

Automatic Structured Text Summarization with Concept Maps

Vom Fachbereich Informatik
der Technischen Universität Darmstadt
genehmigte

Dissertation

zur Erlangung des akademischen Grades
Dr.-Ing.

vorgelegt von

Tobias Falke, M.Sc.
geboren in Arnsherg.

Tag der Einreichung: 16. November 2018

Tag der Disputation: 29. Januar 2019

Referenten: Prof. Dr. Iryna Gurevych, Darmstadt
Prof. Dr. Ido Dagan, Ramat Gan

Darmstadt 2019

D17

Please cite this document as

URN: urn:nbn:de:tuda-tuprints-84304

URL: <http://tuprints.ulb.tu-darmstadt.de/8430>

This document is provided by tuprints,
E-Publishing-Service of the TU Darmstadt
<http://tuprints.ulb.tu-darmstadt.de>
mailto: tuprints@ulb.tu-darmstadt.de



This work is published under the following Creative Commons license:
Attribution – Non Commercial – No Derivative Works 4.0 International
<https://creativecommons.org/licenses/by-nc-nd/4.0/>

Abstract

Efficiently exploring a collection of text documents in order to answer a complex question is a challenge that many people face. As abundant information on almost any topic is electronically available nowadays, supporting tools are needed to ensure that people can profit from the information's availability rather than suffer from the information overload. Structured summaries can help in this situation: They can be used to provide a concise overview of the contents of a document collection, they can reveal interesting relationships and they can be used as a navigation structure to further explore the documents.

A concept map, which is a graph representing concepts and their relationships, is a specific form of a structured summary that offers these benefits. However, despite its appealing properties, only a limited amount of research has studied how concept maps can be automatically created to summarize documents. Automating that task is challenging and requires a variety of text processing techniques including information extraction, coreference resolution and summarization. The goal of this thesis is to better understand these challenges and to develop computational models that can address them.

As a first contribution, this thesis lays the necessary ground for comparable research on computational models for concept map-based summarization. We propose a precise definition of the task together with suitable evaluation protocols and carry out experimental comparisons of previously proposed methods. As a result, we point out limitations of existing methods and gaps that have to be closed to successfully create summary concept maps. Towards that end, we also release a new benchmark corpus for the task that has been created with a novel, scalable crowdsourcing strategy.

Furthermore, we propose new techniques for several subtasks of creating summary concept maps. First, we introduce the usage of predicate-argument analysis for the extraction of concept and relation mentions, which greatly simplifies the development of extraction methods. Second, we demonstrate that a predicate-argument analysis tool can be ported from English to German with low effort, indicating that the extraction technique can also be applied to other languages. We further propose to group concept mentions using pairwise classifications and set partitioning, which significantly improves the quality of the created

summary concept maps. We show similar improvements for a new supervised importance estimation model and an optimal subgraph selection procedure. By combining these techniques in a pipeline, we establish a new state-of-the-art for the summarization task. Additionally, we study the use of neural networks to model the summarization problem as a single end-to-end task. While such approaches are not yet competitive with pipeline-based approaches, we report several experiments that illustrate the challenges — mostly related to training data — that currently limit the performance of this technique.

We conclude the thesis by presenting a prototype system that demonstrates the use of automatically generated summary concept maps in practice and by pointing out promising directions for future research on the topic of this thesis.

Zusammenfassung

Textdokumente effizient zu durchsuchen um eine komplexe Frage zu beantworten ist eine Herausforderung, der viele Menschen gegenüberstehen. Da heutzutage zu fast jedem Thema zahlreiche Informationen elektronisch verfügbar sind, sind unterstützende Tools erforderlich, die sicherstellen, dass wir von der Verfügbarkeit der Informationen profitieren anstatt in der Informationsflut unterzugehen. Strukturierte Zusammenfassungen können in dieser Situation helfen: Sie können einen prägnanten Überblick über den Inhalt einer Dokumentensammlung geben, können interessante Beziehungen aufzeigen und können als Navigationsstruktur zur weiteren Erkundung der Dokumente dienen.

Eine Concept Map, ein Graph bestehend aus Konzepten und ihrer Beziehungen, ist eine Form strukturierter Zusammenfassungen die genau diese Vorteile bietet. Trotz ihrer ansprechenden Eigenschaften wurde bisher jedoch nur wenig untersucht, wie Concept Maps automatisch erstellt werden können um Dokumente zusammenzufassen. Die Automatisierung dieser Aufgabe ist herausfordernd und erfordert eine Vielzahl von Sprachverarbeitungstechniken, insbesondere Methoden der Informationsextraktion, der Koreferenzauflösung und der Zusammenfassung. Das Ziel dieser Arbeit ist es, diese Herausforderungen besser zu verstehen und passende Modelle und Algorithmen zu entwickeln.

Zuerst legt diese Arbeit daher den Grundstein für eine vergleichbare Forschung an Methoden für die automatische Textzusammenfassung auf Basis von Concept Maps. Wir führen eine präzise Definition dieses Problems ein, schlagen Evaluierungsprotokolle vor und führen experimentelle Vergleiche existierender Methoden durch. Dabei zeigen sich Einschränkungen bestehender Methoden und noch nicht abgedeckte Teilprobleme des Zusammenfassungsproblems. Zudem veröffentlichen wir ein neues Evaluierungs-Korpus, das mit einer neuartigen, skalierbaren Crowdsourcing-Methode erstellt wurde.

Darüber hinaus schlagen wir neue Techniken für mehrere Teilaufgaben der Erstellung von Concept Maps vor. Zunächst führen wir die Verwendung von Prädikat-Argument-Analyse zur Extraktion von Konzept- und Beziehungserwähnungen ein, was die Entwicklung von Extraktionsmethoden erheblich vereinfacht. Zweitens zeigen wir, dass ein Tool zur Prädikat-Argument-Analyse mit geringem Aufwand von Englisch nach Deutsch por-

tiert werden kann, was unterstreicht, dass diese Extraktionstechnik auch auf andere Sprachen angewendet werden kann. Wir schlagen außerdem vor, Konzepterwähnungen mithilfe paarweiser Klassifizierungen zu partitionieren, wodurch die Qualität der erstellten Zusammenfassungen deutlich verbessert wird. Wir zeigen ähnliche Verbesserungen für ein neues Modell zur Abschätzung der Wichtigkeit von Konzepten und ein optimales Selektionsverfahren für Zusammenfassungs-Teilgraphen. Durch die Kombination dieser Techniken in einer Pipeline erstellen wir zudem das aktuell beste System zur Erstellung von Concept Map-basierten Textzusammenfassungen. Darüber hinaus untersuchen wir die Verwendung neuronaler Netze, um das Zusammenfassungsproblem als ein einziges End-to-End-Problem zu modellieren. Zwar können derartige Ansätze zur Zeit noch nicht mit Pipeline-basierten Ansätzen konkurrieren, wir zeigen jedoch durch mehrere Experimente auf, welche Herausforderungen – die überwiegend im Zusammenhang mit Trainingsdaten stehen – die Leistungsfähigkeit dieser Technik derzeit noch einschränken.

Zum Abschluss der Arbeit stellen wir einen Anwendungsprototyp vor, der die praktische Nutzung von automatisch generierten Concept Maps demonstriert und beschreiben Richtungen für zukünftige Forschung in diesem Bereich.

Acknowledgements

This dissertation would not exist without the support of many people. I would like to thank my advisor Prof. Dr. Iryna Gurevych for giving me the opportunity to pursue a Ph.D. and for supporting me throughout this journey. I am also deeply grateful to my co-advisor Dr. Christian Meyer for the incredible amount of time he spent on reading and improving my publications and on discussing research ideas. I would further like to thank Prof. Ido Dagan for inviting me to his lab in Israel, for offering valuable feedback on my work at various occasions and for agreeing to be a reviewer of this thesis. My appreciation also goes to the DFG, who generously funded this research through the research training group “Adaptive Preparation of Information from Heterogeneous Sources” (AIPHES, GRK 1994/1).

I would like to thank everyone involved in AIPHES for creating an inspiring and collaborative research environment that I appreciated working in. In particular, I am grateful to Andreas, Avinesh, Christopher, Gerold, Markus, Maxime, Teresa and Thomas for numerous interesting discussions that made the lunches enjoyable despite the cafeteria’s food, to Benjamin for his unmatched knowledge of recent papers on any topic, to Ana and Todor for traveling to Darmstadt regularly to enrich our discussions with new perspectives and to Aicha for helping me to convince the others to get decent coffee. Special thanks also go to Gabi, with whom I collaborated in the beginning of my research, leading to my very first publication. And to Christian, Ivan, Daniil, Yevgeniy and many other current and former members of the UKP lab for valuable discussions, advice and ideas. I became friends with many of you and I thank you for all the support throughout the last years.

I have also greatly benefited from staying at Google for several months and I want to thank the team – David, Greg, Ben, Juri, Samantha and all the Alexes and Kevins – for giving me this opportunity and an exciting time in New York.

Finally, I would also like to express my gratitude to my friends and family. Only with them, a project as big as a dissertation is possible. And to Hannah. For everything.

