out to clean the data. This includes case folding and the elimination of stop words and special characters. Special characters within words, such as ‘IT-Consultant’, are preserved. URLs are also recognized and projected onto their hostnames. This means that URLs from a company with different depths are mapped to the same token. This results in a partial sequence of tokens  $\bar{d} \in \bar{D}$  representing each document.

In a third step, the *Doc2Vec* method projects the cleaned documents  $\bar{D}$  into the feature space  $D$ . As already mentioned in Section 2.4.3, the two models *distributed bag-of-words* and *distributed memory* are provided by *Doc2Vec*. Within the present work, the *distributed memory* model is chosen on the basis of the results presented by Le and Mikolov [95].

### 5.5.1.3 Preselection of Job Postings

The purpose of preselection is to reduce the set of job postings  $\tilde{D}$  to a subset of  $\tilde{S} \subseteq \tilde{D}$ , depending on preferences of the user. A distinction is made between the subsets  $\tilde{S}^+$  and  $\tilde{S}^-$ . The subset  $\tilde{S}^+$  is supposed to contain documents the user prioritizes. The subset  $\tilde{S}^-$  is supposed to contain the job postings that are least favored by the user.

The preselection applies a clustering approach, sorting the documents of the whole document set into clusters of similar documents. For this, the vector representations of job postings  $d \in D$  are clustered using the *mini-batch k-means* algorithm [149], as discussed in Section 2.5.1. This method is based on the *k-means* algorithm [48] and was optimized for large datasets. The clustering produces  $\phi_k$  clusters  $C_1, \dots, C_{\phi_k} D$ , containing subsets of the documents in  $D$ . Then, for each user  $\tilde{u} \in \tilde{U}$ , the clusters are ranked depending on the preferences provided by the user's ratings for the documents in  $\tilde{L}_{\tilde{u}}$ . For each of the clusters  $C_i$ , a rating  $w_i$  is calculated based on the similarity between the corresponding cluster center  $c_i$  and the vector representations  $d$  of clicked job posting documents from  $\tilde{L}_{\tilde{u}}$ , applying the previously described feedback weighting function  $\text{fbw}$  as shown in Equation 22.

$$w_i = \sum_{(\tilde{d}, x) \in \tilde{L}} \text{fbw}(x) \text{sim}(d, c_i) \quad (22)$$

As similarity measure, the *Cosine similarity* is used to compare the document vectors with each other, similarly to the works of Zhang and Cheng [186] and Huang [76]. Based on the values  $w_i$  for each cluster and for each user, the clusters can be ranked, which represents the user's preferences. To create the subsets  $\tilde{S}_{\tilde{u}}^+$  and  $\tilde{S}_{\tilde{u}}^-$  for each user  $\tilde{u}$ , the job postings from the highest respectively lowest ranked clusters are included up to a desired *selection size*  $\phi_s$ .

#### 5.5.1.4 Classification and Recommendation using KNN

The recommendation is provided as a classification task, distinguishing job postings to be recommended or not, based on the distance to a calculated user vector. This user profile  $u$  is calculated for each user  $\tilde{u} \in \tilde{U}$ . The first approach uses the averaged KNN algorithm as described by Musto et al. [116] and used by Huang [76] for the AVGKNN-D2V baseline. In this averaged KNN algorithm, a user profile  $u$  is calculated as the weighted average value of all positively rated job postings  $\tilde{L}_{\tilde{u}}^+$  of user  $\tilde{u}$ :

$$u = \frac{1}{|\tilde{L}_{\tilde{u}}^+|} \sum_{(\tilde{d}, x) \in \tilde{L}_{\tilde{u}}^+} \text{fbw}(x) d \quad (23)$$

Thus, the user  $\tilde{u}$  is represented by the user profile vector  $u$  in the same vector space as the documents. The documents  $\tilde{d} \in \tilde{D}$  are then ranked according to the distance respectively similarity of to their corresponding vectors  $d \in D$  and the user profile vector  $u$ . The  $\tilde{R}$  recommendation set of user  $\tilde{u}$  contains all elements that satisfy the condition

$$\forall(\tilde{d}_i, \tilde{d}_j) \in \tilde{R} : \text{sim}(u, d_i) \geq \text{sim}(u, d_j) \quad (24)$$

The user can then be recommended the  $k$  highest ranking documents. The complexity of this algorithm is  $O(|\tilde{D}|)$ .

Introducing the preselection to this algorithm reduces the number of considered documents  $\tilde{L}_{\tilde{u}}^+$  for each user. The positively preselected subset of the user  $\tilde{S}_{\tilde{u}}^+$  is

used to consider only such documents in  $\tilde{L}_u^+$  which are also found in  $\tilde{S}^+$ :  $\tilde{L}_u^+ \cap \tilde{S}_u^+$ . Therefore, the user profile  $u$  is calculated accordingly:

$$u = \frac{1}{|\tilde{L}_u^+ \cap \tilde{S}^+|} \sum_{(\tilde{d}, x) \in \tilde{L}_u^+} \text{fbw}(x) d \quad (25)$$

In contrast to the visual examples, *Cosine similarity* is used to compare two vectors, as in the works of Zhang and Cheng [186] and Huang [76].

As the user profile is calculated as an average value, this may provide an issue for a correct recommendation, depending on how the vector space is formed. Users who visited job postings from distinct areas in this vector space might be projected into a space between these areas, where no relevant documents are close to their user vector. The example in Figure 30 should clarify the problem. For a better understanding, the *Euclidean* distance is used in the figure for distance measurement. The different symbols indicate six different clusters. Symbols containing a + are rated positively. Symbols containing a - are considered as negatively rated, which is relevant only for the hyperplane classification in the next section. The average KNN algorithm projects the user based on the positively rated documents into the vector space. Therefore, the user is projected in between the (positive) clusters shown as a rectangle but close to the not relevant clusters shown as a triangles. For a recommendation without preselection, the three documents closest to the user are recommended, which includes these non-relevant documents (triangles), as indicated by the dashed arrows.

Introducing the preselection, only such documents are considered for recommendation, which are projected into a cluster  $C \subseteq \tilde{S}^+$  which is part of the preselection. Therefore, in the example, the preselection removes the document clusters (triangles) and leaves only the preferred clusters of the user for recommendation (rectangles). According to this example, the preselection is supposed to improve the recommendation performance.

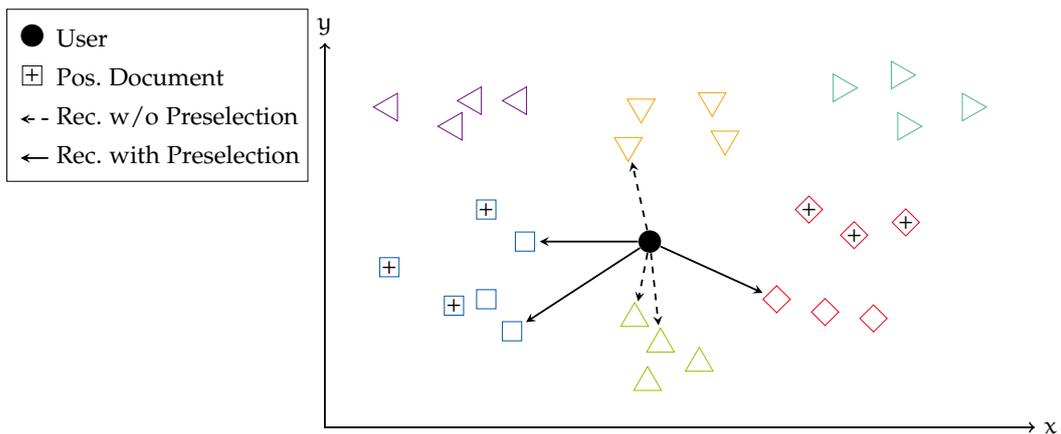


Figure 30: Example of the averaged KNN algorithm using preselection.

### 5.5.1.5 Hyperplane Classification and Recommendation

The second approach uses the hyperplane approach of Vuurens et al. [174] instead of KNN, to rank the job postings. The method optimizes for a user  $\tilde{u}$  a hyperplane so that the distance of the documents to the plane provides the ranking. The hyperplane is placed within the vector space, separating positively rated documents from negatively rated documents. Therefore, documents projected to the positive side of the hyperplane are ranked higher, the greater their distance towards the hyperplane. Documents projected to the negative side of the hyperplane are ranked lower, the greater their distance towards the hyperplane. The hyperplane is represented by a normalized orthogonal vector  $u$ , which is referred to as the user vector or user profile. The “dot product with this vector projects the [document] vectors to a one-dimensional space according to their squared distance to the hyperplane.” Therefore, the distance is calculated by the dot product of user and document vector  $u^T d$ . In the example in Figure 31 the blue line represents the user’s hyperplane, which separates the negatively valued objects from the positives. For  $k = 3$ , the three most distant documents are recommended to the user.

The goal is to position the hyperplane so that the distance  $-(u^T d_i)$  between all negatively rated documents  $\tilde{d}_i$  and the hyperplane, and the distance  $(u^T d_j)$  between all positively rated documents  $\tilde{d}_j$  and the hyperplane, is maximized. Vuurens et al. [174] achieve this by training a neural network. At first, the hyperplane is placed randomly in the vector space. Then in each training step of the neural network two differently rated documents  $d^- \in \tilde{L}^-$  and  $d^+ \in \tilde{L}^+$  of the user  $\tilde{u}$  are provided to the neural network. Therefore, this method requires positive and negative training sets of the same size. The neural network is trained using a gradient descent method. When the neural network is sufficiently trained, the hyperplane is used to determine recommendations, depending on the distance calculated from the dot product  $u^T d$ . Using the normal vector  $u$  of the user’s hyperplane all documents  $\tilde{d}_i, \tilde{d}_j \in D$  can be ranked according to the following condition:

$$\forall(\tilde{d}_i, \tilde{d}_j) \in \tilde{R} : u^T d_i \geq u^T d_j \quad (26)$$

The preselection is used to determine a negative set  $\tilde{L}^-$  for each user. In the publication of Ambati et al. [4], a negative set of objects is randomly selected from the set of all unrated objects. The number of negative training samples is equal to the number of positive training samples. This idea is applied to the present application case by forming a random subset  $\tilde{L}^- \subseteq \tilde{S}^-$ , so that the positive and negative subsets are equal in size:  $|\tilde{L}^-| = |\tilde{L}^+ \cap \tilde{S}^+|$ .

Additionally, a similar issue for the recommendation is identified as described in Section 5.5.1.4. In the case of positively rated job postings from distinct areas, intermediate job postings from areas which are not relevant to the user may be recommended. The example in Figure 31 visualizes this problem. In contrast to the example in Figure 30, this approach requires negative documents shown in the figure with a  $-$  sign. The user’s hyperplane is placed between the negative and positive documents. Without preselection, the documents from the not relevant clusters (triangles) are recommended, as indicated by the dashed arrows. Introducing the preselection

removes such documents and leaves only the preferred clusters of the user for recommendation.

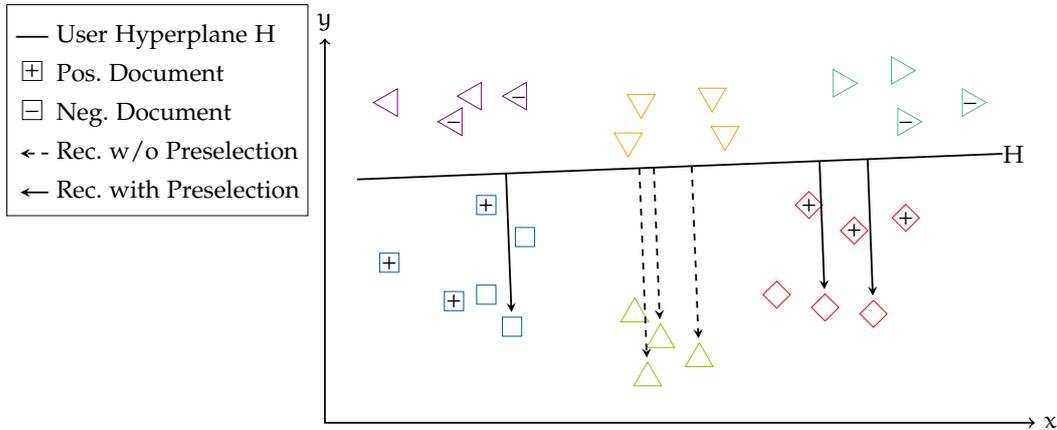


Figure 31: Example of hyperplane classification using preselection.

### 5.5.2 Evaluation of Preselection and Job Posting Recommendation

In this section, first, the dataset and the general evaluation setup are described. Three different evaluation approaches are presented: offline evaluation on the whole dataset, offline evaluation on a representative sample, and an evaluation by experts. Throughout the description of the evaluation, the baselines and the presented approaches are named according to their applied methods. The recommendation approaches relying on preselection are on the one hand the KNN approach which is referred to as PRS-AVGKNN-D2V, and the hyperplane recommendation approach as PRS-HP-D2V. The baseline of Huang [76] is referred to as AVGKNN-D2V, the method of Guo et al. [58] as DSTKNN-BOW, and the method of Zhang and Cheng [186] as DSTKNN-D2V. The evaluation is carried out to optimize the performance of the recommendation by maximizing the number of relevant documents within the first ten ranked jobs postings. Therefore, *Precision@10* is chosen as a representative evaluation measure.

#### 5.5.2.1 Evaluation Setup

The dataset contains 8,297,158 clicks from 341,638 users on 1,999,779 job postings from the second quarter of 2016 from a German job search engine. Cleaning the dataset to contain meaningful user interactions, leads to a dataset containing 300,876 users. Other users were filtered out as they were observed to be either bots, crawlers or provided too few clicks (less than ten) for further consideration.

From the original 1,999,779 documents, some of them contain only the job titles. Filtering these documents lead to a document set of 1,919,745 job postings.

Analyzing the online presence of users and documents, it can be observed, that not all job postings are available within the online period of the user. It is assumed

that the user is present within the period  $t_1$  to  $t_2$ , derived from the first and last click of the user. Accordingly, the user was only able to see documents available during this time period. It is very likely, that there exist highly relevant documents outside of this time period. They are removed from the training set for this user to reduce the possibility that such documents, which the user had no chance to provide implicit feedback for, are used as negative feedback.