Contents

1	Introduction	1
1.1	Contributions	3
1.2	Publication Record	5
1.3	Thesis Outline	5
2	Background	9
2.1	Exploratory Search in Document Collections	9
2.1.1	Exploratory Search	9
2.1.2	User Behavior and Requirements	12
2.1.3	Structured Text Representations	14
2.2	Concept Maps	16
2.2.1	Origin and Form	16
2.2.2	Applications	18
2.2.3	Manual Creation	21
2.3	NLP Methods Supporting Document Exploration	22
2.3.1	Concept Map Mining	22
2.3.2	Text Summarization	27
2.3.3	Information Extraction	31
2.3.4	Other Methods	33
2.3.5	Exploratory Search Systems	35
2.4	Chapter Summary	36
3	Structured Summarization with Concept Maps	39
3.1	Motivation	39
3.2	Task Definition	41
3.3	Subtasks and Challenges	43
3.3.1	Concept Mention Extraction	43
3.3.2	Concept Mention Grouping	43

3.3.3	Relation Mention Extraction	45
3.3.4	Relation Mention Grouping	46
3.3.5	Concept and Relation Labeling	46
3.3.6	Importance Estimation	47
3.3.7	Concept Map Construction	47
3.3.8	Task-Level Complexity	49
3.4	Relations to Existing Tasks	49
3.4.1	Concept Map Mining	49
3.4.2	Text Summarization	50
3.4.3	Information Extraction	51
3.4.4	Knowledge Graphs	52
3.5	Evaluation	53
3.5.1	Evaluation Methods for Text Summarization	53
3.5.2	Proposed Evaluation Methods for CM-MDS	54
3.6	Chapter Summary	58
4	Creation of Benchmark Corpora	59
4.1	Motivation and Challenges	59
4.2	Automatic Corpus Creation	61
4.2.1	Using Existing Concept Maps	61
4.2.2	Using Existing Concept Annotations	63
4.2.3	Comparison and Limitations	63
4.3	Manual Corpus Creation	65
4.3.1	Importance Annotation via Crowdsourcing	65
4.3.2	Scalable Manual Corpus Creation	68
4.3.3	Corpus Analysis and Experiments	72
4.4	Chapter Summary	76
5	Concept and Relation Extraction	79
5.1	Motivation and Challenges	79
5.2	Extraction with Predicate-Argument Analysis	81
5.2.1	Predicate-Argument Structures	81
5.2.2	Experiments	84
5.3	Predicate-Argument Analysis for German	91
5.3.1	PropS	93
5.3.2	Porting Rules to German	94
5.3.3	Experiments	98
5.4	Chapter Summary	101

6	Pipeline-based Approaches	103
6.1	Motivation and Challenges	103
6.2	Concept Mention Grouping	105
6.2.1	Pairwise Mention Classification	105
6.2.2	Mention Partitioning	107
6.2.3	Experiments	113
6.3	Importance Estimation	117
6.3.1	Modeling Approaches	117
6.3.2	Features	119
6.3.3	Experiments	122
6.4	Concept Map Construction	126
6.4.1	Integer Linear Programming Approach	126
6.4.2	Experiments	128
6.5	Full Pipeline Experiments	130
6.5.1	Pipeline Overview	130
6.5.2	Experimental Setup	132
6.5.3	Results	133
6.5.4	Error Analysis	136
6.5.5	Runtime Complexity and Optimizations	138
6.6	Chapter Summary	140
7	End-to-End Modeling Approaches	141
7.1	Motivation and Challenges	141
7.2	Synthetic Training Corpora	143
7.3	Sequence Transduction Models	145
7.3.1	Graph Linearization	145
7.3.2	Pre-Summarization	146
7.3.3	Experiments	147
7.4	Memory-based Graph Manipulation Models	150
7.4.1	Memory-based Graph Representations	151
7.4.2	Sequence-to-Graph Networks	153
7.4.3	Experiments	159
7.5	End-to-End Experiments	162
7.5.1	Experimental Setup	162
7.5.2	Quantitative Results	162
7.5.3	Qualitative Analysis	165
7.6	Chapter Summary	166

8 Exploratory Search with Concept Maps	169
8.1 Motivation	169
8.2 Exploratory Search System	169
8.3 User Study	172
8.4 Chapter Summary	173
9 Conclusions	175
9.1 Summary of Findings and Contributions	175
9.2 Future Research Directions	178
Index	183
List of Figures	185
List of Tables	187
List of Acronyms	189
Bibliography	191

CHAPTER 1

Introduction

“Getting information off the internet is like taking a drink from a fire hydrant.”

— Mitch Kapor

In the last decades, the way in which information is stored and distributed changed dramatically. The ubiquitous availability of computing devices such as computers and mobile phones and the widespread use of the internet and world wide web, all technologies invented in the second half of the 20th century (Isaacson, 2014), have been driving these changes. While libraries storing large numbers of printed books used to be the guardians of information in the past, large parts of that content are nowadays electronically available and can be used free of charge by anyone with access to the internet.

However, this development also introduced a challenge: The amount of available information on any given topic is typically so large that it is far beyond what a person can process in a reasonable amount of time. For instance, the English version of Wikipedia contained almost 6 million articles in September 2018¹, the Google Books project had digitized over 25 million books by 2015² and the recently leaked Panama Papers consisted of 11.5 million documents³. The total number of pages in the (indexable part of) the internet was estimated to be 4.4 billion⁴ in September 2018. Clearly, the amount of available information is huge and people can easily be drowned in information. This problem is often referred to as *information overload* (Patterson et al., 2001, Keim et al., 2008).

¹5,718,754 articles on September 19 2018, according to <https://en.wikipedia.org/wiki/Wikipedia:Statistics>.

²We could not find more recent statistics on the project. Numbers for 2015 according to <https://www.nytimes.com/2015/10/29/arts/international/google-books-a-complex-and-controversial-experiment.html>.

³According to <https://www.statista.com/statistics/531269/panama-papers-data-leak-size/>.

⁴Estimated on September 19 2018, according to <http://www.worldwidewebsite.com/> using the estimation methodology outlined in van den Bosch et al. (2016).

To cope with information repositories as large as the internet, search engines such as Google or Bing have established themselves as invaluable tools that are used by most people on a daily basis. But, despite their undeniable usefulness, they cannot cover all requirements that arise in information overload scenarios. As a first example, consider a researcher who wants to start working on a new problem. They want to get an overview of the existing research on that problem, identify and compare different strands of research and eventually decide how to focus their own work. To achieve that, rather than retrieving specific research publications by keywords, they need to get an overview of a whole corpus of publications, identify patterns in them and explore different parts of the corpus. Similar requirements exist in other domains, such as journalism, law or intelligence analysis. In such scenarios, which are known as *exploratory search* (Marchionini, 2006), search engines often cannot fulfill all user requirements and additional tools are necessary to enable a user to cope with the amount of information effectively and efficiently.

In this thesis, we study the automatic creation of structured summaries for document collections. In exploratory search scenarios, such a summary can be used to convey the key ideas of documents in an easily consumable way, allowing a user to quickly get an overview of the content of a document collection without much reading. Additionally, due to their structured nature, they can already reveal interesting patterns and relationships in documents that would otherwise need to be manually discovered by the user. To explore different parts of a collection interactively, structured summaries can also serve as a navigation structure. In the past, many types of structures as well as techniques to automatically derive them from natural language text have been proposed. Well-known examples include lists of keyphrases, tables-of-contents, mind maps or document clustering. As we will argue in the thesis, each of these fulfill different user requirements to different extents and are therefore more or less suitable for the exploratory search scenario.

One type of structured text representation that is particularly interesting are so-called *concept maps* (Novak and Gowin, 1984). Concept maps are labeled graphs that represent concepts and their relationships in a visual and concise form. Figure 1.1 shows a small example. When they are used to summarize the content of large document collections, they are a powerful tool to support exploratory search in those documents, as they provide an overview, reveal structure and allow navigation to details.

However, despite their desirable properties with regard to user requirements, the existing research on computational methods to automatically create them from text is limited. Most work on automatic text summarization in the past focused on producing textual summaries rather than structured ones. Outside of the natural language processing (NLP) community, several other researchers explicitly worked on the automatic creation of concept maps, but the amount of work is limited, spread across different communities and uses various evaluation protocols all lacking comparative experiments. No clear state-of-the-art method for the creation of concept maps from text exists.

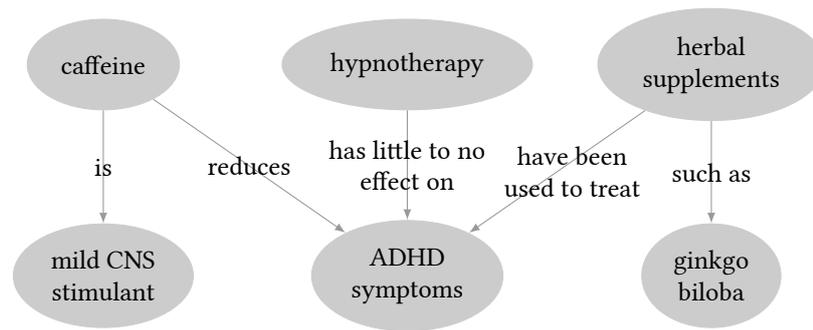


Figure 1.1: An example for a concept map, showing six concepts and relations between them. It was created based on a text discussing alternative treatment options for ADHD.

Motivated by these two facts – the usefulness of concept maps for exploratory search and the limited amount of existing work on their automatic creation – the goal of this thesis is to improve automatic methods that can summarize document collections in the form of concept maps. More specifically, the research presented in this thesis is guided by the following three high-level research questions:

- (1) How good are previously proposed methods to automatically create concept maps in the context of our application scenario?
- (2) How can such methods be improved to create concept maps of higher quality?
- (3) How can research on the task be better aligned, made more comparable and receive more attention in communities such as NLP?

1.1 Contributions

In order to answer the research questions, we present new computational methods and comprehensive experiments resulting in a new state-of-the-art for the automatic creation of summary concept maps from text. In addition, by proposing a clear definition of the task, evaluation protocols and benchmark corpora as well as pointing out open challenges, we hope to inspire and guide future work on this research topic.

In detail, the contributions we make are the following:

- **We standardize the task and consolidate existing work:**
 - We propose a formal definition of the task of summarizing document collections in the form of concept maps on which future work can build upon to develop comparable methods.
 - We introduce a set of automatic and manual evaluation protocols inspired by the work on textual summarization that allow a direct and easy comparison of different computational approaches to the task.

- We develop different annotation methods to create evaluation data for the task and present several corpora created with them.
- As a side product of the previous contribution, we also propose a new strategy to collect importance annotations via crowdsourcing that can be used to create benchmark corpora for different types of summarization.
- We provide a detailed discussion of the subtasks that need to be solved to create summary concept maps, their specific challenges and how previous work approached each of them, resulting in the identification of open challenges.
- We reimplement previously suggested methods and carry out the first experimental comparison between them to identify which perform best.
- **We propose new models to improve performance and close gaps:**
 - We propose to extract concept and relation mentions from text using predicate-argument analysis, which alleviates the effort of manually designing extraction rules while achieving comparable or better extraction performance than previously created rule sets.
 - We perform a case-study of porting a predicate-argument analysis tool from English to German to obtain insights into how challenging it is to make concept and relation extraction approaches available in additional languages.
 - We propose new models for the subtasks of concept mention grouping, concept importance estimation and subgraph selection, three essential steps in creating summary concept maps that have received little attention in previous work, leading to a first pipeline-based approach that can cover all steps of the task and is the current state-of-the-art model for it.
 - In order to model the task as a single end-to-end problem, we propose a set of techniques that allow us to approach it with common sequence transduction models based on neural networks.
 - As an alternative, we propose an end-to-end model that is based on a novel neural architecture that can map text sequences to labeled, directed graphs.
 - We compare both end-to-end approaches in first experiments that evaluate their performance and provide a detailed discussion of remaining challenges.
- **We study the interactive downstream use of summary concept maps:**
 - We implement and present a first prototype application that demonstrates how summary concept maps can support a user while browsing a document collection during exploratory search.

1.2 Publication Record

The majority of the contributions outlined above have been published and presented at peer-reviewed international conferences in the area of NLP. In the following paragraphs, we describe these publications and point out which chapters of this thesis are based on and reuse parts, including verbatim quotes, of these publications.

The concept map-based summarization task is first proposed in Falke and Gurevych (2017a). Further, that publication also introduces evaluation metrics, a new benchmark corpus together with the novel crowdsourcing technique as well as a first baseline for the task. These contents are used partly in Chapter 3, in particular in Section 3.2 and Section 3.5.2, and to a large extent in Section 4.3 of Chapter 4.

The publication by Falke and Gurevych (2017c) focuses on concept and relation extraction approaches. It reviews, reimplements and evaluates rule-based extraction techniques proposed in previous work. In addition, it introduces our proposal of using predicate-argument analysis tools for the extraction and includes them in the experimental comparison. In Section 5.2 of this thesis, we incorporate the content of that publication. The case study of porting a predicate-argument analysis tool from English to German has been published in Falke et al. (2016). Section 5.3 of this thesis is based on it.

Moreover, we published our improved methods for concept mention grouping, importance estimation and subgraph selection in Falke et al. (2017). The paper further contains a description of the current state-of-the-art pipeline system that covers all steps of the task and experimental comparisons against a range of techniques proposed in previous work. The content of that paper, together with additional details and intermediate experimental results, is the basis for Chapter 6 of this thesis.

And finally, the publication by Falke and Gurevych (2017b) presents an interactive document exploration system based on concept maps. The publication focuses on the use of the system for experimental comparisons of structured text representations in user studies. Chapter 8 briefly describes the application. An additional joint publication with other researchers, Zopf et al. (2018a), investigates the usefulness of different linguistic annotations for the identification of summary-worthy content. The techniques for concept extraction and grouping described in Chapter 6 are an annotation that this work contributed to the joint project. However, the content of that publication is not part of this thesis.

1.3 Thesis Outline

The structure of the remainder of this thesis is as follows: We first provide the necessary background on both user requirements and existing previous work in Chapter 2. In Chapter 3, we introduce the central task of the thesis followed by corresponding evaluation corpora in Chapter 4. Chapters 5, 6 and 7 then focus on different subtasks as well as pipeline-

based and end-to-end approaches to the task. In Chapter 8, we briefly look at an application scenario for concept maps and close with a summary and outlook in Chapter 9.

We start Chapter 2 by defining the application scenario that motivates this research and illustrate it with several practical examples. Further, we review user studies that have been carried out in this setting and derive a set of user requirements from it. We then compare a range of text representation tools against these requirements to determine how well they can support a user during exploratory search. Furthermore, we present one representation with many desired properties – concept maps – in detail, discussing its origin, advantages and applications in practice. In the final part of the chapter, we give an overview of existing approaches to automatically create concept maps from natural language text as well as a range of other NLP methods that attempt to support users during document exploration.

In Chapter 3, we formally define the central problem studied in this thesis, the automatic creation of multi-document summaries in the form of concept maps. First, we motivate the task based on the review of existing work and user requirements laid out in the previous chapter. A formal definition of the task, a discussion of its challenges for computational models and a comparison to existing tasks follows. We close the chapter by suggesting several methods to evaluate and compare automatic methods for the task.

In Chapter 4, we look at the data that is needed to train and evaluate computational methods for the newly proposed summarization task. We discuss requirements for suitable corpora and show that corresponding data does not yet exist. Therefore, we describe two different strategies to collect data – by automatically extending partial annotations and by creating annotations from scratch with scalable methods – and present the corpora that we obtained using these strategies.

In Chapter 5, we focus on the subtasks of concept and relation mention extraction. Using the datasets introduced in the previous chapter, we will present a series of experiments that, for the first time, directly compare different extraction approaches proposed in previous work. Moreover, we will introduce the idea of using predicate-argument analysis for concept and relation extraction and include such methods in the experimental comparison. And finally, as most work on concept maps in the past has focused on the English language, we will dedicate the second part of the chapter to studying how such extraction methods can be ported to other languages.

In Chapter 6, we focus on the remaining subtasks, namely mention grouping, importance estimation and concept map construction. We first study each subtask in isolation and propose new techniques to address its challenges. In the final part of the chapter, we then combine techniques for all subtasks into a pipeline and evaluate its overall task performance, the quality of the generated concept maps and the scalability of the pipeline.

In Chapter 7, we focus on alternative models that try to approach the task end-to-end rather than with a pipeline of multiple steps. For various tasks in NLP, such approaches have recently been very successful. We first discuss how sequence-to-sequence models

can be applied. Then, we propose an alternative architecture which we call a sequence-to-graph network. We evaluate both approaches experimentally to assess the applicability of end-to-end modeling for the task.

In Chapter 8, we take a look at potential applications of the technology developed in this thesis in exploratory search scenarios. Specifically, we present a corresponding prototype application and report results from a first user study.

The final Chapter 9 summarizes the findings of the thesis and outlines promising directions for future research on structured summarization with concept maps.

CHAPTER 2

Background

In this chapter, we introduce relevant related work on exploratory search in document collections. First, we define the application scenario that motivates this research and illustrate it with several practical examples. Second, we review user studies that have been carried out in this setting and derive a set of user requirements from them. We then compare a range of text representation tools against these requirements to determine how well they can support a user during exploratory search.

Furthermore, we present one representation with many desired properties — concept maps — in detail, discussing its origin, advantages and applications in practice. In the final part of the chapter, we give an overview of existing approaches to automatically create concept maps from natural language text as well as a range of other NLP methods that attempt to support users during document exploration.

2.1 Exploratory Search in Document Collections

2.1.1 Exploratory Search

Exploratory search is a common scenario faced by many people. It refers to information seeking activities that go beyond the lookup of facts. Typical examples are activities aiming at extending one’s knowledge about a topic, comparing or aggregating data or concepts, gaining new insights and discovering conceptual boundaries (Marchionini, 2006).

While it is difficult to find a succinct and comprehensive definition of exploratory search in the literature, White and Roth (2009) note that it can be characterized either by the nature of a searcher’s goal or the process they use to reach that goal. The goal is usually complex, open-ended and multi-faceted. Users aim to “develop enhanced mental capacities” (White and Roth, 2009). Often, the information need cannot be clearly stated. Process-wise, exploratory search tasks require “opportunistic, iterative, multi-tactical” (White and Roth,

2009) strategies, including “selection, navigation and trial-and-error tactics” (Marchionini, 2006). In contrast, lookup tasks, such as fact finding or question answering, can typically be handled with one or several keyword queries to a traditional search engine (Marchionini, 2006). Athukorala et al. (2016) empirically compare the search behavior of users with exploratory and lookup tasks and observe differences in query length, scrolling depth and task completion time. Given the different kinds of search behavior, they argue that more tailored and adaptive search tools should be offered for this type of search.