### 5.5.2.2 Offline Evaluation on the whole set of Documents

The offline evaluation is performed on all documents of the dataset, sampling a representative set of users and evaluating different parameters for the preselection approaches PRS-AVGKNN-D2V and PRS-HP-D2V compared to the baseline AVGKNN-D2V. Dividing the training and test set for each user has to be done taking care of the timed series of repeated clicks on documents. Dividing training and test set first and then removing duplicate clicks may lead to an empty test set. Deleting duplicate clicks first and then dividing into train and test set solves this problem.

The number of users for the sample is calculated using the following formula [82], where a sample of  $n$  users may not exceed an absolute error of  $e$  with a probability of  $\alpha\%$ . On the basis of 12,000 users,  $s^2$  is estimated as follows.

$$n \geq z_{1-\frac{\alpha}{2}}^2 \frac{s^2}{e^2}$$

An  $\alpha$  of 95% is used for the probability, and a sample size with a maximum error of  $e = 10^{-3}$  is determined. The PRS-AVGKNN-D2V approach requires at least 3907 users with an estimated variance of  $s^2 = 1,017 \cdot 10^{-3}$ , the PRS-HP-D2V approach requires 1502 users with  $s^2 = 3,909 \cdot 10^{-4}$  and the AVGKNN-D2V baseline requires 3519 users with  $s^2 = 9,161 \cdot 10^{-4}$ . Figure 32 visualizes the *Precision@k* depending on the number of users. The dashed lines show the calculated minimum number of users for the corresponding methods. To ensure meaningful results, 10,000 users and their corresponding documents are sampled from the entire dataset for further detailed investigation.

As described in Section 5.5.1, the values for the parameters of the preselection approach have to be selected. The applied *Word2Vec* model depends on the *size of the feature vector*  $\phi_f$ . Evaluating the baseline and the approaches showed, that the *Precision* increased with the vector size up to the maximum evaluated  $\phi_f = 300$ , which is therefore chosen for the evaluation of the influence of the preselection size. The *mini-batch k-means* algorithm requires a certain *number of clusters*  $\phi_k$ . After evaluating the number of clusters between 50 and 250 with a step of 50,  $\phi_k = 100$  was fixed as the best value for the following evaluations.

The evaluation of the *preselection size*  $\phi_s$  serves as the performance evaluation of the approaches PRS-AVGKNN-D2V and PRS-HP-D2V compared to the baseline AVGKNN-D2V.

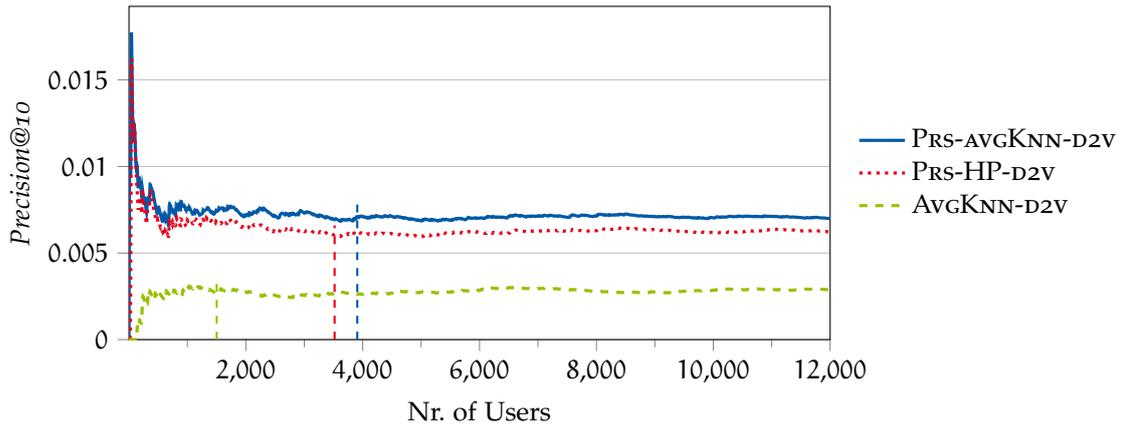


Figure 32: Determination of the sample size for meaningful results.

### 5.5.2.3 Evaluation of the Preselection

In this section, the influence of the selection size  $\phi_s$  is evaluated, whereby the sizes 50,000 to 600,000 are evaluated in steps of 50,000 documents. Figure 33 presents the  $Precision@k$  of the three methods depending on the selection size. Comparing the impact of the preselection size for the approaches using preselection, against the baseline approach without preselection, leads to the constant value for the baseline AVGKNN-D2V, which is trained with  $\frac{2}{3}$  of the documents ( $\sim 1.3$  million). The  $Precision@k$  of the two methods PRS-AVGKNN-D2V and PRS-HP-D2V increases initially with an increasing selection size. While PRS-HP-D2V stays below the AVGKNN-D2V baseline, the PRS-AVGKNN-D2V approach surpasses the baseline as soon as 300,000 documents ( $\sim 23\%$  of the AVGKNN-D2V training set) are included and reaches its maximum at 450,000 documents ( $\sim 34\%$ ). Therefore, the PRS-AVGKNN-D2V approach provides better results requiring only a fraction of the data for training. The results

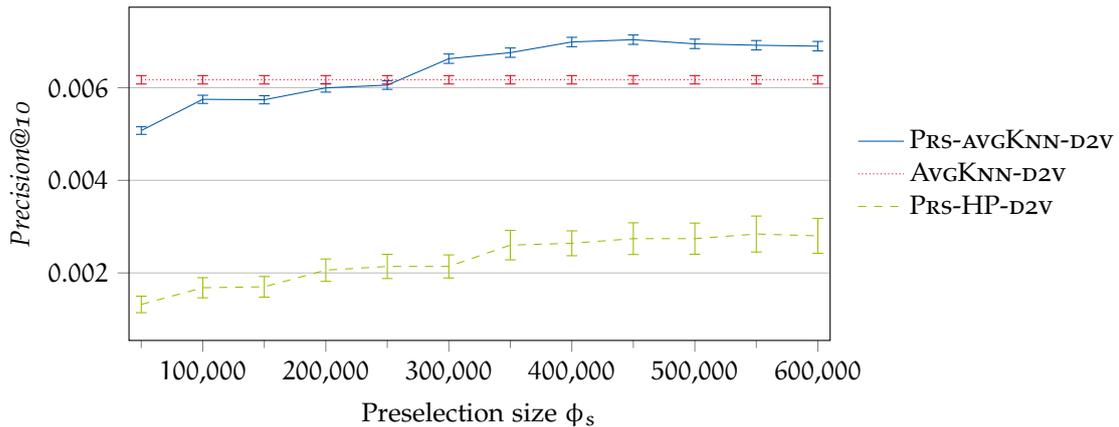


Figure 33: Comparison of selection size within preselection.

for PRS-AVGKNN-D2V and PRS-HP-D2V are influenced by the number of relevant documents in the set  $\tilde{S}^+$  of the corresponding users. The amount of non-relevant documents in the  $\tilde{S}^-$  quantity affects  $Precision@k$  for the PRS-HP-D2V approach only. Figure 34 shows the percentage of test documents (true positives) of users in the preselected sets depending on the selection size. In the best case,  $\tilde{S}^+$  covers all such documents and  $\tilde{S}^-$  should not cover any test documents. However, the coverage of the test documents in  $\tilde{S}^+$  and  $\tilde{S}^-$  grows almost linearly when increasing the selection size. This shows that the preselection, though already providing good results for the PRS-AVGKNN-D2V approach, could be further optimized, especially when selecting the negative samples  $\tilde{S}^-$  for the PRS-HP-D2V approach.

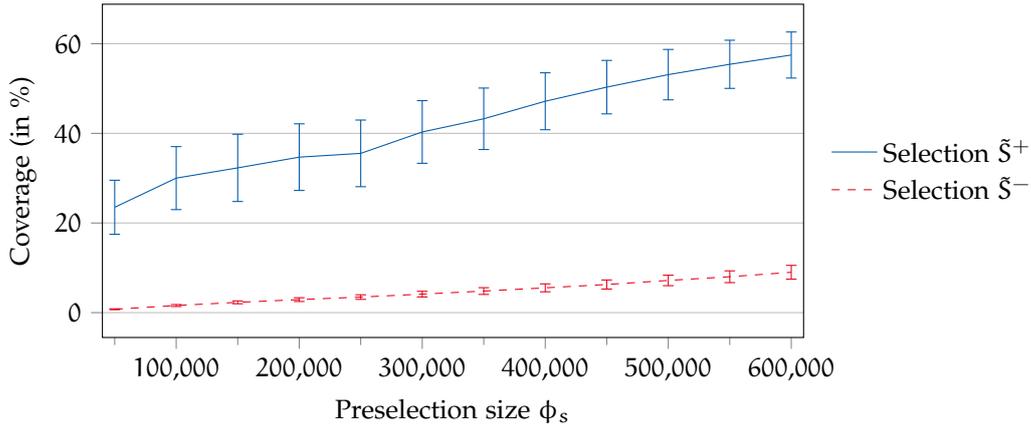


Figure 34: The test set coverage at the preselection with respect to the selection size.

#### 5.5.2.4 Offline evaluation on subsets of the entire dataset

Comparing the approaches with preselection against the AVGKNN-D2V baseline without preselection showed, that the PRS-AVGKNN-D2V approach performs better than this baseline. The other two baseline mechanisms DSTKNN-BOW and DSTKNN-D2V require a high execution time. This is due to the used KNN algorithm. The averaged KNN algorithm is dependent on the number of documents ( $O(|\tilde{D}|)$ ), as discussed in Section 5.5.1.4. The distributed KNN computes for each document  $d$  the similarity between this document and all positively rated documents. Thus, the distributed KNN depends on the number of documents and the number of positively rated documents:  $O(|\tilde{D}||\tilde{L}^+|)$ .

The comparison with this baseline mechanism is provided on the averaged results of three subsets including 1,000 users and 40,000 documents each. The details of the partial datasets are given in Table 26. It appears that the smaller amount of documents positively influences the cluster formation. Compared to the previously evaluated total dataset the number of clusters ( $\phi_k = 100$ ) is kept constant. It can be observed, that the test document coverage in the partial dataset of  $\tilde{S}^+$  is higher by  $\sim 7.1\%$  and the coverage of the  $\tilde{S}^-$  is slightly lower ( $\sim 0.2\%$ ). This should have a positive effect on the preselection approaches.

Table 26: Comparison of results on all dataset and subsets .

PARAMETER / EVALUATION MEASURE	TOTAL DATASET	PARTIAL DATASET
Record Size	$\sim 2,000,000$	40,000
Number of clusters $\phi_k$	100	100
Selection size $\phi_s$	200,000	4,000
Coverage in $\tilde{S}^+$	0.3469	0.4181
Coverage in $\tilde{S}^-$	0.0290	0.0270

Figure 35 shows the corresponding evaluation results of the compared methods. Within this evaluation, the baseline AVGKNN-D2V performs worst followed by the PRS-HP-D2V approach and the PRS-AVGKNN-D2V approach presented in this chapter. The most computationally expensive baselines DSTKNN-BOW and DSTKNN-D2V perform best. The presented PRS-AVGKNN-D2V approach is significantly outperformed by the DSTKNN-BOW baseline, while it almost reaches the performance of the DSTKNN-D2V baseline. As the presented approach PRS-AVGKNN-D2V is based on the AVGKNN-D2V baseline, it can be concluded, that the performance of the AVGKNN-D2V approach is increased using the preselection to match the performance of the DSTKNN-D2V approach. However, the PRS-AVGKNN-D2V approach is computationally less expensive than any of the evaluated methods, as can be seen in Figure 36. The baselines DSTKNN-BOW and DSTKNN-D2V both use the distributed KNN algorithm, while DSTKNN-BOW uses a BoW model and DSTKNN-D2V uses the *Doc2Vec* model. When

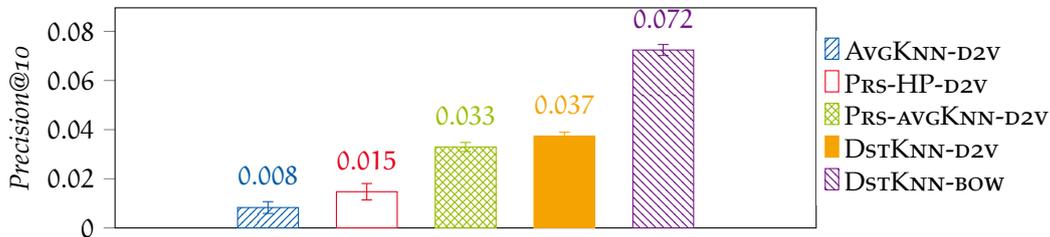


Figure 35: Comparison of methods on subsets of the whole corpus.

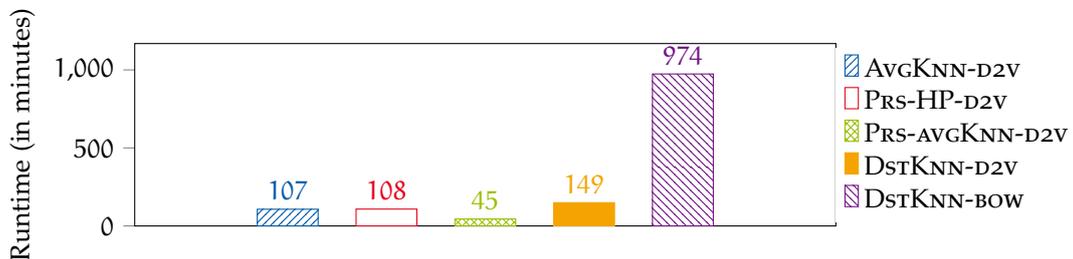


Figure 36: Comparison of runtime of methods on subsets of the whole corpus.

comparing the runtimes of the different approaches, the following has to be considered. The BoW and *Doc2Vec* model are created on the entire dataset in all evaluations. Therefore, the runtimes of feature generation are considered separately from the runtimes of the classification. The runtime information is based on the execution of the method on a PC with an Intel Core i5-2500 CPU and 16GB RAM. The CPU has four physical cores and a clock rate of 3.3GHz. The creation of the BoW model or the calculation of the TF-IDF values of all tokens takes about 587 minutes. The *Doc2Vec* model, on the other hand, requires double the runtime of about 1,203 minutes. The duration for clustering all documents is about 60 minutes and on the subsets about 8 minutes. These times are independent of the number of users and are created in advance.

Figure 36 represents runtimes concerning classification on partial datasets and therefore on 1.000 users. The PRS-AVGKNN-D2V approach reduces the runtime of the AVGKNN-D2V baseline to more than half applying the preselection. Due to the complex learning process within the neural network, the PRS-HP-D2V procedure takes similarly long as AVGKNN-D2V despite preselection. The run-times of the baselines DSTKNN-BOW and DSTKNN-D2V are higher than the other approaches based on the distributed KNN. However, it should be noted that the runtime of the distributed KNN increases quadratically with an increasing number of job postings.

#### 5.5.2.5 Expert based evaluation

In the offline evaluations, in general, a very low *Precision@k* is achieved. The reason for this is that only a small proportion of a user's actual relevant documents are known. Therefore, an expert-based evaluation is executed to assess the practical feasibility of the PRS-AVGKNN-D2V approach. Thus, a sample of 100 random users is examined by three experts. An expert is provided with ten documents from the training set and the ten most highly ranked documents from the recommendation set. The expert decides which recommendations are suitable or not, depending on the documents provided for training.

Based on this information the *Precision@k* for  $k = 1, 2, \dots, 10$  can be calculated, which is shown in Figure 37. It can be seen that about 55% of the ten documents predicted are useful recommendations for users. 58% of job postings are suitable for the user, when five documents are recommended. When a single document is recommended, this increases to 60%. This allows concluding that the presented method can be used to determine suitable recommendations which are based on the personal preferences of the user.

#### 5.5.3 Conclusion

In the presented work, a personalized recommendation system for job postings was developed. Based on the textual description of the job postings and the click data of the users, it presents a method to pre-select the documents for each user according to the similarities of task descriptions. This step leads to the elimination of non-

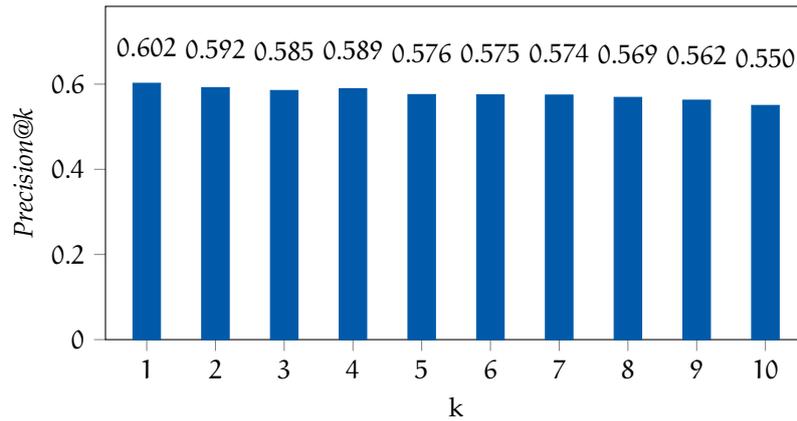


Figure 37: Evaluation of the method on the basis of expert evaluation.

relevant job postings and therefore has a positive effect on the performance of the recommendations and scalability.

Filtering nouns, verbs or their combination does not improve the prediction of job postings. The preselection delivers the best results with a selection size of 450,000, which corresponds to about a quarter of all documents. It is to be noted that with suitable parameter selection in the preselection the presented KNN approach exceeds the baseline of Huang [76]. Therefore, the preselection has a positive effect on the quality of the number of recommendations. On the partial datasets, even a four-fold improvement of the *Precision@k* compared to the method of Huang [76] can be observed, which may be due to better clustering. Ranking job postings using a hyper-plane generally produces worse results than the k-neighbor algorithm. On the partial datasets, this method exceeds the baseline of Huang [76], but this improvement is due to the preselection. This approach may be optimized by a more precise selection of negative training examples and an alternative parameter selection within the neural network. The baselines of Guo et al. [58] and Zhang and Cheng [186] provide the best results. Firstly, it can be seen that the distributed KNN performs better than the averaged KNN algorithm. However, the results of the distributed KNN can be approximately achieved by using the preselection in combination with the average KNN. The advantage of preselection in combination with the averaged KNN is that this approach requires a significantly shorter runtime than the distributed KNN.

Within the expert-based evaluation, it can be seen that about 55% of the recommendations apply to the user. Therefore, it can be concluded that the method presented in the context of this paper determines appropriate recommendations for the user.

Due to the applicability of this method, its transferability becomes relevant. It should be noted that due to the existence of the task description and user feedback, the method can also be used within micro-task markets.

To conclude the thesis, the contents are summarized, and the main contributions are discussed. Finally, a conclusion for the thesis is presented, and an outlook for further research opportunities is given.

## 6.1 SUMMARY OF THE THESIS

In Chapter 1, the challenges for task recommendation in crowdsourcing platforms were described, and the research goals were motivated. Chapter 3 provided an extensive literature review and a qualitative study with interviews from eleven experts to define requirements for the design of ECP with a focus on task assignment strategies. The derived design was implemented in a prototypical ECP and evaluated in an industry setting with employees from the enterprise of focus. The worker's perspective, their demands towards task recommendation and their perception of similarity aspects of tasks on PCPs were examined in Chapter 4. Two user studies are presented that show the requirement for task recommendations and that such recommendations have to be based on the semantic similarity of task descriptions. Especially the similarity aspect "required action" was chosen to be of relevance for workers on micro-task markets. A method to calculate the similarity between task descriptions with respect to the *required action* was designed, implemented and evaluated as described in Section 5.2. In addition, Chapter 5 provided approaches for task recommendation for several kinds of task markets from location-based micro-task markets through project staffing to job markets. A crowd selection approach for the recommendation of new location-based micro-tasks was developed on the basis of a clustering approach applying and evaluating distributional similarities like *Word2Vec*, *FastText* and *GloVe* in Section 5.3. Section 5.4 described an approach of extracting skills from project descriptions and employee profiles to provide assignment in a project staffing scenario. Section 5.5 provided a preselection approach for documents in the domain of job postings in order to enhance the performance of state-of-the-art recommender approaches based on distributional document similarities relying only on the implicit feedback of click data.

### 6.1.1 Contributions

The goal of this thesis was to define requirements towards task recommendation (RG1), design strategies for task assignment in crowdsourcing systems (RG2) and to provide task recommendation based on textual task descriptions (RG3). In contrast to most existing work, this thesis considers different perspectives to define the requirements towards task recommendation from varying viewpoints, especially the

worker's perspective. Another differentiation is the continuous consideration of textual task descriptions in the design of recommendation methods to handle the dynamic nature of task markets and the workers perspective. To reach these goals, the following contributions were presented in this thesis.

The gathering of requirements towards task recommendation in task distribution platforms by a literature review and a qualitative study leads to an extended ECP model. It is extended by the component of an incentive and compensation management and by the functions of community management and task composition and decomposition. Additionally, a classification of enterprise crowdsourcing scenarios described in the literature is provided. Taking the viewpoints of employers and employees in an ECP, a method for crowd selection and task assignment strategies are derived from the gathered requirements, developed and evaluated.

Focusing on the worker's perspective, two user studies emphasized the need for task recommendation based on information from the textual task descriptions and their semantic similarity. Here, workers from a PCP voted that the similarity of tasks is an essential factor for crowd workers to receive recommendations. Additionally, the most relevant similarity aspects of tasks are identified as perceived by the workers. Especially the most agreed upon aspect of *required action* is defined, but also the importance of semantic similarities (*purpose, context*) in contrast to factual similarities is specified.

The evaluation of the classification approach of micro-tasks showed that textual descriptions can be used to identify the kind of task which is described. Finally, different task assignment strategies were examined by providing task recommendation and crowd selection approaches. Spanning the spectrum from micro-task to job-postings, four distinct approaches are presented for recommending workers respectively tasks. A method to determine task similarities with regard to the similarity aspect *required action* is defined to provide a recommendation of micro-tasks based on verb phrase similarities. A crowd selection approach for the recommendation of location-based micro-tasks is provided, evaluating different distributional similarity measures. In the scenario of location-based crowdsourcing, the evaluation showed, that a recommendation can be improved significantly by applying the similarity of tasks. To provide project assignment recommendations, a skill extraction system is developed on the basis of project descriptions. Also, an approach for a job recommendation scenario is provided, enhancing the performance of classification algorithms based on a preselection of document clusters derived from the distributional similarities of clicked on job postings in a job market platform. The evaluation of the job posting recommendation revealed that such methods are capable of enhancing the performance while lowering the runtime of classification based recommender. Additionally, the evaluation of job recommendation showed that such methods provide relevant task recommendations to workers.

### 6.1.2 Conclusions

This thesis explored the requirements and possibilities of recommending tasks in task market platforms. Compared to existing works the presented results are derived from a focus on the worker's perspective. Additionally, it is shown, that different task market domains can benefit from the recommendation of tasks based on their textual descriptions. Taking the similarities of only the textual descriptions into account, while not relying on given categorizations or classifications of tasks renders these approaches independent from the underlying task market model, as long as textual descriptions from a worker's history are provided. The evaluation of a classification of tasks showed that pure textual descriptions can be used to define the nature of a task. The evaluations of the recommender approaches showed that this can be applied to various task markets.

## 6.2 OUTLOOK

The presented approaches provide the foundation for developing task recommendations based on analysis and comparison of textual descriptions in task markets. Especially the area of applied similarity measures can be explored further. This thesis showed that task similarities derived from textual descriptions of tasks can improve task recommendation and support the worker task selection. The fast developing field of distributional similarities found by neural networks holds the opportunity to define task similarities based on textual descriptions in an even more fine-grained manner. Following this insight, an extensive study on the effect of task recommendation can help to clarify, which similarity measures provide the best results for recommendation within these different domains.

After identifying the most valued similarity aspects of workers, this thesis provided an implementation of the similarity aspect *required action*. Following the list of identified similarity aspects, further methods to determine task similarity with respect to these similarity aspects can be defined and may be combined in a weighted manner in order to provide recommendations fitted to all needs of the workers. After *required action*, the next four similarity aspects of *comprehensibility*, *domain*, *purpose*, and *complexity* could be derived from the textual descriptions as well.

Regarding the modeling of crowdsourcing platforms and ECPs in particular, this thesis presented the design of two task assignment strategies derived from scenarios in the industry. The provided assignment strategies and workflows are defined in an abstract manner, allowing the models to be fitted to the individual requirements of enterprises implementing them. However, studying how these models are fitted in industry scenarios may help to identify further assignment strategies and to provide further enhancements on the existing models.

In general, this thesis provided requirements of task assignment in task distribution platforms, focused on the worker's perspective, and provided methods for task recommendation based on similarity aspects, crowd selection, skill extraction and preselection of documents, spanning the spectrum of tasks from micro-task to job

recommendation. In order to broaden the applicability of these methods, the focus can be broadened to focus on different perspectives towards the challenges, and include again different aspects of tasks like payment, motivation, and context to for the worker task selection.

## BIBLIOGRAPHY

---

- [1] Khaled Abdalgader and Andrew Skabar. "Short-Text Similarity Measurement Using Word Sense Disambiguation and Synonym Expansion." In: *Australasian Joint Conference on Artificial Intelligence*. Springer, 2010, pp. 435–444.
- [2] Wael Alkhatib, Christoph Rensing, and Johannes Silberbauer. "Multi-Label Text Classification Using Semantic Features and Dimensionality Reduction with Autoencoders." In: *International Conference on Language, Data and Knowledge*. Springer, 2017, pp. 380–394.
- [3] Florian Alt, Alireza Sahami Shirazi, Albrecht Schmidt, Urs Kramer, and Zahid Nawaz. "Location-Based Crowdsourcing: Extending Crowdsourcing to the Real World." In: *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*. NordiCHI '10. New York, NY, USA: ACM, 2010, pp. 13–22. ISBN: 978-1-60558-934-3.
- [4] Vamshi Ambati, Stephan Vogel, and Jaime Carbonell. "Towards Task Recommendation in Micro-Task Markets." In: *Proceedings of the 11th AAAI Conference on Human Computation*. AAAIWS'11-11. AAAI Press, 2011, pp. 80–83.
- [5] Yael Amsterdamer, Tova Milo, Amit Somech, and Brit Youngmann. "December: A Declarative Tool for Crowd Member Selection." In: *Proceedings of the VLDB Endowment* 9.13 (2016), pp. 1485–1488.
- [6] Anupriya Ankolekar, Filippo Balestrieri, and Sitaram Asur. "MET: An Enterprise Market for Tasks." In: *Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion*. New York, NY, USA: ACM, 2016, pp. 225–228. ISBN: 978-1-4503-3950-6.
- [7] Piyush Arora, Debasis Ganguly, and Gareth J. F. Jones. "The Good, the Bad and Their Kins: Identifying Questions with Negative Scores in StackOverflow." In: *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*. ASONAM '15. New York, NY, USA: ACM, 2015, pp. 1232–1239. ISBN: 978-1-4503-3854-7.
- [8] Sepehr Assadi, Justin Hsu, and Shahin Jabbari. "Online Assignment of Heterogeneous Tasks in Crowdsourcing Markets." In: *Third AAAI Conference on Human Computation and Crowdsourcing*. 2015.
- [9] Sitaram Asur, Anupriya Ankolekar, and Bernardo Huberman. "Task Crowdsourcing within an Enterprise." US20140214467 A1 (US Patent: US20140214467 A1). U.S. Classification 705/7.14; International Classification G06Q10/06; Cooperative Classification G06Q10/063112. July 2014.