For the purpose of this thesis, we are interested in exploratory search in a collection of textual documents and define it, in line with previous work, as follows:

Definition 1: Exploratory Search in Document Collections

Exploratory search in a document collection is an information seeking activity with a complex goal requiring the combination and synthesis of data from multiple sources and multi-tactical search behavior to find relevant data in the documents.

There are many practical use cases in which people process large collections of textual documents with a complex goal, some of which have been the subject of user studies and other research. The following is a selection:

- Intelligence analysts regularly work with large sets of documents and process them to assess threats and recommend actions. Several studies have been conducted to understand their search behavior (Chin et al., 2009, Pirolli and Card, 2005).
- Investigative journalists process large document collections, such as those released by WikiLeaks, in order to find newsworthy stories, which often requires to find connections between facts across documents (Yimam et al., 2016, Kirkpatrick, 2015).
- Researchers have to monitor and read vast amounts of published scientific papers to stay up to date. They try to find connections, differences and trends within that content to guide their research (Jackson et al., 2016, Lee et al., 2005).
- Lawyers work with a wide range of legal documents such as legislation, case reports and legal comments on a daily basis. They need to find relevant documents for a case and process them to derive arguments and conclusions (van Noortwijk, 2017).

Clearly, the more documents one has to work with, the more difficult this search task becomes due to information overload. Patterson et al. (2001) conducted a study on information overload in which 10 analysts had to prepare a report on the causes of the Ariane 501 failure. The subjects were given 2000 documents, could use a simple keyword search and worked against a time limit. A detailed analysis of the reports revealed that, although being professionally trained analysts, all subjects missed relevant information and some even included incorrect statements in their reports.

In order to cope with the information overload problem in such scenarios, a wide range of supporting software has been proposed and developed. In this thesis, we refer to such support systems as *exploratory search systems*:

Definition 2: Exploratory Search System

An exploratory search system is a computer application with the goal to help a user perform an exploratory search task in a document collection more efficiently.

The notion of exploratory search systems as defined above is very broad, covering many applications that aim to support different parts of exploratory search with a variety of techniques and successes. We review some of them in Section 2.3. In order to further narrow down the goal of this thesis, we look at exploratory search in more detail.

The cognitive process of working with data to arrive at a result, e.g. a recommended action, is known as *sense-making*. Based on their work with intelligence analysts, Pirolli and Card (2005) define a prototypical sense-making process that spans four steps:

- (1) gathering information
- (2) structuring the information
- (3) manipulating it to gain insights
- (4) creating the final product (hypothesis, conclusion or action)

They also describe a more detailed version of this process with 16 different steps and artifacts produced along them. Going through this process, the amount of data an analyst works with decreases due to filtering and aggregation activities, while the performed activities become more demanding, requiring the analyst to synthesize and reason with the data. Given this typical sense-making process, we argue that the biggest potential to support people lies in the first steps. While automatic computational methods can easily handle large amounts of documents, freeing a user from the overload problem and the tedious processing of all documents, the cognitively more challenging tasks later in the process can be left to the user, as they would require higher-level natural language understanding capabilities that cannot yet be automated (Marcus, 2018, Pearl, 2018, Battaglia et al., 2018).

The first steps in the sense-making process focus on gathering information and structuring it to answer *Who & What?* and *How are they related?* questions (Pirolli and Card, 2005). Therefore, *structured text representations* which already reveal some structures of interest can be helpful and can simplify this part of the process for a user.

Definition 3: Structured Text Representations

A structured text representation presents (parts of) the content of a document collection in an alternative form that reveals structures expressed in the documents. It may be used instead of or in combination with the original representation.

Given the defined terms, we can state the research goal of this thesis and its practical usage scenario more clearly: We aim to develop automatic methods that derive *structured text representations* from document collections such that they can be used in *exploratory search systems* to support users during *exploratory search* in the collection.

2.1.2 User Behavior and Requirements

In this section, we review the findings of a range of user studies to understand how a useful text representation should look like. Although a plethora on text structuring algorithms and exploratory search systems have been suggested in the literature (see Section 2.3), for only a subset of these experiments have been conducted to verify if the proposed approach is helpful. In those cases, the common setup is to compare the approach against a set of simpler baselines to prove its added benefit. As a result, many systems and representations have been shown to be useful in the sense of beating baselines, but which of them is most helpful is largely unknown due to the lack of direct comparisons.

We circumvent this issue here by looking primarily at studies that observed how users approach exploratory search *naturally*, i.e. when having none or only simple tools supporting them, and derive requirements from these observations. An experiment conducted by Loizides and Buchanan (2009) looked at how users judge the relevance of scientific papers and which parts of the documents they look at. Chin et al. (2009) observed professional intelligence analysts during a staged threat assessment task, studying their information seeking tactics. Kang et al. (2011) carried out an evaluation of an exploratory search system with university students, including a control group using only pen and paper for which detailed observations are reported. Yimam et al. (2016) report requirements of journalists gathered via structured interviews.

Based on the observed user behavior, we derive the following set of requirements:

R1: Key Units *The representation should clearly reveal the key units of information discussed in the document collection.* A common activity observed in all studies is that users try to identify key elements such as facts, events, places, persons and organizations (Chin et al., 2009, Kang et al., 2011, Yimam et al., 2016). A text representation that already identifies and extracts these units from the original text can help a user, in particular, if the number of documents is large and reading all of them would require a lot of time.

R2: Relations *The representation should make it easy to understand important relationships between the key elements.* In all studies, in accordance with the sense-making process (Pirolli and Card, 2005), users spend a considerable amount of time on identifying relationships between key elements to understand the content. The subjects captured these relations mostly by drawing graphs or networks, and sometimes also maps or timelines (Chin et al., 2009, Kang et al., 2011). Likewise, journalists also report to be particularly interested in connections (Yimam et al., 2016). If a structured representation already shows these relations directly, a user has to spend less time on reading in order to discover them.

R3: Overview *The representation should provide an aggregated view of the documents such that a user can easily get an overview of its contents.* To avoid that users have to process all documents in a collection completely, even if they are irrelevant, which is the common behavior when using no supporting tools (Chin et al., 2009, Kang et al., 2011), a focused representation can provide an overview by leaving out irrelevant parts and aggregating redundant and related information. Multi-document summarization systems, which attempt to do this producing a non-structured representation, were found to be helpful during exploratory search (McKeown et al., 2005, Maña-López et al., 2004, Roussinov and Chen, 2001). We note that traditional information retrieval systems cannot provide an overview of what is in a collection, but only retrieve documents for a specified query. In exploratory search, queries are difficult to define due to complex information needs (Marchionini, 2006).

R4: Detail *The representation should allow a user to retrieve more detailed information for any element provided in the overview.* While the previous requirement is reasonable to enable efficient handling of large document collections, an aggregated view always has to leave out detail information that might also be relevant. Kang et al. (2011) observed that “overview first, filter and selection, and elaborate on details” was the most common search strategy among their study participants. It is also a common guideline to design information visualization tools (Shneiderman, 1996). A good structured representation should make it easy to navigate from the overview to details and vice versa.

R5: Conciseness *The representation should be as concise as possible such that a user can process it quickly and without much effort.* Loizides and Buchanan (2009) observed that their subjects mainly focused on parts of the papers that can be easily processed, such as the title, headings, pictures and the paper’s abstract, reading only small or no parts of the full text. When searching in a document collection, many subjects added highlights or took notes to represent relevant information more concisely (Chin et al., 2009, Kang et al., 2011). A succinct representation allows a user to process information faster and is thus beneficial.

R6: Intuitiveness *The representation should be easy to understand and should not require specific training or experience to use it.* In order to be broadly applicable and accepted by users, it should be intuitive to understand. As a negative example, clustering techniques, both traditional document clustering (Sanderson and Lawrie, 2000) and topic models (Boyd-Graber et al., 2017), have been criticized to sometimes yield hard to interpret clusters that are more confusing than helpful for users. A good representation should avoid this.

While we are confident that these collected requirements are reasonable and supported by empirical evidence, they are not necessarily complete. As we show in the next section, they are useful to characterize differences between common text representation formats. However, when trying to determine the best representation for a specific practical application, additional requirements might need to be considered.

2.1.3 Structured Text Representations

Given the requirements, we can use them to compare popular structured text representations. Table 2.1 summarizes this qualitative analysis. We include the original full text of a document collection as well as textual summaries as reference points in the comparison, although they are not covered by our definition of structured text representations, lacking explicit structure in the representation.

Full Text Apart from being intuitive to use (R6), full documents in their original form meet none of the other requirements, since they neither explicitly show structure (R1, R2) nor represent the content concisely (R5) or provide an overview (R3). They do provide all details (R4), but no means to quickly navigate to the details of a specific aspect.

Textual Summary Multi-document summaries are representations of the content of a document collection of limited size, containing only the key information (Nenkova and McKeown, 2011). They are intuitive to use (R6) and provide an overview (R3). However, such a shorter text is still not very concise (R5) as it still requires a user to parse and understand potentially long sentences. While it can be easier to find key units (R1) in the summaries, they do not explicitly show relations (R2). No links between the summary's and document's content allow to quickly navigate to details (R4).