- [10] Chithralekha Balamurugan, Shruti Kunde, Avantika Gupta, Deepthi Chander, and Koustuv Dasgupta. "Service Assurance Framework for Enterprise Task Crowdsourcing." In: *Services Computing (SCC), 2015 IEEE International Conference On*. IEEE, 2015, pp. 616–623.
- [11] Daniel Bär, Torsten Zesch, and Iryna Gurevych. "A Reflective View on Text Similarity." In: *RANLP*. 2011, pp. 515–520.
- [12] Ahilton Barreto, Márcio de O. Barros, and Cláudia ML Werner. "Staffing a Software Project: A Constraint Satisfaction and Optimization-Based Approach." In: *Computers & Operations Research* 35.10 (2008), pp. 3073–3089.
- [13] D. Basak. "Task Recommendation in Human Computation." PhD thesis. TU Delft, Delft University of Technology, 2014.
- [14] Martin Becker, Kathrin Borchert, Matthias Hirth, Hauke Mewes, Andreas Hotho, and Phuoc Tran-Gia. "MicroTrails: Comparing Hypotheses about Task Selection on a Crowdsourcing Platform." In: *Proceedings of the 15th International Conference on Knowledge Technologies and Data-Driven Business*. ACM, 2015.
- [15] Darina Benikova, Chris Biemann, Max Kisselew, and Sebastian Pado. "GermEval 2014 Named Entity Recognition Shared Task: Companion Paper." In: *Workshop Proceedings of the 12th KONVENS*. 2014. ISBN: 978-3-934105-47-8.
- [16] Ernst Biesalski and Andreas Abecker. "Skill-Profile Matching with Similarity Measures." In: *International Conference on Enterprise Information Systems*. Springer, 2006, pp. 210–218.
- [17] Steven Bird. "NLTK: The Natural Language Toolkit." In: *Proceedings of the Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING/ACL) on Interactive Presentation Sessions*. Association for Computational Linguistics, 2006.
- [18] Christopher Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. New York: Springer-Verlag, 2006. ISBN: 978-0-387-31073-2.
- [19] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford university press, 1995.
- [20] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. "Recommender Systems Survey." In: *Knowledge-based systems* 46 (2013), pp. 109–132.
- [21] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. "Enriching Word Vectors with Subword Information." In: *arXiv preprint arXiv:1607.04606* (2016).
- [22] Christian Borg-Krebs. "Konzeption einer EnterpriseCrowdsourcing-Plattform anhand von konkreten Anforderungen und Szenarien aus der Industrie." Master Thesis. TU Darmstadt, 2016.

- [23] Daren C. Brabham. "Moving the Crowd at iStockphoto: The Composition of the Crowd and Motivations for Participation in a Crowdsourcing Application." In: *First monday* 13.6 (2008).
- [24] Daren C. Brabham. "Moving the Crowd at Threadless: Motivations for Participation in a Crowdsourcing Application." en. In: *Information, Communication & Society* 13.8 (Dec. 2010), pp. 1122–1145. ISSN: 1369-118X, 1468-4462.
- [25] Daren C. Brabham. *Crowdsourcing*. Mit Press, 2013.
- [26] Robin Burke and Maryam Ramezani. "Matching Recommendation Technologies and Domains." In: *Recommender Systems Handbook*. Springer, 2011, pp. 367–386.
- [27] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. "Density-Based Clustering Based on Hierarchical Density Estimates." en. In: *Advances in Knowledge Discovery and Data Mining*. Ed. by Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, pp. 160–172. ISBN: 978-3-642-37456-2.
- [28] Lewis Carroll. *Through the Looking Glass: And What Alice Found There*. Rand, McNally, 1917.
- [29] Jia Wei Chang, Ming Che Lee, and Tzone I. Wang. "Integrating a Semantic-Based Retrieval Agent into Case-Based Reasoning Systems: A Case Study of an Online Bookstore." In: *Computers in Industry* 78 (2016), pp. 29–42.
- [30] Lydia B. Chilton, John J. Horton, Robert C. Miller, and Shiri Azenkot. "Task Search in a Human Computation Market." In: *Proceedings of the ACM SIGKDD Workshop on Human Computation*. ACM, 2010, pp. 1–9.
- [31] Ronan Collobert and Jason Weston. "A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning." In: *Proceedings of the 25th International Conference on Machine Learning*. ACM, 2008, pp. 160–167.
- [32] Harris M. Cooper. "Organizing Knowledge Syntheses: A Taxonomy of Literature Reviews." In: *Knowledge in Society* 1.1 (1988), pp. 104–126.
- [33] Thomas Dickinson, Miriam Fernandez, Lisa A. Thomas, Paul Mulholland, Pam Briggs, and Harith Alani. "Identifying Prominent Life Events on Twitter." In: *Proceedings of the 8th International Conference on Knowledge Capture*. ACM, 2015.
- [34] Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, and Philippe Cudré-Mauroux. "Scaling-Up the Crowd: Micro-Task Pricing Schemes for Worker Retention and Latency Improvement." en. In: *Second AAAI Conference on Human Computation and Crowdsourcing*. Sept. 2014.

- [35] Djellel Eddine Difallah, Gianluca Demartini, and Philippe Cudré-Mauroux. "Pick-a-Crowd: Tell Me What You Like, and I'll Tell You What to Do." In: *Proceedings of the 22Nd International Conference on World Wide Web*. WWW '13. New York, NY, USA: ACM, 2013, pp. 367–374. ISBN: 978-1-4503-2035-1.
- [36] Stephen Dill, Robert Kern, Erika Flint, and Melissa Cefkin. "The Work Exchange: Peer-to-Peer Enterprise Crowdsourcing." In: *First AAAI Conference on Human Computation and Crowdsourcing*. 2013.
- [37] Anhai Doan, Raghu Ramakrishnan, and Alon Y. Halevy. "Crowdsourcing Systems on the World-Wide Web." In: *Communications of the ACM* 54.4 (2011), pp. 86–96.
- [38] Julie S. Downs, Mandy B. Holbrook, Steve Sheng, and Lorrie Faith Cranor. "Are Your Participants Gaming the System?: Screening Mechanical Turk Workers." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2010, pp. 2399–2402.
- [39] David Durward, Ivo Blohm, and Jan Marco Leimeister. "Crowd Work." In: *Business & Information Systems Engineering* 58.4 (2016), pp. 281–286.
- [40] Anurag Dwarakanath, Upendra Chintala, Shrikanth N. C., Gurdeep Viridi, Alex Kass, Anitha Chandran, Shubhashis Sengupta, and Sanjoy Paul. "Crowd-Build: A Methodology for Enterprise Software Development Using Crowdsourcing." In: *Proceedings of the Second International Workshop on CrowdSourcing in Software Engineering*. Piscataway, NJ, USA: IEEE Press, 2015, pp. 8–14.
- [41] Sandy El Helou, Christophe Salzmann, and Denis Gillet. "The 3A Personalized, Contextual and Relation-Based Recommender System." In: *J. UCS* 16.16 (2010), pp. 2179–2195.
- [42] M. Erdt, A. Fernandez, and C. Rensing. "Evaluating Recommender Systems for Technology Enhanced Learning: A Quantitative Survey." In: *IEEE Transactions on Learning Technologies* PP.99 (2015), pp. 1–1. ISSN: 1939-1382.
- [43] Lee B. Erickson, I. Petrick, and E. Trauth. "Organizational Uses of the Crowd: Developing a Framework for the Study of Enterprise-Crowdsourcing." In: *Proceedings of the 2012 ACM SIGMIS Computer and People Research Conference*. 2012.
- [44] Lee B. Erickson, Eileen M. Trauth, and Irene Petrick. "Getting inside Your Employees' Heads: Navigating Barriers to Internal-Crowdsourcing for Product and Service Innovation." In: *Thirty Third International Conference on Information Systems (ICIS)*. Orlando, FL, USA, 2012.
- [45] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise." In: *Kdd*. Vol. 96. 1996, pp. 226–231.
- [46] Andrea Esuli and Fabrizio Sebastiani. "SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining." In: *Proceedings of the International Conference on Language Resources and Evaluation* (2006), pp. 417–422.

- [47] FastText. *FastText: Library for efficient text classification and representation learning*. 2018. URL: <https://fasttext.cc/index.html> (Last accessed on November 15, 11/15/2018).
- [48] Edward W. Forgy. "Cluster Analysis of Multivariate Data: Efficiency versus Interpretability of Classifications." In: *biometrics* 21 (1965), pp. 768–769.
- [49] Evgeniy Gabrilovich and Shaul Markovitch. "Computing Semantic Relatedness Using Wikipedia-Based Explicit Semantic Analysis." In: *Proceedings of the 20th International Joint Conference on Artificial Intelligence*. Vol. 6. 2007, p. 12.
- [50] David Geiger. *Personalized Task Recommendation in Crowdsourcing Systems*. Progress in IS. Cham: Springer International Publishing, 2016. ISBN: 978-3-319-22290-5.
- [51] David Geiger and Martin Schader. "Personalized Task Recommendation in Crowdsourcing Information Systems — Current State of the Art." In: *Decision Support Systems*. Crowdsourcing and Social Networks Analysis 65 (Sept. 2014), pp. 3–16. ISSN: 0167-9236.
- [52] David Geiger, Stefan Seedorf, Thimo Schulze, Robert C. Nickerson, and Martin Schader. "Managing the Crowd: Towards a Taxonomy of Crowdsourcing Processes." In: *AMCIS*. 2011.
- [53] Jochen Gläser and Grit Laudel. *Experteninterviews und Qualitative Inhaltsanalyse*. de. VS Verlag, July 2010. ISBN: 978-3-531-17238-5.
- [54] GloVe. *GloVe: Global Vectors for Word Representation*. 2018. URL: <https://nlp.stanford.edu/projects/glove/> (Last accessed on November 15, 11/15/2018).
- [55] Wael H. Gomaa and Aly A. Fahmy. "A Survey of Text Similarity Approaches." In: *International Journal of Computer Applications* 68.13 (2013), pp. 13–18.
- [56] Robert Gunning. *The Technique of Clear Writing*. New York: McGraw-Hill, 1968.
- [57] Cheng Guo, Hongyu Lu, Shaoyun Shi, Bin Hao, Bin Liu, Min Zhang, Yiqun Liu, and Shaoping Ma. "How Integration Helps on Cold-Start Recommendations." In: *Proceedings of the Recommender Systems Challenge 2017*. ACM, 2017.
- [58] Xingsheng Guo, Houssein Jerbi, and Michael P. O'Mahony. "An Analysis Framework for Content-Based Job Recommendation." In: *22nd International Conference on Case-Based Reasoning (ICCBR), Cork, Ireland*. 2014.
- [59] Ido Guy, Anat Hashavit, and Yaniv Corem. "Games for Crowds: A Crowdsourcing Game Platform for the Enterprise." In: *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*. New York, NY, USA: ACM, 2015, pp. 1860–1871. ISBN: 978-1-4503-2922-4.
- [60] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. "The WEKA Data Mining Software: An Update." In: *ACM SIGKDD Explorations Newsletter* vol. 11 (2009).
- [61] Ali Harb, Kafil Hajlaoui, and Xavier Boucher. "Competence Mining for Collaborative Virtual Enterprise." In: *Working Conference on Virtual Enterprises*. Springer, 2011, pp. 351–358.

- [62] Zellig S. Harris. "Distributional Structure." In: *Papers in Structural and Transformational Linguistics*. Springer, 1970, pp. 775–794.
- [63] Lars Hetmank. "Components and Functions of Crowdsourcing Systems – A Systematic Literature Review." In: *Wirtschaftsinformatik Proceedings* (2013).
- [64] Lars Hetmank. "Towards A Semantic Standard for Enterprise Crowdsourcing- A Scenario-Based Evaluation of a Conceptual Prototype." In: *Proceedings of the 21st European Conference on Information Systems (ECIS)*. 2013, p. 118.
- [65] Lars Hetmank. "A Synopsis of Enterprise Crowdsourcing Literature." In: *ECIS 2014 Proceedings* (2014).
- [66] Lars Hetmank. *An Ontology for Enhancing Automation and Interoperability in Enterprise Crowdsourcing Environments*. Tech. rep. Technische Universität Dresden, Fakultät Wirtschaftswissenschaften, Wirtschaftsinformatik, insb. Informationsmanagement, 2014.
- [67] Lars Hetmank. "Developing an Ontology for Enterprise Crowdsourcing." In: *Multikonferenz Wirtschaftsinformatik, Paderborn* (2014), pp. 1089–1100.
- [68] Matthias Hirth. "Modeling Crowdsourcing Platforms - A Use-Case Driven Approach." PhD Thesis. Universität Würzburg, 2016.
- [69] Matthias Hirth, Tobias Hoßfeld, and Phuoc Tran-Gia. "Anatomy of a Crowdsourcing Platform - Using the Example of Microworkers.Com." In: *Proceedings of the 5th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*. IEEE, 2011.
- [70] Matthias Hirth, Tobias Hoßfeld, and Phuoc Tran-Gia. "Analyzing Costs and Accuracy of Validation Mechanisms for Crowdsourcing Platforms." In: *Mathematical and Computer Modelling* 57.11-12 (2013), pp. 2918–2932.
- [71] Chien-Ju Ho and Jennifer Wortman Vaughan. "Online Task Assignment in Crowdsourcing Markets." In: *AAAI*. Vol. 12. 2012, pp. 45–51.
- [72] Tobias Hoßfeld, Matthias Hirth, and Phuoc Tran-Gia. "Modeling of Crowdsourcing Platforms and Granularity of Work Organization in Future Internet." In: *Proceedings of the 23rd International Teletraffic Congress*. International Teletraffic Congress, 2011, pp. 142–149.
- [73] Tobias Hoßfeld, Christian Keimel, Matthias Hirth, Bruno Gardlo, Julian Habigt, Klaus Diepold, and Phuoc Tran-Gia. "Best Practices for QoE Crowdttesting: QoE Assessment with Crowdsourcing." In: *Multimedia, IEEE Transactions on* 16.2 (2014), pp. 541–558.
- [74] Jeff Howe. "The Rise of Crowdsourcing." In: *Wired magazine* 14.6 (2006).
- [75] Jeff Howe. *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business*. Random House, 2008.
- [76] Yanbo Huang. "Exploiting Embedding in Content-Based Recommender Systems." Master's Thesis. TU Delft, Delft University of Technology, 2016.

- [77] Faizan Javed, Phuong Hoang, Thomas Mahoney, and Matt McNair. "Large-Scale Occupational Skills Normalization for Online Recruitment." In: *AAAI*. 2017, pp. 4627–4634.
- [78] Ranganathan Jayakanthan and Deepak Sundararajan. "Enterprise Crowdsourcing Solutions for Software Development and Ideation." In: *Proceedings of the Second International Workshop on Ubiquitous Crowdsourcing*. New York, NY, USA: ACM, 2011, pp. 25–28. ISBN: 978-1-4503-0927-1.
- [79] Ranganathan Jayakanthan and Deepak Sundararajan. "Enterprise Crowdsourcing Solution for Software Development in an Outsourcing Organization." In: *Current Trends in Web Engineering*. Springer, 2012, pp. 177–180.
- [80] Thorsten Joachims. *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*. Springer, 1998.
- [81] D. Jurafsky, J.H. Martin, A. Kehler, K. Vander Linden, and N. Ward. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Vol. 163. MIT Press, 2000.
- [82] Goeran Kauermann and Helmut Kuechenhoff. *Stichproben: Methoden Und Praktische Umsetzung Mit R*. Springer-Verlag, 2010.
- [83] Nicolas Kaufmann, Thimo Schulze, and Daniel Veit. "More than Fun and Money. Worker Motivation in Crowdsourcing-A Study on Mechanical Turk." In: *AMCIS*. Vol. 11. 2011, pp. 1–11.
- [84] Hanieh Javadi Khasraghi and Mohammad Jafar Tarokh. "Efficient Business Process Reengineering with Crowdsourcing." In: *International Journal of Applied Information Systems* 2.7 (2012).
- [85] kimeta GmbH. *Kimeta .de, Jobsuchmaschine*. 2018. URL: <https://www.kimeta.de/> (Last accessed on November 15, 11/15/2018).
- [86] Aniket Kittur, Ed H. Chi, and Bongwon Suh. "Crowdsourcing User Studies with Mechanical Turk." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2008, pp. 453–456.
- [87] Aniket Kittur, Jeffrey V. Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. "The Future of Crowd Work." In: *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*. ACM, 2013, pp. 1301–1318.
- [88] Ilkka Kivimäki, Alexander Panchenko, Adrien Dessy, Dries Verdegem, Pascal Francq, Hugues Bersini, and Marco Saerens. "A Graph-Based Approach to Skill Extraction from Text." In: *Proceedings of TextGraphs-8 Graph-based Methods for Natural Language Processing* (2013), pp. 79–87.
- [89] Johannes Konert, Dmitriy Burlak, and Ralf Steinmetz. "The Group Formation Problem: An Algorithmic Approach to Learning Group Formation." In: *European Conference on Technology Enhanced Learning*. Springer, 2014, pp. 221–234.
- [90] Yehuda Koren, Robert Bell, and Chris Volinsky. "Matrix Factorization Techniques for Recommender Systems." In: *Computer* 8 (2009), pp. 30–37.

- [91] Maurice Kügler, Stefan Smolnik, and Philip Raeth. "Determining the Factors Influencing Enterprise Social Software Usage: Development of a Measurement Instrument for Empirical Assessment." In: *2013 46th Hawaii International Conference on System Sciences*. IEEE, 2013, pp. 3635–3644.
- [92] Gioacchino La Vecchia and Antonio Cisternino. "Collaborative Workforce, Business Process Crowdsourcing as an Alternative of BPO." In: *International Conference on Web Engineering*. Springer, 2010, pp. 425–430.
- [93] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. "Neural Architectures for Named Entity Recognition." In: *arXiv preprint arXiv:1603.01360* (2016).
- [94] Thomas K. Landauer and Susan T. Dumais. "A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge." In: *Psychological review* 104.2 (1997), p. 211.
- [95] Quoc Le and Tomas Mikolov. "Distributed Representations of Sentences and Documents." In: *International Conference on Machine Learning*. 2014, pp. 1188–1196.
- [96] C. Leacock and M. Chodorow. "Combining Local Context and WordNet Sense Similarity for Word Sense Identification. WordNet, An Electronic Lexical Database." In: *The MIT Press* (1998).
- [97] Robert Leaman and Zhiyong Lu. "TaggerOne: Joint Named Entity Recognition and Normalization with Semi-Markov Models." In: *Bioinformatics* 32.18 (2016), pp. 2839–2846.
- [98] Vasily Leksin and Andrey Ostapets. "Job Recommendation Based on Factorization Machine and Topic Modelling." In: *Proceedings of the Recommender Systems Challenge*. ACM, 2016, p. 6.
- [99] Omer Levy, Yoav Goldberg, and Ido Dagan. "Improving Distributional Similarity with Lessons Learned from Word Embeddings." In: *Transactions of the Association for Computational Linguistics* 3 (2015), pp. 211–225.
- [100] K. Li, J. Xiao, Y. Wang, and Q. Wang. "Analysis of the Key Factors for Software Quality in Crowdsourcing Development: An Empirical Study on Top-Coder.Com." In: *Computer Software and Applications Conference (COMPSAC), 2013 IEEE 37th Annual*. IEEE, July 2013, pp. 812–817.
- [101] Kuan Liu, Xing Shi, Anoop Kumar, Linhong Zhu, and Prem Natarajan. "Temporal Learning and Sequence Modeling for a Job Recommender System." In: *Proceedings of the Recommender Systems Challenge*. ACM, 2016, p. 7.
- [102] Mariana Lopez, Maja Vukovic, and Jim Laredo. "PeopleCloud Service for Enterprise Crowdsourcing." In: *Services Computing (SCC), 2010 IEEE International Conference On*. IEEE, 2010, pp. 538–545.
- [103] Yao Lu, Sandy El Helou, and Denis Gillet. "A Recommender System for Job Seeking and Recruiting Website." In: *Proceedings of the 22nd International Conference on World Wide Web*. ACM, 2013, pp. 963–966.

- [104] Christopher S. Manning. *Foundations of Statistical Natural Language Processing*. Mit Press, July 1999.
- [105] Nikos Manouselis, Hendrik Drachler, Riina Vuorikari, Hans Hummel, and Rob Koper. "Recommender Systems in Technology Enhanced Learning." In: *Recommender Systems Handbook*. Springer, 2011, pp. 387–415.
- [106] Ke Mao, Ye Yang, Qing Wang, Yue Jia, and Mark Harman. "Developer Recommendation for Crowdsourced Software Development Tasks." In: *Service-Oriented System Engineering (SOSE), 2015 IEEE Symposium On*. IEEE, 2015, pp. 347–356.
- [107] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. "Building a Large Annotated Corpus of English: The Penn Treebank." In: *Computational linguistics* 19.2 (1993), pp. 313–330.
- [108] Takeshi Masuyama and Hiroshi Nakagawa. "Cascaded Feature Selection in SVMs Text Categorization." In: *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, 2003, pp. 588–591.
- [109] Panagiotis Mavridis, David Gross-Amblard, and Zoltán Miklós. "Using Hierarchical Skills for Optimized Task Assignment in Knowledge-Intensive Crowdsourcing." In: *Proceedings of the 25th International Conference on World Wide Web*. Montreal, Canada, Apr. 2016, pp. 843–853.
- [110] Graham McDonald, Craig Macdonald, and Iadh Ounis. "Using Part-of-Speech n-Grams for Sensitive-Text Classification." In: *Proceedings of the 2015 International Conference on The Theory of Information Retrieval*. ACM, 2015, pp. 381–384.
- [111] Prem Melville, Raymond J. Mooney, and Ramadass Nagarajan. "Content-Boosted Collaborative Filtering for Improved Recommendations." In: *Aaai/iaai* 23 (2002), pp. 187–192.
- [112] Rada Mihalcea, Courtney Corley, and Carlo Strapparava. "Corpus-Based and Knowledge-Based Measures of Text Semantic Similarity." In: *AAAI*. Vol. 6. 2006, pp. 775–780.
- [113] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. "Distributed Representations of Words and Phrases and Their Compositionality." In: *Advances in Neural Information Processing Systems*. 2013, pp. 3111–3119.
- [114] George A. Miller. "WordNet: A Lexical Database for English." In: *Communications of the ACM* 38.11 (1995), pp. 39–41.
- [115] Behrang Mohit. "Named Entity Recognition." In: *Natural Language Processing of Semitic Languages*. Springer, 2014, pp. 221–245.
- [116] Cataldo Musto, Giovanni Semeraro, Marco de Gemmis, and Pasquale Lops. "Learning Word Embeddings from Wikipedia for Content-Based Recommender Systems." In: *European Conference on Information Retrieval*. Springer, 2016, pp. 729–734.

- [117] 2010 NAACL. CSLDAMT '10: *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010.
- [118] Svenja Neitzel. "Towards Similarity-Based Task Recommendation in Crowdsourcing Systems." Master Thesis. TU Darmstadt, 2016.
- [119] Michael J. Pazzani and Daniel Billsus. "Content-Based Recommendation Systems." In: *The Adaptive Web*. Springer, 2007, pp. 325–341.
- [120] Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. "WordNet:: Similarity: Measuring the Relatedness of Concepts." In: *HLT-NAACL 2004*. Association for Computational Linguistics, 2004, pp. 38–41.
- [121] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, and Vincent Dubourg. "Scikit-Learn: Machine Learning in Python." In: *The Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.
- [122] Jeffrey Pennington, Richard Socher, and Christopher Manning. "Glove: Global Vectors for Word Representation." In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543.
- [123] Indika Perera and Pramuk A. Perera. "Developments and Leanings of Crowdsourcing Industry: Implications of China and India." In: *Industrial & Commercial Training* 46.2 (Mar. 2014), pp. 92–99. ISSN: 00197858.
- [124] Jeffrey K. Pinto and Dennis P. Slevin. "Critical Success Factors Across the Project Life Cycle." In: *Project Management Journal* 19 (1988).
- [125] Marc Poch, Núria Bel, Sergio Espeja, and Felipe Navio. "Ranking Job Offers for Candidates: Learning Hidden Knowledge from Big Data." In: *LREC*. 2014, pp. 2076–2082.
- [126] Alexandra Pomares Quimbaya, Alejandro Sierra Múnera, Rafael Andrés González Rivera, Julián Camilo Daza Rodríguez, Oscar Mauricio Muñoz Velandia, Angel Alberto García Peña, and Cyril Labbé. "Named Entity Recognition over Electronic Health Records through a Combined Dictionary-Based Approach." In: *Procedia Computer Science* 100 (2016), pp. 55–61.
- [127] Alexander J. Quinn and Benjamin B. Bederson. "Human Computation: A Survey and Taxonomy of a Growing Field." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2011, pp. 1403–1412.
- [128] Pushkar Rawat. "Crowd Selection for Task Recommendation in Location Based Crowdsourcing." Master Thesis. TU Darmstadt, 2018.
- [129] Delphine Reinhardt, Martin Michalak, and Robert Lokaiczkyk. "Job Alerts in the Wild: Study of Expectations and Effects of Location-Based Notifications in an Existing Mobile Crowdsourcing Application." In: *Proceedings of the 13th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. MOBIQUITOUS 2016. New York, NY, USA: ACM, 2016, pp. 65–74. ISBN: 978-1-4503-4750-1.

- [130] Dominik Reis. "Personalisierte und inhaltsbasierte Empfehlung von Stellenausschreibungen unter Verwendung neuronaler Netze." Master Thesis. TU Darmstadt, 2017.
- [131] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, eds. *Recommender Systems Handbook*. en. Boston, MA: Springer US, 2011. ISBN: 978-0-387-85820-3.
- [132] Sergio Rodrigues, Jonice Oliveira, and Jano Moreira de Souza. "Competence Mining for Virtual Scientific Community Creation." In: *International Journal of Web Based Communities* 1.1 (2004), pp. 90–102.
- [133] Sérgio Rodrigues, Jonice Oliveira, and Jano Moreira De Souza. "Competence Mining for Team Formation and Virtual Community Recommendation." In: *Computer Supported Cooperative Work in Design, 2005. Proceedings of the Ninth International Conference On*. Vol. 1. IEEE, 2005, pp. 44–49.
- [134] Anne C. Rouse. "A Preliminary Taxonomy of Crowdsourcing." In: *ACIS 2010 Proceedings* 76 (2010), pp. 1–10.
- [135] Peter J. Rousseeuw. "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis." In: *Journal of computational and applied mathematics* 20 (1987), pp. 53–65.
- [136] Neil Rubens, Mehdi Elahi, Masashi Sugiyama, and Dain Kaplan. "Active Learning in Recommender Systems." In: *Recommender Systems Handbook*. Springer, 2015, pp. 809–846.
- [137] Beatrice Santorini. "Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision)." In: *Technical Reports (CIS)* (1990).
- [138] R. L. Saremi and Y. Yang. "Dynamic Simulation of Software Workers and Task Completion." In: *CrowdSourcing in Software Engineering (CSI-SE), 2015 IEEE/ACM 2nd International Workshop On*. IEEE Press, May 2015, pp. 17–23.
- [139] Daniel Schall, Harald Psai, Martin Treiber, and Florian Skopik. *Engineering Service-Oriented Crowdsourcing for Enterprise Environments*. Tech. rep. 2010.
- [140] Sebastian Schmidt, Steffen Schnitzer\*, and Christoph Rensing. "Text Classification Based Filters for a Domain-Specific Search Engine." In: *Computers in Industry. Natural Language Processing and Text Analytics in Industry* 78 (2016). \*Equally contributing first author, pp. 70–79. ISSN: 0166-3615.
- [141] Sebastian Schmidt, Philipp Scholl, Christoph Rensing, and Ralf Steinmetz. "Cross-Lingual Recommendations in a Resource-Based Learning Scenario." en. In: *Towards Ubiquitous Learning*. Ed. by Carlos Delgado Kloos, Denis Gillet, Raquel M. Crespo García, Fridolin Wild, and Martin Wolpers. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, pp. 356–369. ISBN: 978-3-642-23985-4.