Keyphrases Keyphrases are words or short phrases assigned to documents in order to indicate their topic and index them for searching (Hasan and Ng, 2014, Gutwin et al., 1999). The phrases can represent the key units of content (R1) and provide an overview (R3) in a concise way (R5). The idea of keyphrases is intuitive and commonly known (R6). By linking them with their mentions in the documents, access to a limited amount of details can be provided (R4). However, with keyphrases alone, no relations can be represented (R2).

Representation	R1 Key Units	R2 Relations	R3 Overview	R4 Detail	R5 Concise	R6 Intuitive
Full Text(s)	-	-	-	+	-	+
Textual Summary	o	-	+	-	o	+
Keyphrases	+	-	+	o	+	+
Table-of-contents	o	o	+	+	+	+
Mind Map	+	o	+	+	+	+
Concept Map	+	+	+	+	+	+
Formal Map	+	+	+	+	o	-
Labeled Cluster	o	o	+	o	+	-

Table 2.1: Common text representations compared by user requirements. The symbols in each cell denote full (+), partial (o) or no (-) support of the requirement. See text for explanations.

Table-of-contents A table-of-contents as included in almost every book shows the content of a document by arranging headlines in a hierarchical structure. Typically, they only exist for single documents. Tables-of-contents are intuitive to use (R6) and their main purpose is to provide an overview (R3) in a concise manner (R5) and allow navigation to details (R4). But they only partially satisfy R1 and R2 as they always show topics and their hierarchical relations, which are different from the units and relations most people were interested in as observed in the user studies.

Mind Map Mind maps (Buzan, 1984, 2002) are graphs with concepts as labeled nodes and unlabeled edges indicating relations between them. They extend tables-of-contents in the sense that relations do not have to be hierarchical, concepts are shown instead of headlines and multiple documents can be covered. They therefore satisfy similar requirements, but are also able to represent key units of arbitrary nature as concepts (R1). Their capability to represent relations (R2) is still limited, as only one type of relation can be shown.

Concept Map A concept map (Novak and Gowin, 1984) extends the idea of a mind map by adding labels to edges. Using different labels, many types of relations can be represented in a single map (R2). The representation is concise (R5), easy to understand (R6) and it can be used to show a limited number of key units (R1) to provide an overview (R3). Similar to keyphrases and mind maps, concepts (and also relations) can be linked to mentions in the document collection to facilitate access to details (R4).

Formal Map Several other graph-based representations that show concepts and relations exist, including conceptual graphs (Sowa, 1984), topic maps (Parker, 2003) or ontologies

based on semantic web standards such as Resource Description Framework (RDF) and Web Ontology Language (OWL) (Maedche, 2002, Breitman et al., 2007). The main difference between them and concept maps is that all these representations have formally defined syntax and semantics, which makes them machine-readable but less useful and intuitive for humans not familiar with the syntax (R6).

Labeled Cluster Document clustering has been used repeatedly to facilitate browsing in document collections, using for instance agglomerative clustering (Cutting et al., 1992) or hierarchical topic models (Smith et al., 2014). It is well suited to provide an overview of a collection (R3) and, when clusters have keyphrase-like labels, it is also a concise representation (R5) that can show key units (R1). However, relations beyond the topic-relatedness of documents cannot be represented (R2) and access to fine-grained details is difficult (R4), as most methods operate on the document level. And in addition, as mentioned earlier, several studies have found users to have problems interpreting clusters (R6).

The comparison showed that concept maps, although less known than other structured text representations, have several desirable properties and extend more common representations in useful ways. From a user requirements' point-of-view, it seems to be very promising to study them as representations to support document exploration.

2.2 Concept Maps

2.2.1 Origin and Form

Concept maps have their origins in the area of learning psychology. They were invented in the 1970s as part of Joseph Novak's research at Cornell University where he and his team studied how children understand science concepts and how that understanding changes over time (Novak and Gowin, 1984). In order to document and visualize the conceptual understanding that a student has about a certain topic, they developed concept maps. Their research program was based on David Ausubel's cognitive psychology (Ausubel, 1968) whose fundamental idea is that a learner's existing knowledge is organized into concepts and propositions and that learning happens by assimilating new concepts and propositions into that framework. Consequently, they created concept maps as a knowledge representation formalism that closely resembles this cognitive structure (Novak and Cañas, 2007).

Figure 2.1 shows a concept map created by Novak and Cañas (2007) that describes concept maps themselves. A concept map is a labeled graph with nodes and edges.

Every node represents a *concept*, which is defined as a “perceived regularity in events or objects, or records of events or objects, designated by a label” (Novak and Cañas, 2008). This notion of a concept is very broad and encompasses classes of persons, objects, ideas

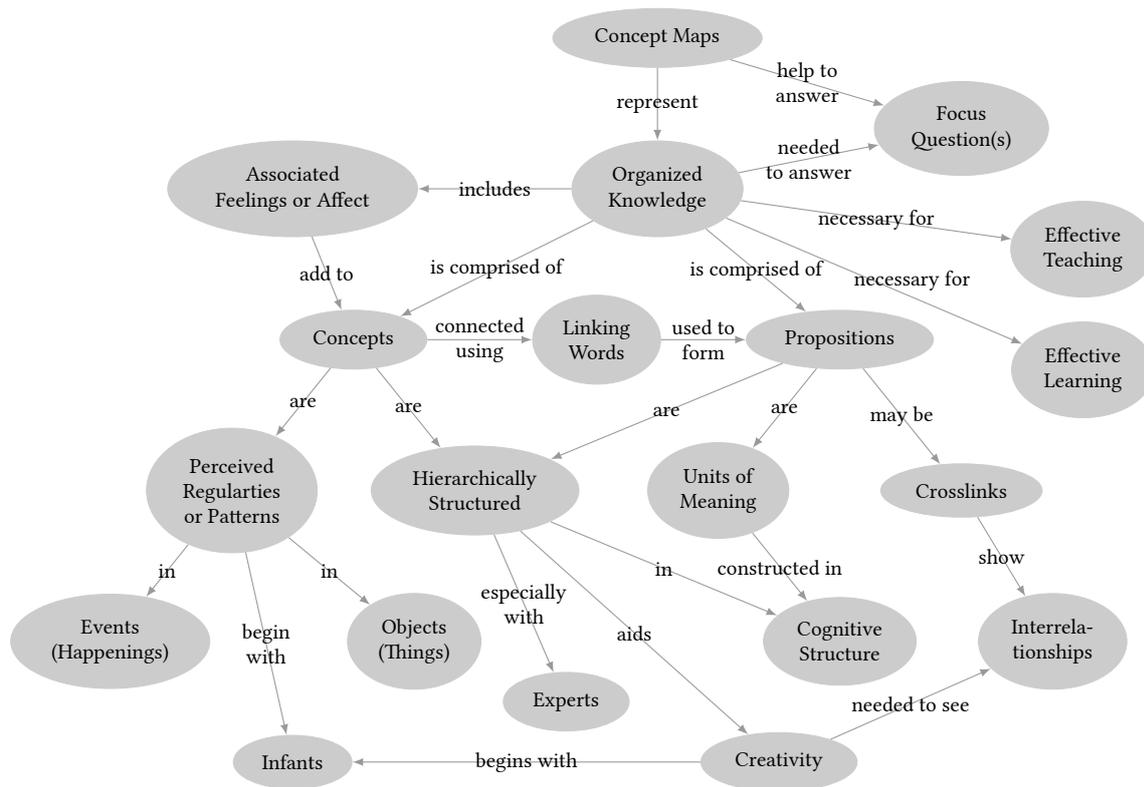


Figure 2.1: A concept map created by its inventor Joseph Novak that describes the idea of concept maps (own visualization based on Novak and Cañas (2007)).

and activities at arbitrary levels of abstraction. It also includes, as the authors explicitly mention, concepts at the most specific level, such as a particular person, if including it is helpful. Note that this notion of a concept is broader than what is usually studied as (named) entities in NLP. It is therefore possible to represent a wide range of different types of information in a concept map.

Concepts are connected with edges representing *relations*. The relation labels should describe the relationship between the connected concepts such that the triple of a concept, relation and concept forms a meaningful *proposition*. In Figure 2.1, (*concept maps - represent - organized knowledge*) is an example for a meaningful proposition. The labeled relations are a key characteristic of concept maps that distinguish them from other representations such as mind maps. Since concepts can be interpreted and understood in the context of their relationships to other concepts, concept maps are a powerful tool for learning and knowledge representation in general.

To be a concept map, a graph of concepts and relations should satisfy the following properties that Cañas et al. (2005) list as features of concept maps:

Open Vocabulary: Labels for concepts and relations can be chosen freely by the author and can be any sequence of words. There is no predefined set of valid labels.

Diverse Relations: The meaning of a relation is only defined by its label and there is no limit to the types of relationships that can be represented in a single concept map.

Succinct Labels: Labels should be as short as possible while being descriptive enough to be meaningful for a reader of the concept map.

Well-formed Propositions: Every triple of a concept, relation and concept should form a meaningful statement and should not rely on other relations to make sense.

Hierarchical Organization: A concept map should be structured such that the most inclusive concepts are at the top and more specific concepts organized beneath them.

Hoffman et al. (2005) introduced the term *propositional coherence* to describe a concept map in which all propositions are well-formed, which is usually desired.⁵ The activity of creating and working with a concept map is known as *concept mapping*.