- [142] Steffen Schnitzer, Nihal Islam, Christian Borg-Krebs, and Christoph Rensing. *A Literature Review on the Design of Enterprise Crowdsourcing Platforms*. Tech. rep. KOM-TR-2017-02. Darmstadt, Germany: TU Darmstadt, 2017. URL: [https://www.kom.tu-darmstadt.de/research-results/publications/publications-details/?pub\\_id=SIBR17-1](https://www.kom.tu-darmstadt.de/research-results/publications/publications-details/?pub_id=SIBR17-1).
- [143] Steffen Schnitzer, Svenja Neitzel, and Christoph Rensing. "From Task Classification Towards Similarity Measures for Recommendation in Crowdsourcing Systems." In: *Presented at the 5th Conference on Human Computation and Crowdsourcing (HCOMP 2017)*. 2017. URL: <http://arxiv.org/abs/1707.06562>.
- [144] Steffen Schnitzer, Svenja Neitzel, Sebastian Schmidt, and Christoph Rensing. "Perceived Task Similarities for Task Recommendation in Crowdsourcing Systems." In: *Proceedings of the 25th International Conference Companion on World Wide Web. (WWW '16)*. International World Wide Web Conferences Steering Committee. 2016, pp. 585–590.
- [145] Steffen Schnitzer, Dominik Reis, Wael Alkhatib, Christoph Rensing, and Ralf Steinmetz. "Preselection of Documents for Personalized Recommendations of Job Postings based on Word Embeddings." In: *Proceedings of ACM SAC Conference (SAC'19)*. ACM, New York, NY, USA. Accepted for publication. 2019.
- [146] Steffen Schnitzer, Christoph Rensing, Sebastian Schmidt, Kathrin Borchert, Matthias Hirth, and Phuoc Tran-Gia. "Demands on Task Recommendation in Crowdsourcing Platforms - The Worker's Perspective." In: *Proceedings of the CrowdRec Workshop at ACM Recommender Systems Conference*. 2015.
- [147] Steffen Schnitzer, Sebastian Schmidt, Christoph Rensing, and Bettina Harriehausen-Mühlbauer. "Combining active and ensemble learning for efficient classification of web documents." In: *Polibits 49 (2014)*, pp. 39–45.
- [148] Thimo Schulze, Stefan Seedorf, David Geiger, Nicolas Kaufmann, and Martin Schader. "Exploring Task Properties in Crowdsourcing - an Empirical Study on Mechanical Turk." In: *ECIS 2011 Proceedings* (Oct. 2011).
- [149] David Sculley. "Web-Scale k-Means Clustering." In: *Proceedings of the 19th International Conference on World Wide Web*. ACM, 2010, pp. 1177–1178.
- [150] Henri Simula and Tuomas Ahola. "A Network Perspective on Idea and Innovation Crowdsourcing in Industrial Firms." In: *Industrial Marketing Management* 43.3 (Apr. 2014), pp. 400–408. ISSN: 00198501.
- [151] Henri Simula and Mervi Vuori. "Benefits and Barriers of Crowdsourcing in B2b Firms: Generating Ideas with Internal and External Crowds." In: *International Journal of Innovation Management* 16.6 (Dec. 2012), pp. 1–20. ISSN: 13639196.
- [152] Florian Skopik, Daniel Schall, and Schahram Dustdar. "Discovering and Managing Social Compositions in Collaborative Enterprise Crowdsourcing Systems." In: *International Journal of Cooperative Information Systems* 21.4 (Dec. 2012), pp. 297–341. ISSN: 02188430.

- [153] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. "Recursive Deep Models for Semantic Compositionality over a Sentiment Treebank." In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 2013, pp. 1631–1642.
- [154] Karen Sparck Jones. "A Statistical Interpretation of Term Specificity and Its Application in Retrieval." In: *Journal of documentation* 28.1 (1972), pp. 11–21.
- [155] Daniel Specht. "Automated Project Staffing based on Skill Extraction from Text." Master Thesis. TU Darmstadt, 2017.
- [156] Osamuyimen Stewart, Juan M. Huerta, and Melissa Sader. "Designing Crowdsourcing Community for the Enterprise." In: *Proceedings of the ACM SIGKDD Workshop on Human Computation*. New York, NY, USA: ACM, 2009, pp. 50–53. ISBN: 978-1-60558-672-4.
- [157] Osamuyimen Stewart, David Lubensky, and Juan M. Huerta. "Crowdsourcing Participation Inequality: A SCOUT Model for the Enterprise Domain." In: *Proceedings of the ACM SIGKDD Workshop on Human Computation*. New York, NY, USA: ACM, 2010, pp. 30–33. ISBN: 978-1-4503-0222-7.
- [158] Daniel Stieger, Kurt Matzler, Sayan Chatterjee, and Florian Ladstaetter-Fussenegger. "Democratizing Strategy: How Crowdsourcing Can Be Used for Strategy Dialogues." In: *California Management Review* 54.4 (2012), pp. 44–68. ISSN: 00081256.
- [159] Nguyen Hoang Thuan, Pedro Antunes, and David Johnstone. "Factors Influencing the Decision to Crowdsourc: A Systematic Literature Review." In: *Information Systems Frontiers* 18.1 (Feb. 2016), pp. 47–68.
- [160] Nguyen Hoang Thuan, Pedro Antunes, David Johnstone, and X. S. Ha. "Building an Enterprise Ontology of Business Process Crowdsourcing: A Design Science Approach." In: *PACIS 2015 Proceedings* (2015).
- [161] Stefano Tranquillini, Florian Daniel, Pavel Kucherbaev, and Fabio Casati. "Modeling, Enacting, and Integrating Custom Crowdsourcing Processes." In: *ACM Trans. Web* 9.2 (May 2015), 7:1–7:43. ISSN: 1559-1131.
- [162] Pooja Tripathi, Ruchi Agarwal, and Tanushi Vashishtha. "Review of Job Recommender System Using Big Data Analytics." In: *Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference On*. IEEE, 2016, pp. 3773–3777.
- [163] Tzu-Liang (Bill) Tseng, Chun-Che Huang, How-Wei Chu, and Roger R. Gung. "Novel Approach to Multi-Functional Project Team Formation." In: *International Journal of Project Management* 22.2 (Feb. 2004), pp. 147–159. ISSN: 0263-7863.
- [164] Jan Vom Brocke, Alexander Simons, Bjoern Niehaves, Kai Riemer, Ralf Platfaut, Anne Cleven, et al. "Reconstructing the Giant: On the Importance of Rigour in Documenting the Literature Search Process." In: *European Conference on Information Systems (ECIS)*. Vol. 9. 2009, pp. 2206–2217.

- [165] Maja Vukovic. "Crowdsourcing for Enterprises." In: *2009 Congress on Services*. Vol. 1. IEEE, July 2009, pp. 686–692. ISBN: 978-0-7695-3708-5.
- [166] Maja Vukovic and Claudio Bartolini. "Towards a Research Agenda for Enterprise Crowdsourcing." In: *Leveraging Applications of Formal Methods, Verification, and Validation*. Springer, 2010, pp. 425–434.
- [167] Maja Vukovic and Rajarshi Das. "Decision Making in Enterprise Crowdsourcing Services." In: *Service-Oriented Computing*. Springer, 2013, pp. 624–638.
- [168] Maja Vukovic, Jim Laredo, and Sriram Rajagopal. "Challenges and Experiences in Deploying Enterprise Crowdsourcing Service." In: *Web Engineering*. Springer, 2010, pp. 460–467.
- [169] Maja Vukovic, Jim Laredo, Yaoping Ruan, Milton Hernandez, and Sridhar Rajagopal. "Assessing Service Deployment Readiness Using Enterprise Crowdsourcing." In: *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium On*. IEEE, 2013, pp. 984–989.
- [170] Maja Vukovic, Mariana Lopez, and Jim Laredo. "PeopleCloud for the Globally Integrated Enterprise." en. In: *Service-Oriented Computing. ICSOC/ServiceWave 2009 Workshops*. Springer Berlin Heidelberg, 2010, pp. 109–114.
- [171] Maja Vukovic and Arutselvan Natarajan. "Operational Excellence in It Services Using Enterprise Crowdsourcing." In: *Services Computing (SCC), 2013 IEEE International Conference On*. IEEE, 2013, pp. 494–501.
- [172] Maja Vukovic and S. Rajagopal. "Workload Configuration and Client Strategy Discovery Using Crowdsourcing." In: *2014 IEEE Network Operations and Management Symposium (NOMS)*. IEEE, May 2014, pp. 1–15. ISBN: 978-1-4799-0913-1.
- [173] Maja Vukovic, V. Salapura, and S. Rajagopal. "Crowd-Enabled Technical Writing Services." In: *Services Computing (SCC), 2013 IEEE International Conference On*. IEEE, June 2013, pp. 635–642.
- [174] Jeroen BP Vuurens, Martha Larson, and Arjen P. de Vries. "Exploring Deep Space: Learning Personalized Ranking in a Semantic Space." In: *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 2016, pp. 23–28.
- [175] Jane Webster and Richard T. Watson. "Analyzing the Past to Prepare for the Future: Writing a Literature Review." In: *MIS quarterly* (2002), pp. xiii–xxiii.
- [176] Gary M. Weiss and Foster Provost. "Learning When Training Data Are Costly: The Effect of Class Distribution on Tree Induction." In: *Journal of Artificial Intelligence Research* (2003).
- [177] Sholom M. Weiss, Nitin Indurkha, Tong Zhang, and Fred Damerau. *Text Mining: Predictive Methods for Analyzing Unstructured Information*. Springer Science & Business Media, 2010.

- [178] Clemens Wiener, Isaac Newton Acquah, Michael Heiss, Thomas Mayerdorfer, Manfred Langen, and Walter C. Kammergruber. "Targeting the Right Crowd for Corporate Problem Solving - A Siemens Case Study with TechnoWeb 2.0." In: *Technology Management Conference (ITMC), 2012 IEEE International*. IEEE, 2012, pp. 239–247. ISBN: 978-1-4673-2134-1.
- [179] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2011.
- [180] Stephen M. Wolfson and Matthew Lease. "Look before You Leap: Legal Pitfalls of Crowdsourcing." en. In: *Proceedings of the American Society for Information Science and Technology* 48.1 (Jan. 2011), pp. 1–10. ISSN: 1550-8390.
- [181] Zhibiao Wu and Martha Palmer. "Verbs Semantics and Lexical Selection." In: *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 1994, pp. 133–138.
- [182] Philip Yaffe. "Making Sense of Nonsense: Writing Advice from Lewis Carroll and the Jabberwocky." In: *Ubiquity* 2008.May (May 2008). ISSN: 1530-2180.
- [183] Man-Ching Yuen, Irwin King, and Kwong-Sak Leung. "Task Matching in Crowdsourcing." In: *Internet of Things (iThings/CPSCoM), 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing*. IEEE, 2011, pp. 409–412.
- [184] Man-Ching Yuen, Irwin King, and Kwong-Sak Leung. "Task Recommendation in Crowdsourcing Systems." In: *Proceedings of the 1st International Workshop on Crowdsourcing and Data Mining*. ACM, 2012.
- [185] Man-Ching Yuen, Irwin King, and Kwong-Sak Leung. "Taskrec: A Task Recommendation Framework in Crowdsourcing Systems." In: *Neural Processing Letters* 41.2 (2015), pp. 223–238.
- [186] Chenrui Zhang and Xueqi Cheng. "An Ensemble Method for Job Recommender Systems." In: *Proceedings of the Recommender Systems Challenge*. ACM, 2016.
- [187] Meng Zhao, Faizan Javed, Feroosh Jacob, and Matt McNair. "SKILL: A System for Skill Identification and Normalization." In: *AAAI*. 2015, pp. 4012–4018.
- [188] Zhou Zhao, Furu Wei, Ming Zhou, Weikeng Chen, and Wilfred Ng. "Crowd-Selection Query Processing in Crowdsourcing Databases: A Task-Driven Approach." In: *EDBT*. 2015, pp. 397–408.
- [189] Shkodran Zogaj, Ulrich Bretschneider, and Jan Marco Leimeister. "Managing Crowdsourced Software Testing: A Case Study Based Insight on the Challenges of a Crowdsourcing Intermediary." In: *Journal of Business Economics* 84.3 (2014), pp. 375–405.
- [190] Shkodran Zogaj, Niklas Leicht, Ivo Blohm, and Ulrich Bretschneider. "Towards Successful Crowdsourcing Projects: Evaluating the Implementation of Governance Mechanisms." In: *International Conference on Information Systems (ICIS 2015)*. Fort Worth, USA, 2015.

- [191] Oliver Zuchowski, Oliver Posegga, Daniel Schlagwein, and Kai Fischbach. "Internal Crowdsourcing: Conceptual Framework, Structured Review, and Research Agenda." In: *Journal of Information Technology* 31.2 (2016), pp. 166–184.

*All web pages cited in this work have been checked in November 2018. However, due to the dynamic nature of the World Wide Web, their long-term availability cannot be guaranteed.*

APPENDIX

---

## A.1 INTERVIEW GUIDELINE

The interviews were held in German language. Therefore, the interview guideline is presented - as is - in German language.

*Einführung*

## Einführung in Enterprise Crowdsourcing

Crowdsourcing ist das Auslagern von Aufgaben an eine Gruppe von Nutzern über das Internet.

**Enterprise Crowdsourcing** (Internes Crowdsourcing):

- Kein Crowd Funding, sondern mit Humanressourcen
- Einrichtung eines Marktplatzes für kleine Aufgaben und nur für Mitarbeiter
- Mitarbeiter können Aufgaben aus verschiedenen Bereichen auswählen (auch Fachfremd)
- Zuordnung der Aufgaben über Mechanismen oder selbst Selektion
- Neue Arbeitsmodelle Bsp. Ein Tag die Woche frei für Crowdsourcing
- Projektunterstützung durch Enterprise Crowdsourcing

## Ziel des Interviews

Wir befassen uns mit dem Entwurf einer Enterprise Crowdsourcing Plattform und hierzu versuchen wir Anforderungen und Anwendungsszenarien aufzunehmen die von den verschiedenen Anspruchsgruppen an eine solche Plattform gestellt werden. Sie Fallen hierbei in die Gruppe:

1. Entscheider über die Einführung einer solchen Plattform
2. Möglicher Nutzer einer solchen Plattform
3. Möglicher Entwickler einer solchen Plattform

## Überleitung zu den Interviewfragen

Fragen ob Einwände gegen die Aufzeichnung und deren Verwertung bestehen?

*Leitfaden für Anwender*

## Einstiegsfrage:

1. Haben sie schon mal an Crowdsourcing teilgenommen?
  - (JA) welche Erfahrungen haben sie dabei gemacht?
  - (NEIN) Nachfrage, ob sie teilnehmen würden und was ausschlaggebend ist?

## Szenarien

1. Welche Aufgaben könnten ihrer Meinung nach mit einer Enterprise Crowdsourcing Plattform ausgeführt werden?
  - Aus ihrem Bereich
  - Aus ihrem Unternehmen
  - ggf. in der Diskussion Fragen aus den unteren Katalogen zu Vertraulichkeit, Workflows (Aufgabenverteilung), Kompetenzen etc. stellen.
2. Was sind die Voraussetzungen, die erfüllt sein müssen, damit diese Aufgaben über eine Enterprise Crowdsourcing-Plattform vergeben werden können?
1. Welche weiteren Szenarien aus ihrem Unternehmensalltag könnten ihrer Meinung nach in Aufgaben einer Enterprise Crowdsourcing Plattform einfließen?
  - Aus ihrem Bereich
  - Aus ihrem Unternehmen
  - ggf. in der Diskussion Fragen aus den unteren Katalogen zu Vertraulichkeit, Workflows (Aufgabenverteilung), Kompetenzen etc. stellen.
2. Was sind die Voraussetzungen, die erfüllt sein müssen, damit solche Szenarien in eine Enterprise Crowdsourcing-Plattform einfließen?

## Integration in Systemlandschaft:

1. Welche Kollaborations- und Ticketsysteme setzen sie in ihrem Unternehmen ein?
  - Bei welchen dieser Tools sehen sie die Möglichkeit eine Enterprise Crowdsourcing-Plattform einzubinden?
  - Bei welchen dieser Tools halten sie es für notwendig eine Enterprise Crowdsourcing-Plattform einzubinden?
  - Welche dieser Tools könnten in die Enterprise Crowdsourcing eingebunden werden?

## Vertraulichkeit:

1. Sollen alle Aufgaben auf der Plattform für alle sichtbar sein? (Werksstudent bis Management)

- (JA)
- (NEIN) Wie sollte die Sichtbarkeit eingeschränkt werden?
  - Alle Personen im Unternehmen (Firmengeheimnisse Idots)
  - Nur feste Mitarbeiter
  - Abteilung
  - Zeitlich ...
  - Qualifikation
  - Standort
  - Erfahrung

Workflow (Jobs, Ressourcen Freigabe): (Besonders intensiv bei Entscheidern)

1. Wie sollte ihrer Meinung nach die Vergabe der Aufgaben geschehen?
  - Automatisiert
  - Selbstselektion
  - Vorschläge
  - Vorgesetzter
2. Sollte bei der Vergabe eine weitere Instanz (Vorgesetzter) mitbestimmen?
3. Sollte jeder uneingeschränkt Aufgaben einstellen können?
  - (Nein) Welche Einschränkungen würden sie Vorschlagen?
    - Position im Unternehmen
    - Qualifikation
    - Vertraulichkeit
    - Ggf. Prüfung durch Vorgesetzten
    - ...

Kompetenzen (Tools Software, Handicaps ...):

1. Wäre für sie Kompetenzaufbau ein Motiv die Enterprise Crowdsourcing Plattform zu nutzen?
2. Welche Kompetenzen, Eigenschaften und Kenntnisse können mit Enterprise Crowdsourcing Angesprochen werden?
3. Welche Kompetenzen, Eigenschaften und Kenntnisse werden für Aufgaben in ihrem Bereich benötigt?
  - Lizenzen (Tools)
  - Qualifikation
  - Körperliche Eigenschaften
  - Erfahrung

### Erwartungen

1. Welche Verbesserungen Versprechen sie für sich selbst von einer solchen Plattform?
2. Welche Verbesserungen sehen sie für das Unternehmen durch eine solche Plattform?

### Privatsphäre / Datensicherheit:

1. Wer sollte die Auswertung der Plattform machen dürfen? (Rollen)
  - Team
  - Direkter Vorgesetzter
  - Management
  - Betriebsrat
  - ...

### Abschluss des Interviews

1. Gibt es Fragen ihrerseits?
2. Weiteres Vorgehen
3. ...

*Leitfaden für Entscheider*

1. Zunächst Fragen der Anwender
2. Haben sie schon mal über die Einführung einer Enterprise Crowdsourcing-Plattform nachgedacht?
3. Besteht langfristig Interesse an der Einführung einer solchen Plattform in ihrem Unternehmen?
4. Was sind die ausschlaggebenden Gründe für Sie solch eine Plattform einzuführen?
5. Was sind Gründe solch eine Plattform nicht einzuführen?
6. In wie weit wären sie bereit Ressourcen für die Einführung und den Betrieb einer Enterprise Crowdsourcing-Plattform bereitzustellen?
7. Was wären ihre Erwartungen an eine Enterprise Crowdsourcing-Plattform?
8. Was wären Messwerte/Kennzahlen die zur Messung des Erfolges einer Enterprise Crowdsourcing-Plattform für sie notwendig sind?

## Abschluss des Interviews

1. Gibt es Fragen ihrerseits?
2. Weiteres Vorgehen
3. ...

## A.2 POS-TAGS

The following tables list common PoS tags for the English language in Table 27, and used PoS tags for the German language in Table 28.

Table 27: Selection of common PoS tags from the Penn treebank tag set [107].

TAG	PART-OF-SPEECH
DT	Determiner
IN	Preposition or subordinating conjunction
JJ	Adjective
NN	Noun, singular or mass
NNP	Proper noun, singular
NNS	Noun, plural
RB	Adverb
TO	to
VB	Verb, base form
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present

Table 28: Section from the STTS-day table of the University of Stuttgart:  
<http://www.ims.uni-stuttgart.de/forschung/ressourcen/lexika/TagSets/stts-table.html>  
 Lastchecked:December19th,2018)

POS TAG	DESCRIPTION	EXAMPLE
NN	normales Nomen	Tisch, Herr, [das] Reisen
NE	Eigennamen	Hans, Hamburg, HSV
VVFIN	finites Verb, voll	[du] gehst, [wir] kommen [an]
VVIMP	Imperativ, voll	komm [!]
VVINF	Infinitiv, voll	gehen, ankommen
VVIZU	Infinitiv mit "zu", voll	anzukommen, loszulassen
VVPP	Partizip Perfekt, voll	gegangen, angekommen

## A.3 ADDITIONAL RESULTS FOR DETERMINING TASK SIMILARITIES

*Results of Classification*

Table 29 shows evaluation results for the classification of micro-tasks from *Microworkers*. The results show the different feature sets for JRip, Support Vector Machine (SMO) and Random Forest classifiers. Given are the weighted average accuracy (Acc), the weighted average *Precision* (Prec), the weighted average *Recall* (Rec) and weighted average F1-score (F1). The highest F1-scores for every classifier are highlighted in boldface.

Table 29: Classification evaluation results.

FEATURESET	JRIP				SMO				RANDOM FOREST			
	ACC	PREC	REC	F1	ACC	PREC	REC	F1	ACC	PREC	REC	F1
factual ( <i>fac</i> )	0.83	0.82	0.83	0.82	0.76	0.72	0.76	0.73	0.87	0.86	0.87	0.86
structural ( <i>str</i> )	0.77	0.74	0.77	0.74	0.64	0.50	0.64	0.54	0.83	0.82	0.83	0.81
semantic ( <i>sem</i> )	0.78	0.81	0.78	0.75	0.85	0.87	0.85	0.84	0.84	0.84	0.84	0.83
content ( <i>con</i> )	0.91	0.91	0.91	0.91	0.92	0.92	0.92	0.92	0.93	0.93	0.93	<b>0.92</b>
con (+ proof + title)	0.94	0.94	0.94	<b>0.94</b>	0.97	0.97	0.97	<b>0.96</b>	0.95	0.94	0.95	<b>0.94</b>
con (+ TF-IDF) ( <i>con*</i> )	0.92	0.92	0.92	<b>0.92</b>	0.94	0.94	0.94	<b>0.94</b>	0.93	0.92	0.93	<b>0.92</b>
con (+ stemming)	0.92	0.92	0.92	<b>0.92</b>	0.92	0.91	0.92	0.91	0.92	0.92	0.92	<b>0.92</b>
con (+ bi- + trigrams)	0.92	0.91	0.92	<b>0.92</b>	0.92	0.92	0.92	0.92	0.92	0.91	0.92	0.91
con* & fac	0.92	0.92	0.92	0.91	0.93	0.93	0.93	0.93	0.92	0.92	0.92	0.91
con* & str	0.91	0.91	0.91	0.91	0.94	0.94	0.94	<b>0.94</b>	0.91	0.91	0.91	0.91
con* & sem	0.92	0.91	0.92	0.91	0.94	0.93	0.94	0.93	0.93	0.92	0.93	<b>0.92</b>
fac & sem	0.85	0.84	0.85	0.84	0.87	0.86	0.87	0.86	0.87	0.87	0.87	0.86
fac & str	0.84	0.83	0.84	0.83	0.84	0.82	0.84	0.82	0.88	0.87	0.88	0.87
sem & str	0.81	0.80	0.81	0.79	0.86	0.86	0.86	0.85	0.86	0.86	0.86	0.84
fac & sem & str	0.85	0.84	0.85	0.84	0.87	0.87	0.87	0.86	0.87	0.87	0.87	0.86
con* & fac & sem	0.92	0.92	0.92	0.91	0.93	0.93	0.93	0.92	0.91	0.91	0.91	0.90
con* & fac & str	0.91	0.91	0.91	0.90	0.93	0.93	0.93	0.93	0.91	0.91	0.91	0.90
con* & sem & str	0.92	0.92	0.92	<b>0.92</b>	0.94	0.93	0.94	0.93	0.91	0.91	0.91	0.90
con* & fac & sem & str	0.92	0.91	0.92	0.91	0.93	0.93	0.93	0.93	0.91	0.91	0.91	0.90

*Evaluation of Similarity Measures*

Table 30 shows the evaluation results of the distance values regarding the method to calculate the similarity with respect to the similarity aspect *required action*.

Table 30: Evaluation of the distance values regarding the aspect *required action*. The Spearman rank correlation ( $r_s$ ) and the mean absolute error (mae) with respect to the gold standard are given for rankings  $R_1, \dots, R_{10}$ .

		$R_1$	$R_2$	$R_3$	$R_4$	$R_5$	$R_6$	$R_7$	$R_8$	$R_9$	$R_{10}$
$WN_{path}$	$r_s$	0.697	0.310	<b>0.636</b>	<b>0.673</b>	0.527	0.382	0.358	0.248	0.333	<b>0.636</b>
	mae	5.722	6.304	7.569	6.171	6.024	5.591	6.920	8.046	6.760	8.130
$WN_{wup}$	$r_s$	0.624	0.182	0.212	0.600	<b>0.552</b>	0.297	0.164	0.091	0.358	0.309
	mae	5.161	5.907	7.823	5.877	4.702	5.632	5.982	7.498	5.340	7.110
$WN_{lch}$	$r_s$	0.321	0.085	0.079	0.358	0.321	<b>0.503</b>	0.273	-0.067	0.297	-0.067
	mae	6.158	5.106	6.220	5.468	5.023	4.777	4.678	7.403	4.797	6.376
$WN_{path}$ no URL hosts	$r_s$	0.358	<b>0.498</b>	0.430	0.588	0.491	0.273	0.345	0.309	<b>0.588</b>	0.564
	mae	5.825	5.684	7.901	6.262	6.074	5.331	6.722	8.685	5.898	8.277
$WN_{path}$ cplx. grammar	$r_s$	<b>0.830</b>	0.134	<b>0.636</b>	<b>0.673</b>	0.491	0.358	0.200	0.091	0.406	0.455
	mae	3.919	5.244	6.042	5.558	5.596	4.227	6.312	7.277	6.202	7.111
$WN_{path}$ whole corpus	$r_s$	0.564	0.340	0.103	0.636	0.297	0.248	<b>0.430</b>	<b>0.370</b>	0.370	0.406
	mae	6.745	5.025	7.291	5.391	5.170	4.652	5.298	7.640	4.955	7.865

## Clustering Similar Tasks – Word Frequencies

Table 31 and Table 32 show the word frequencies of the 15 most frequent words in the 14 clusters.

Table 31: Word frequencies obtained with DBSCAN (eps = 2, minPts = 10).

	A <sub>1</sub>	A <sub>2</sub>	A <sub>3</sub>	A <sub>4</sub>	A <sub>5</sub>	A <sub>6</sub>	A <sub>7</sub>
	735 tasks 54429 words	133 tasks 27664 words	124 tasks 14326 words	76 tasks 1786 words	51 tasks 1814 words	36 tasks 1367 words	35 tasks 1188 words
1	dst (2045)	post (1596)	least (750)	dst (78)	dst (175)	email (81)	dst (125)
2	profile (1959)	forum (798)	twitter (496)	search (72)	page (120)	go (63)	website (65)
3	google (1488)	url (798)	account (490)	go (70)	open (92)	select (60)	open (63)
4	task (1036)	dst (665)	recent (384)	must (67)	proof (84)	sign (54)	proof (63)
5	url (851)	content (665)	tweet (372)	click (50)	number (82)	address (50)	number (62)
6	keyword (839)	use (665)	following (372)	close (44)	website (76)	name (40)	page (52)
7	least (818)	make (665)	requirements (372)	task (43)	search (61)	already (37)	keyword (38)
8	make (739)	page (399)	meets (372)	version (39)	keyword (59)	dst (36)	search (36)
9	go (729)	new (399)	dst (372)	us (39)	look (49)	paid (35)	results (33)
10	click (674)	comment (399)	retweet (360)	browser (35)	title (41)	optional (29)	found (32)
11	page (659)	forums (399)	retweets (335)	tab (35)	sure (41)	enter (28)	look (31)
12	sure (652)	website (399)	tweets (266)	google (34)	results (40)	please (26)	description (30)
13	post (638)	else (399)	count (248)	complete (34)	found (40)	different (26)	title (27)
14	code (524)	link (399)	given (248)	app (33)	following (39)	signed (25)	sure (27)
15	submitted (505)	find (266)	need (248)	spine (32)	make (39)	fields (24)	make (27)

Table 32: Word frequencies obtained with DBSCAN (eps = 2, minPts = 10).