Several software packages can be used to manually create and use concept maps, among which *CmapTools*⁶ is arguably the most popular one. It has been developed at the Institute for Human and Machine Cognition in Florida by a team led by Novak and Cañas and supports, in addition to the creation of concept maps, to also enrich them with images, video content and web links. Cross-links between different concept maps can be created and maps can be embedded into each other to create large maps spanning topics and subtopics. The tool also features a web-based platform to store, share and collaboratively work on maps with others over the internet (Cañas et al., 2005, Novak and Cañas, 2006).

Another important part of the concept mapping ecosystem is the biennial International Conference on Concept Mapping (CMC) started in 2004. It features both papers on recent research on concept maps and experiences from teachers applying concept maps during their teaching activities. To a limited extent, computational approaches to concept mapping have also been presented in this venue, which we will review in Section 2.3.1.

2.2.2 Applications

This section gives a broad, but not necessarily exhaustive, overview of applications of concept maps that have been reported in the literature. Their first and most extensively studied applications have been, due to their origin, in the area of education. Other applications include library access, ontology creation, expert training or web search.

Nesbit and Adesope (2006) performed an extensive meta-analysis on the effects of using concept maps during teaching for which they selected 55 high-quality studies with a total

⁵Note that the example in Figure 2.1, although regularly used in introductions of concept maps, violates this principle. The triple (*perceived regularities or patterns - in - events (happenings)*) (bottom left), among others, depends on (*concepts - are - perceived regularities or patterns*) to be meaningful.

⁶<https://cmap.ihmc.us/>

of 5,818 participants. In 25 studies, students created new or modified existing concept maps. With regard to knowledge retention and transfer, these activities were found to be more effective than reading texts, attending lectures or classroom discussions and slightly more effective than constructive activities such as writing summaries or outlines. These findings were consistent across a broad range of educational levels, subject areas and experimental settings. In the other 30 of the analyzed studies, students merely studied provided concept maps rather than constructing them. Also in this setting, concept maps were found to be more effective for knowledge retention than studying text passages, lists or outlines. For both settings, the researchers point out that while the observed findings are significant, more and larger-sized studies are needed to better understand the effects and the conditions necessary to observe strong benefits. With regard to reasons for concept maps' effectiveness, they point out that the empirical findings are "consistent with theories that concept maps lower extrinsic cognitive load by arranging nodes in two-dimensional space to represent relatedness, consolidating all references to a concept in a single symbol, and explicitly labeling links to identify relationships" (Nesbit and Adesope, 2006).

An alternative application of concept maps in education is as a testing tool. Edwards and Fraser (1983) performed an early experiment with 24 ninth-grade students in which they assessed the student's science knowledge by letting them write reports or create concept maps on a given topic. They compared the assessments to the results of interviews, a technique which is known to be most accurate (but time-consuming) to evaluate a student's understanding of science concepts. They found that the reports mostly underestimated the students' understanding as determined by the interviews, because students gave incorrect, incomplete or ambiguous written answers. Using concept maps, they were able to more clearly express their understanding and the results aligned better with the interviews.

Later work by McClure et al. (1999) further strengthens the argument to use concept mapping as an assessment tool in schools. The authors conclude that scoring concept maps created by students yields reliable evaluation scores and that the required effort, consisting of training the students in concept mapping, having them create concept maps on the test topic and letting a teacher score the maps, is comparable to other testing methods. However, they observed that the reliability of the scores depends on the technique used to score the student maps. In addition to the examples outlined here, many more studies have been conducted on using concept maps for educational purposes, including most papers presented at the aforementioned biennial International Conference on Concept Mapping.

Besides the educational domain, concept maps have been regularly used to structure information repositories and provide means of easy access and navigation to users. Carnot et al. (2001) conducted a first study that compared the performance of 62 students who were given questions on developmental psychology. All students had access to the contents of an introductory book chapter on that topic, which was provided either as a concept map, as a simple text with hyperlinks covering the same content or as a multimedia-enriched

and more verbose web page. Concept map users answered significantly more questions correctly, leading the authors to conclude that concept maps successfully support users that try to navigate and find information. The group using the reduced text interface did not perform better than the one using the more verbose web pages, indicating that it is not only the reduction of content but also the form of the concept map that is helpful. A similar study by Valerio et al. (2012) observed large improvements in response time at only small drops in accuracy when answering questions given a concept map instead of the source text, even when the map had been automatically generated from the text.

Practical applications have been reported by Hoffman et al. (2001), who created concept maps to represent expert knowledge about weather forecasting and to provide access to a repository of learning materials for their employees, and by Briggs et al. (2004), who used concept maps to provide access to a large multimedia repository explaining the NASA's activities to explore Mars. The Mars concept maps were made available online and the authors report a large interest from the public. Gaines and Shaw (1994) report using concept maps, in addition to other knowledge representation techniques, to collaboratively capture shared knowledge in large research projects. The work by Shen et al. (2003) and Richardson and Fox (2005) proposes to use concept maps to provide access to library contents and describes their ongoing efforts. They argue that concept maps created for books or book chapters would be easier for a user to consume than an abstract and that they can be used to effectively provide an overview and summarize contents. They also express their desire to use automatic methods to create the concept maps and later report on first steps that they have taken in that direction (Richardson and Fox, 2007).

Carvalho et al. (2001) use a concept map as the context for a traditional web search and develop algorithms that rerank retrieved web pages based on the content and structure of the concept map. Lee (2004) also uses concept maps in the area of search engines and presents a system that lets a user organize queries and their results in a concept map that is constructed throughout the search session. However, what they call a concept map is very different from a Novakian concept map. Leake et al. (Leake et al., 2003, 2004, Cañas et al., 2004) propose to use search engines to support users creating a concept map. They develop several algorithms that construct queries from the content of a partial concept map in order to retrieve documents that help the user find additional concepts and relations for the map.

Another application domain is writing support, which Villalon et al. study in their research (Villalon and Calvo, 2008, 2009, Villalon, 2012). Their idea is that concept maps constructed from student essays can be a valuable tool for the students to improve their writing, as the maps provide a visualization of the content and structure of the essay. Towards that goal, they develop algorithms to automatically create concept maps from student essays, annotate a corpus for the task, propose evaluation metrics and integrate their methods into larger writing support systems. We will revisit the different parts of their work in detail in later chapters of this thesis.

Due to their similarity with formal knowledge representations, automatic methods to create concept maps from text have also been applied to create domain ontologies. For that purpose, algorithms as presented in Section 2.3.1 can be combined with additional filtering and conversion steps, partly automating a laborious process that is usually done manually by domain experts (Zouaq and Nkambou, 2008, 2009, Zouaq et al., 2011).

2.2.3 Manual Creation

Since this thesis investigates the automatic creation of concept maps, a look at how humans perform this task can give valuable insights. The following procedure is the established best-practice that is recommended by Novak and Cañas (2007, 2008):

- (1) Define the topic of the map, ideally as a focus question that should be answered.
- (2) Identify key concepts for the topic. A set of 15 to 25 concepts is usually sufficient.
- (3) Order the set of concepts (approximately) from most general to most specific. The concept map will be build from this list known as the *parking lot*.
- (4) Create a preliminary map by adding concepts from the parking lot and adding relations between them. Make use of the ordering to ensure the map is hierarchical.
- (5) Iteratively revise the map by
 - adding concepts still available in the parking lot,
 - adding non-hierarchical relations, known as *cross-links*,
 - adjusting the layout to make the growing map easier to read.

Edwards and Fraser (1983) report giving similar instructions to the participants of their experiment. Novak and Cañas (2007) note that it often happens that some concepts turn out to be difficult to connect to the rest and therefore remain in the parking lot. They also point out that good maps need many revisions and are rarely created right away.

Villalon et al. (2010) describe an annotation study of concept maps that gives further insights into how humans create concept maps. Their work is different from the general, unrestricted concept mapping use case assumed by the instructions above in the sense that they require users to create a map that precisely reflects the content of a given text. This setup is particularly relevant for this thesis, as it is this setup that a computational approach to create concept maps from text faces.

In their experiment, two annotators created concept maps for 42 student essays with an average length of 468 words. The annotators used an annotation tool that forced them to choose concept and relation labels extractively from the given essay and that only allowed them to add relations if both concepts and the relation label occurred within the same paragraph. In this restrictive setup, the authors observed high agreements of around 80%.⁷

⁷The authors measure the fraction of overlapping concepts between concept maps from two annotators and a second metric computing the relation overlap of relations connecting shared concepts. The observed agree-

The created concept maps have on average 21 concepts and 12 relations and it takes an annotator 27 minutes to create one. Based on a constituency parse of the text, the authors find that 81% of the text passages selected to label concepts are noun phrases, followed by adjectival phrases (8%) and verb phrases (7%). This shows that while noun constructions are by far the most dominant linguistic category for concept labels, the notion of a concept is broader and not limited to nouns. For relations, 47% of the selected labels are verb phrases, 21% noun phrases and 8% adjectival phrases, showing an even more diverse distribution among phrase categories. Further observations are that the frequency with which selected concepts occur in the source text follows Zipf’s law, such that a few concepts occur frequently but most only once or twice. For 92% of the propositions in the created concept maps, both concepts and the relation have labels taken from a single sentence.

2.3 NLP Methods Supporting Document Exploration

While the previous sections focused on exploratory search and structured text representations from a user’s point-of-view, we now look at existing computational methods to create such structures. By automating this process, it becomes possible to handle large document collections. Given our focus on concept maps, we first review automatic methods to create them, and then move to the related tasks of summarization and information extraction. We also give a brief overview of computational methods pursuing other directions, e.g. different representations, and finish presenting existing exploratory search systems that integrate these computational methods into end user applications.