	A <sub>8</sub>	A <sub>9</sub>	A <sub>10</sub>	A <sub>11</sub>	A <sub>12</sub>	A <sub>13</sub>	A <sub>14</sub>
	27 tasks 2349 words	25 tasks 2300 words	23 tasks 1079 words	19 tasks 691 words	18 tasks 340 words	13 tasks 179 words	12 tasks 207 words
1	dst (108)	dst (100)	dst (128)	app (58)	comment (28)	rate (26)	email (13)
2	url (81)	website (100)	search (115)	review (46)	go (21)	offering (13)	go (12)
3	tasks (54)	links (75)	google (92)	write (28)	dst (21)	dst (13)	click (12)
4	pin (54)	create (75)	website (92)	honest (22)	video (16)	best (13)	fill (12)
5	text (54)	profile (75)	note (69)	install (19)	leave (14)	button (13)	dst (12)
6	go (54)	link (75)	find (59)	feedback (18)	click (11)	green (13)	required (11)
7	provided (54)	url (75)	result (46)	dst (18)	like (9)	select (13)	confirm (8)
8	pinterest (54)	within (50)	keyphrase (46)	note (17)	page (9)	go (13)	sign (7)
9	board (54)	shown (50)	results (23)	download (17)	add (7)	table (13)	information (7)
10	newly (54)	new (50)	code (23)	least (17)	youtube (7)	next (13)	log (6)
11	javascript (54)	address (50)	go (23)	words (16)	article (6)	click (13)	address (5)
12	created (54)	using (50)	close (23)	see (14)	search (6)	bank (13)	join (4)
13	already (54)	post (50)	click (23)	go (13)	must (5)	apr (5)	registration (3)
14	bookmarklet (54)	note (50)	write (23)	use (10)	mixtape (5)	cd (5)	xnumk (3)
15	ignored (27)	ignored (25)	generated (23)	must (9)	watch (4)	- (0)	- (0)



## A.4 ACRONYMS AND KEYWORDS

A.4.1 *Acronyms*

BoW	Bag-of-Words 18, 67, 70, 73, 78, 98, 110, 111
CBOW	Continuous Bag of Words 20
DF	Document Frequency 18
ECP	Enterprise Crowdsourcing Platform 2, 5, 6, 10, 25–33, 35, 36, 38–48, 53, 113–115, 143
HIT	Human Intelligence Task 9
KNN	K-Nearest-Neighbor 22, 68, 70, 71, 74, 98, 99, 103–106, 109–112
ML	Machine Learning 1, 4, 7, 9, 13, 16, 21, 22, 24, 45
NEN	Named Entity Normalization 17, 69
NER	Named Entity Recognition 16, 17, 69, 74, 94
NLP	Natural Language Processing 1, 4, 5, 7, 13, 16
NLTK	Natural Language Tool Kit 74, 78, 79
PCP	Public Crowdsourcing Platform 1, 2, 10, 25, 26, 45, 55, 57, 58, 60, 67, 113, 114, 143
PoS	Part-of-Speech 16, 17, 74, 78, 79, 99, 102, 138
SVM	Support Vector Machine 22, 70, 74, 94, 96, 139
TF	Term Frequency 18
TF-IDF	Term Frequency – Inverse Document Frequency 18, 20, 67, 73, 74, 76, 78, 89, 111, 139

A.4.2 *Keywords*

Accuracy	22, 139
CareerBuilder	70, 98
Cosine similarity	19, 70, 79, 80, 101, 103, 104
Cross validation	24, 76, 90
Crowdsourcing platform	2, 3, 5, 10, 26, 40, 55, 113, 115 see: ECP 5, 10, 15, 65, 86, 88, 90, 91, 113, 114 see: PCP
DBSCAN	21, 82, 89, 141
Distributed bag-of-words	20, 102

Distributed memory	20, 21, 102
Doc2Vec	20, 70, 71, 89, 98, 99, 102, 110, 111
Enterprise crowdsourcing	4, 15, 25–33, 35, 39, 42, 44, 45, 49, 114
F1	23, 76, 77, 89, 94, 95, 139
FastText	20, 89, 94–96, 113
GloVe	20, 94, 113
HDBSCAN	21, 89
Job posting	2, 4–7, 10, 15, 16, 65, 66, 70, 71, 98, 99, 101–106, 111–114
K-means	21, 70, 89, 103
Matrix factorization	6, 19, 66, 70, 86, 90
Micro-task	1, 4, 6–10, 14, 65, 72, 77, 88, 98, 113–115, 139
Micro-task market	2, 4, 5, 8–11, 13, 14, 25, 55, 65, 67, 74, 78, 84–86, 90, 91, 98, 112, 113
Microworkers	2, 57, 58, 60, 72, 74, 84, 85, 139
Mini-batch k-means	21, 99, 103, 107
MTurk	2, 8, 9, 26, 55, 56
N-gram	18–20, 67, 73, 74, 76, 78
Neural network	4, 16, 17, 20, 22, 69, 94, 98, 105, 115
Open source	1, 2, 7, 8, 31, 74
Precision	23, 90, 92, 94, 95, 107, 139
Precision at k	23, 24, 90, 91, 96, 98, 106–109, 111, 112,
Public crowdsourcing	25, 26
R-Precision	24, 89–91
Recall	23, 92, 94, 95, 139
Recall at k	23, 24, 90, 91, 96,
Singular value decomposition	19, 71, 90
Skip-Gram	20, 94
WEKA	74, 94
Wikipedia	1, 2, 7–9, 20, 69, 94
Word embedding	5, 16, 17, 20, 22, 85, 96
WordNet	20, 72, 78–82
Xing	11, 70, 98, 99

## A.5 SUPERVISED STUDENT THESES

- [1] Madhukar Chinnappa. "Using Word Embeddings and Deep Learning for Task Recommendations in Crowdsourcing Systems." Master Thesis. TU Darmstadt, 2018.
- [2] Nils Jansen. "Task Recommendation on Crowdsourcing Platforms - Methods for Task Similarity Computation." Bachelor Thesis. TU Darmstadt, 2018.
- [3] Pushkar Rawat. "Crowd Selection for Task Recommendation in Location Based Crowdsourcing." Master Thesis. TU Darmstadt, 2018.
- [4] Dominik Reis. "Personalisierte und inhaltsbasierte Empfehlung von Stellenausschreibungen unter Verwendung neuronaler Netze." Master Thesis. TU Darmstadt, 2017.
- [5] Daniel Specht. "Automated Project Staffing based on Skill Extraction from Text." Master Thesis. TU Darmstadt, 2017.
- [6] Christian Borg-Krebs. "Konzeption einer EnterpriseCrowdsourcing-Plattform anhand von konkreten Anforderungen und Szenarien aus der Industrie." Master Thesis. TU Darmstadt, 2016.
- [7] Svenja Neitzel. "Towards Similarity-Based Task Recommendation in Crowdsourcing Systems." Master Thesis. TU Darmstadt, 2016.
- [8] Aditi Saini. "Dynamic Pricing in a Location Based Crowdsourcing Platform." Master Thesis. TU Darmstadt, 2016.



AUTHOR'S PUBLICATIONS

---

## JOURNAL ARTICLES

- [1] Sebastian Schmidt, Steffen Schnitzer\*, and Christoph Rensing. "Text Classification Based Filters for a Domain-Specific Search Engine." In: *Computers in Industry. Natural Language Processing and Text Analytics in Industry 78* (2016). \*Equally contributing first author, pp. 70–79. ISSN: 0166-3615.
- [2] Steffen Schnitzer, Sebastian Schmidt, Christoph Rensing, and Bettina Harriehausen-Mühlbauer. "Combining active and ensemble learning for efficient classification of web documents." In: *Polibits 49* (2014), pp. 39–45.

## CONFERENCE, WORKSHOP AND FURTHER ARTICLES

- [3] Steffen Schnitzer, Dominik Reis, Wael Alkhatib, Christoph Rensing, and Ralf Steinmetz. "Preselection of Documents for Personalized Recommendations of Job Postings based on Word Embeddings." In: *Proceedings of ACM SAC Conference (SAC'19)*. ACM, New York, NY, USA. Accepted for publication. 2019.
- [4] Steffen Schnitzer, Nihal Islam, Christian Borg-Krebs, and Christoph Rensing. *A Literature Review on the Design of Enterprise Crowdsourcing Platforms*. Tech. rep. KOM-TR-2017-02. Darmstadt, Germany: TU Darmstadt, 2017. URL: [https://www.kom.tu-darmstadt.de/research-results/publications/publications-details/?pub\\_id=SIBR17-1](https://www.kom.tu-darmstadt.de/research-results/publications/publications-details/?pub_id=SIBR17-1).
- [5] Steffen Schnitzer, Svenja Neitzel, and Christoph Rensing. "From Task Classification Towards Similarity Measures for Recommendation in Crowdsourcing Systems." In: *Presented at the 5th Conference on Human Computation and Crowdsourcing (HCOMP 2017)*. 2017. URL: <http://arxiv.org/abs/1707.06562>.
- [6] Steffen Schnitzer, Svenja Neitzel, Sebastian Schmidt, and Christoph Rensing. "Perceived Task Similarities for Task Recommendation in Crowdsourcing Systems." In: *Proceedings of the 25th International Conference Companion on World Wide Web. (WWW '16)*. International World Wide Web Conferences Steering Committee. 2016, pp. 585–590.
- [7] Steffen Schnitzer, Christoph Rensing, Sebastian Schmidt, Kathrin Borchert, Matthias Hirth, and Phuoc Tran-Gia. "Demands on Task Recommendation in Crowdsourcing Platforms - The Worker's Perspective." In: *Proceedings of the CrowdRec Workshop at ACM Recommender Systems Conference*. 2015.

## CO-AUTHORED PUBLICATIONS

- [8] Wael Alkhatib, Eid Araache, Christoph Rensing, and Steffen Schnitzer. "Ensuring Novelty and Transparency in Learning Resource-Recommendation Based on Deep Learning Techniques." In: *European Conference on Technology Enhanced Learning*. Springer, 2018, pp. 609–612.
- [9] Wael Alkhatib, Steffen Schnitzer, Wei Ding, Peter Jiang, Yassin Alkhalili, and Christoph Rensing. "Comparison of Feature Selection Techniques for Multi-label Text Classification against a New Semantic-based Method." In: *19th International Conference on Computational Linguistics and Intelligent Text Processing (CICLING '19)*. Accepted for publication. 2018.
- [10] Kathrin Borchert, Matthias Hirth, Steffen Schnitzer, and Christoph Rensing. "Impact of Task Recommendation Systems in Crowdsourcing Platforms." In: *Workshop on Responsible Recommendation at RecSys 2017*. 2017.
- [11] Sebastian Schmidt, Steffen Schnitzer, and Christoph Rensing. "Domain-Independent sentence type classification: examining the scenarios of scientific abstracts and scrum protocols." In: *Proceedings of the 14th International Conference on Knowledge Technologies and Data-driven Business*. ACM, 2014, p. 5. URL: <http://dl.acm.org/citation.cfm?id=2638409>.



## CURRICULUM VITÆ

---

### PERSONAL

Name	Steffen Matthias Schnitzer
Date of Birth	February 13, 1987
Place of Birth	Mannheim
Nationality	German

### EDUCATION

Since 06/2014	Technische Universität Darmstadt, Germany Department of Computer Science Doctoral Candidate
09/2011-10/2013	University of Applied Sciences Darmstadt, Germany Joint International Master of Computer Science – Degree: Master of Science
09/2007-05/2011	University of Applied Sciences Mannheim, Germany Computer Science – Degree: Bachelor of Science
08/1997-06/2006	Ursulinen-Gymnasium, Mannheim, Germany Abitur

### PROFESSIONAL EXPERIENCE

Since 06/2014	Technische Universität Darmstadt, Research assistant at the researcher group Knowledge Media at the Multimedia Communications Lab (KOM)
---------------	--

### ACADEMIC EXPERIENCE

Since 06/2014	Researcher funded by DFG (German Research Foundation): “Design und Entwicklung neuer Mechanismen für Crowd- sourcing als neue Form der Arbeitsorganisation im Internet”
2015 – 2018	Supervision of research project funded by BMBF (Federal Ministry of Education and Research) Softwarecampus: <i>TRiCS</i> <i>Task Recommendation in Crowdsourcing Systems</i>

## TEACHING ACTIVITY

- |             |  |
|-------------|--|
| 2015        | Lecture on <i>Net Centric Systems</i> , Teaching Assistant   |
| 2017        | Lecture on <i>Computer Networks and Distributed Systems</i> , Teaching Assistant   |
| 2014 – 2018 | Supervision of 5 groups in Project/Lab<br><i>Multimedia Communications Lab and Project I/II</i>                                  |
| 2014 – 2018 | Supervision of 8 seminar works in KM Seminar<br><i>Current Topics in Web applications, Information Management, and Semantics</i> |
| 2014 – 2018 | Supervision of 4 groups in ATFIR Seminar<br><i>Advanced Topics in Future Internet Research</i>                                   |
| 2014 – 2018 | Supervision of 8 Bachelor's and Master's Theses  |

## HONORS

- |             |  |
|-------------|--|
| 2015 – 2017 | Softwarecampus Grant. Selected as participant and successful graduation in the Softwarecampus program. |
| 05/2014     | Best Master's Thesis Award 2013 from GFFT <i>Gesellschaft zur Förderung des Forschungstransfers</i>    |

*Darmstadt, December 19th, 2018*

---

Steffen Matthias Schnitzer

ERKLÄRUNG LAUT DER PROMOTIONSORDNUNG

---

**§ 8 Abs. 1 lit. c der Promotionsordnung**

Ich versichere hiermit, dass die elektronische Version der vorliegenden Dissertation mit der schriftlichen Version übereinstimmt.

**§ 8 Abs. 1 lit. d der Promotionsordnung**

Ich versichere hiermit, dass von mir zu einem vorherigen Zeitpunkt noch keine Promotion versucht wurde.

**§ 9 Abs. 1 der Promotionsordnung**

Ich versichere hiermit, dass ich die vorliegende Dissertation selbstständig und nur unter Verwendung der in ihr ausdrücklich genannten Hilfen verfasst habe.

**§ 9 Abs. 2 der Promotionsordnung**

Ich versichere hiermit, dass die Arbeit bisher noch nicht zu Prüfungszwecken gedient hat.

*Darmstadt, 19. Dezember 2018*

---

Steffen Matthias Schnitzer



## COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both  $\text{\LaTeX}$  and  $\text{\LyX}$ : <https://bitbucket.org/amiede/classicthesis/>