2.3.1 Concept Map Mining

The automatic creation of concept maps from an unstructured text has been studied in several areas⁸ and is often referred to as *concept map mining*. Different techniques towards that goal have been suggested for single documents (Oliveira et al., 2001, Valerio and Leake, 2006, Villalon and Calvo, 2009, Kowata et al., 2010, Aguiar et al., 2016) and for sets of documents (Rajaraman and Tan, 2002, Zouaq and Nkambou, 2008, Zubrinic et al., 2012, Qasim et al., 2013), spanning a broad range of text genres including scientific papers (Qasim et al., 2013), legal documents (Zubrinic et al., 2012), news articles (Kowata et al., 2010), student essays (Villalon and Calvo, 2009) and general web pages (Rajaraman and Tan, 2002). Most of that work focuses on processing English texts, with notable exceptions that target Portuguese (Kowata et al., 2010) and Croatian (Zubrinic et al., 2012).

ment is 62% and 27% for the first annotation run and 77% and 85% after refining the guidelines and the tool, introducing the described extractiveness requirements (Villalon, 2012).

⁸Being published in different communities, the work follows various scientific standards and practices. As a result, some papers do not provide the level of detail and experimental rigorousness common in the NLP community nowadays, which makes it hard to compare and reproduce such work.

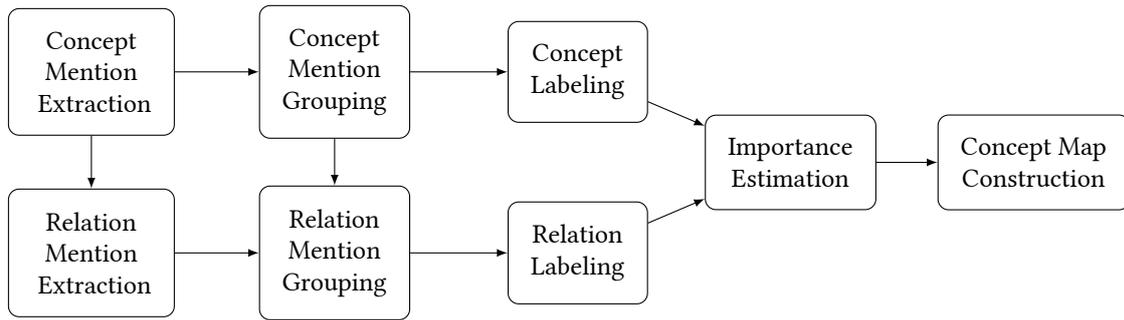


Figure 2.2: Subtasks of concept map mining and their dependencies.

Typically, computational methods approach the task with a pipeline of several steps that turn the input text(s) into a concept map. Within this thesis, we will use the comprehensive list of subtasks depicted in Figure 2.2. It subsumes most other suggested lists, e.g. by Villalon and Calvo (2008), and provides a framework to structure and compare proposed techniques. *Concept* and *relation mention extraction* refer to the tasks of identifying spans in the input documents that describe concepts and relations between them, while the subtask of *mention grouping* deals with determining which of the extracted mentions refer to the same concept or relation. The subsequent steps of *concept* and *relation labeling* and *importance estimation* assign labels to concepts and relations and determine how relevant these elements are. Finally, a concept map is *constructed* from (a subset of) them. In Section 3.3, we discuss these subtasks and their challenges in detail. Some methods also establish a hierarchical organization of the concepts to satisfy Novak’s hierarchy requirement (see Section 2.2.1), but since this is strongly connected to the visual layouting of the concept map, it is usually seen to be out of scope of the concept map mining task.

Existing work on concept map mining used a variety of evaluation protocols to study the effectiveness of their proposed methods, ranging from qualitative expert judgments (Kowata et al., 2010, Zubrinic et al., 2015, Qasim et al., 2013) to automatic comparisons against manual annotations (Aguiar et al., 2016, Villalon, 2012) and extrinsic, task-based evaluations (Rajaraman and Tan, 2002, Valerio et al., 2012). The exact evaluation procedure and data varies from paper to paper and the number of concept maps evaluated is usually small (often <5). While some evaluate their proposed approach in isolation, others compare it against baselines. However, we are not aware of a single paper that makes a direct comparison to any of the other works discussed in this section. This is a serious problem of the research on concept map mining so far, as it remains unclear which method performs best and how absolute and relative performances might differ depending on text genres, document types or other influencing factors.

We want to briefly mention additional work that is related to concept map mining, but does not produce concept maps as defined by Novak. For instance, de la Chica et al. (2008) focus on extracting sentence-long concept descriptions, making their work, although aimed

at concept maps, essentially multi-document summarization (see Section 2.3.2). Olney et al. (2011) propose a method that uses concepts and relations from pre-defined lists for biology education rather than the text, which makes it laborious to apply the method to other domains. Kof et al. (2010) extract concepts from two different documents and try to align them with each other, a process they refer to as “concept mapping”. There is also work on the creation of mind maps that uses the term concept map despite creating unlabeled relations, such as papers by Chen et al. (2006), Tseng et al. (2010), Chen and Bai (2010) and Lee et al. (2015). The applicability of these techniques to concept map mining is very limited, as most of them assume given concepts and solely focus on creating unlabeled relations.

2.3.1.1 Concept Mention Extraction

Given a set of documents, the goal of *concept mention extraction* is to identify all mentions of concepts, i.e. sequences of words in the input documents referring to a concept. All existing work for this subtask relies on automatic syntactic annotations to identify concept mentions, in particular part-of-speech tags and constituency (Marcus et al., 1993) or dependency (de Marneffe and Manning, 2008) parse trees.

Rajaraman and Tan (2002) and Kowata et al. (2010) both use regular expressions to define relevant sequences of part-of-speech tags. They both target noun phrases, covering nouns with optional preceding adjectives and noun phrases nested via prepositions. Valerio and Leake (2006) and Valerio et al. (2008) propose to use constituency parse trees instead of just part-of-speech tags and extract “minimal noun phrases”, i.e. noun phrases that do not have smaller noun phrases within them. Aguiar et al. (2016) define regular expression patterns over constituency parse trees. Likewise, earlier work by Oliveira et al. (2001) claims to rely on parse trees, but does not provide details.

As an alternative approach, Qasim et al. (2013) define patterns over dependency syntax representations. They target two-token concept mentions and extract sequences such as *electronic commerce* or *semantic web*. Using a short list of six patterns, covering adjectival and adverbial modifiers, noun compounds, prepositional constructions, conjunctions and simple, single-token nouns, they report good performance on their corpus. Zouaq and Nkambou (2008, 2009) pursue a similar direction and define 33 patterns. However, they only provide examples and do not reveal the full list of patterns. A slightly different approach has been suggested by Villalon (2012), who first applies a list of five transformations to a dependency tree and then selects all nodes containing a noun as concepts. The transformation operations merge nodes together based on the relation between them, yielding similar noun constructions as Qasim et al. (2013)’s method.

2.3.1.2 Relation Mention Extraction

The *relation mention extraction* subtask is about finding mentions of relations in the input documents and, similar to concept mention extraction, most approaches rely on syntactic structures for this. In addition to its span in the text, a relation mention also references two concept mentions whose relationship it describes. All work we are aware of requires these three mentions to co-occur in the same sentence and does not extract relations that are expressed across sentence boundaries.

In line with their concept extraction strategies, Kowata et al. (2010) and Rajaraman and Tan (2002) use regular expressions over part-of-speech tags, targeting constructions around verbs, Valerio and Leake (2006) and Oliveira et al. (2001) select verb phrases from constituency parses, but do not reveal the exact selection algorithm, and Zouaq and Nkambou (2008, 2009) define patterns over dependency structures. Both Qasim et al. (2013) and Zubrinic et al. (2015) find relations by extracting verb phrases that connect two concept mentions via subject and object dependencies. Using their transformed dependency parse tree, Villalon (2012) selects the tokens along the shortest path between two concept mentions. Olney et al. (2011) rely on semantic role labeling (SRL) in addition to a dependency parse, but, as mentioned earlier, in order to map predicates to a set of pre-defined relations rather than to find the actual words used to mention the relation.

Moreover, some authors also attempt to extract hyponymy relations between concepts that are not explicitly mentioned in the text. Qasim et al. (2013) rely on lexico-syntactic patterns such as “A including B and C”, known as Hearst patterns (Hearst, 1992), to derive that a hyponymy relation between (A, B) and (A, C) holds. Zubrinic et al. (2015) use a simple heuristic that assumes hyponymy if A is a substring of B, e.g. *concept map* is a hyponym of *map*. They also add additional hyponymy relations from a domain thesaurus. Since these relations lack a specific mention in the text, a default label such as “is type of” is typically used when these relations are included in the concept map.

2.3.1.3 Concept and Relation Mention Grouping

Having mentions identified, the goal of *concept mention grouping* is to group mentions of the same concept together. Similarly, *relation mention grouping* attempts this for relations. A simple baseline approach for this task is to group together all mentions that are exactly the same, e.g. all occurrences of *children*. More sophisticated approaches should also recognize morphological variants, such as *child*, or synonyms, such as *kids*, to be mentions of the same concept (or the same relation).

Towards that goal, Villalon (2012) uses stemming to unify morphological variations and Valerio and Leake (2006) merge mentions if one is a substring of another, ignoring tokens that are not tagged as nouns or adjectives. In order to also capture synonyms without any lexical overlap, a popular approach is to lookup such relationships in WordNet (Miller

et al., 1990), as suggested by Oliveira et al. (2001). Villalon (2012) does this to check equivalence on a per token basis, while Aguiar et al. (2016) use Lin (1998)’s algorithm to compute a similarity based on WordNet and merge mentions if the similarity is above a threshold. Zubrinic et al. (2015) additionally use a domain-specific thesaurus. As an alternative, resource-independent approach, Rajaraman and Tan (2002) propose a clustering method using term-frequency-based vector representations of concept mentions. All these works apply their methods to concept mentions, but none attempts to group relation mentions, although the techniques might be partially applicable for that as well.

The grouping subtask can be seen as a special case of coreference resolution (Jurafsky and Martin, 2009, Chapter 21.3ff), in which we are mainly interested in coreferences between noun phrases, as these are the phrases mostly extracted during concept extraction. Some work tries to resolve pronominal anaphora before concept extraction to increase recall (Oliveira et al., 2001, Qasim et al., 2013, Aguiar et al., 2016). However, we are not aware of any work that uses existing coreference resolution systems for the grouping task.

2.3.1.4 Concept and Relation Labeling

Having grouped the mentions together, an additional step is to choose one of them per group as the representative *label* that will be used in the concept map. Rajaraman and Tan (2002) use the most frequent mention when counting exactly repeated mentions within a group. For instance, if a concept has the mentions $\{children, child, children, kids\}$, then *children* is most frequent as it occurs twice. All other work, even if using a grouping strategy, does not report how labels are selected.

2.3.1.5 Importance Estimation and Concept Map Construction

In the final subtask of *concept map construction*, a labeled graph, the final concept map, has to be created. The graph’s nodes are concepts and they are labeled with the representative mentions selected in the previous step. Similarly, edges are relations with their labels.

The majority of existing work simply takes everything that was extracted and adds it to the concept map. To be more selective and create maps of reasonable size even if the input is large, several authors proposed to score concepts — here referred to as *importance estimation* — and then use only a fixed amount or fixed fraction that score highest. Valerio and Leake (2006) compute term frequencies for all stemmed nouns and adjectives in the input document and score a concept with the maximum frequency of its terms. An adaption of the popular TF-IDF scoring model (Spärck Jones, 1972) that works on the level of concepts, called CF-IDF, is used by Zubrinic et al. (2015). Villalon (2012) builds a term-sentence matrix, applies latent semantic analysis (LSA) to it to obtain scores for every sentence and uses only concepts and relations from the highest-scoring sentences. Instead of the input document, Aguiar et al. (2016) use the graph of all concepts and relations for scoring. They

calculate concept scores with the Hub Authority Root Distance (HARD) model (Leake et al., 2004), choosing concepts that are central and highly connected. With regard to relation selection, Qasim et al. (2013) choose among multiple relations for the same pair of concepts with a VF-ICF metric, preferring verbs that often occur (verb frequency) but only co-occur with a small number of concepts (“inverse co-occurrence frequency”).

All the scoring strategies described above aim at determining the importance or relevance of a concept (or relation) to select a subset that is representative for the input. The map construction becomes more difficult if one also tries to optimize for other objectives, such as producing a well-connected map. Simply selecting the most important concepts can yield many unconnected ones, as there might not be any relations between them. Zubrinic et al. (2015) try to avoid this: They pre-select a subset of the 100 most important concepts according to their CF-IDF metric, build a graph from them and then iteratively remove nodes with the lowest degree until reaching a target size of 25 to 30 concepts. Choosing by node degree, their approach tries to keep the concept map as connected as possible.

2.3.2 Text Summarization

If a concept map should be created to provide an overview of a set of documents, the selection of representative concepts and relations, as discussed in the previous section, becomes a crucial part of the task. A large body of research exists for this problem when the overview should be provided in textual form, a task known as *text summarization*.

The goal of automatic text summarization is to create a short, limited-size text that describes the most important contents of a given text document or set of documents (Nenkova and McKeown, 2011). The size limit is usually defined as the maximal number of words the summary text can have. Within that size limit, a good summary should provide as much information about the input document(s) as possible, should prefer the more important aspects of the content and should be a fluent and natural text.

Different variations of this task have been studied in the NLP community. *Single-document summarization (SDS)* deals with a single document that should be summarized, *multi-document summarization (MDS)* with creating a summary for several input documents. Other variations are *query-focused summarization*, where only contents relevant to a given query should be part of the summary (Dang, 2005), and *update summarization*, where the existing knowledge of a user is specified as a set of documents and should not be repeated when creating a summary for another, overlapping set of documents (Dang and Owczarzak, 2008). Many computational approaches for these tasks have been developed, presented and evaluated in the Document Understanding Conference (DUC)⁹ and Text Analysis Conference (TAC)¹⁰ series (Over et al., 2007).

⁹<http://duc.nist.gov>

¹⁰<http://tac.nist.gov>

In the remainder of this section, we present the main computational approaches to the summarization problem and point to seminal or exemplary papers for each direction. For a comprehensive review of all related work, we refer the reader to the surveys by Nenkova and McKeown (2011), Yao et al. (2017) and Gambhir and Gupta (2017).

2.3.2.1 Extractive Summarization

Extractive summarization systems produce summaries reusing parts — mostly complete sentences — taken from the input documents without modifications. More formally, let D be a set of documents, $\mathcal{S}(D)$ the set of all sentences in D and \mathcal{L} the maximal length of the desired summary. The task is then to select a subset of sentences $S \subset \mathcal{S}(D)$ with $\sum_{s \in S} \text{len}(s) \leq \mathcal{L}$, where $\text{len}(s)$ is the length of s in words. Two subtasks, *importance estimation* and *sentence selection*, are usually modeled to create extractive summaries.

Importance Estimation In order to include the most important information in a summary, the importance $i(s)$ of each sentence $s \in \mathcal{S}(D)$ needs to be estimated. Luhn (1958), the very first work on automatic summarization, used word frequencies to derive importance estimates for sentences. Almost 60 years later, summarization systems using frequency as the only indicator for importance still yield competitive results (Boudin et al., 2015). Edmondson (1969) added the position of a sentence in the document and the presence of predefined cue words as additional indicators. Among many other metrics explored in later work, importance estimates derived from graph structures with the PageRank algorithm (Page et al., 1999) had a particularly large impact. Both TextRank (Mihalcea and Tarau, 2004), which uses a graph representing co-occurring words, and LexRank (Erkan and Radev, 2004), using a graph of sentence similarities, have been regularly used as benchmarks. A commonality of all these approaches is that they use a hand-designed indicator (or several ones) to derive importance estimates, which makes these approaches *unsupervised summarization* models.

Given the large number of suggested metrics that indicate importance, *supervised summarization* systems that use annotated data to learn how to combine different indicators to make the best estimate have been explored as well. Early work in this direction was by Kupiec et al. (1995), who combine several features in a Bayesian binary classifier trained to decide if a sentence should be in a summary or not. Later work modeled the problem with probabilistic models such as hidden Markov models (Conroy and O’Leary, 2001) or logistic regression (Hong and Nenkova, 2014) and with support vector machines in classification (Yang et al., 2017) and regression (Li et al., 2007) setups. Typical features include term and document frequencies, sentence lengths, sentence positions, unigrams, bigrams, parts-of-speech, named entities, capitalization and stopwords (Berg-Kirkpatrick et al., 2011, Hong and Nenkova, 2014, Li et al., 2016a, Yang et al., 2017).

Recently, *neural supervised* models for importance estimation have been proposed by several authors. Cheng and Lapata (2016) use a combination of convolutional neural net-

works (CNNs), recurrent neural networks (RNNs) and attention to classify sentences for SDS. Cao et al. propose a regression model based on recursive neural networks for MDS (Cao et al., 2015) and a CNN-based model with attention and a ranking loss for query-focused summarization (Cao et al., 2016). A two-layer RNN with a set of hand-crafted features is developed by Nallapati et al. (2017). Al-Sabahi et al. (2018) propose a similar hierarchical encoder in combination with an attention mechanism. Compared to traditional supervised models, all of these approaches seem to benefit from the powerful distributed representations that neural networks can learn (Goldberg, 2017). A common trend is the use of an attention mechanism. Apart from that, a broad range of neural architectures has been proposed and none of them has so far been identified as being consistently superior.

Sentence Selection Once importance estimates for all sentences are available, the remaining task is to select the subset $S \subset \mathcal{S}(D)$ that makes the best summary. This is usually formulated as an optimization problem maximizing the importance within the size limit:

$$S = \arg \max_{S \subset \mathcal{S}(D)} \sum_{s \in S} i(s) \quad \text{s.t.} \quad \sum_{s \in S} \text{len}(s) \leq \mathcal{L}$$

In other words, one tries to include as many important sentences as possible while not exceeding the size limit. This optimization is difficult, as one has to decide whether it is better to add an important and long sentence to the summary or instead a less important but also shorter sentence, leaving more space for additional sentences. To make the best decision, one has to consider the full search space of all subsequent decisions, i.e. optimize globally. The optimization problem is known as the 0-1 *knapsack problem* and is NP-hard (McDonald, 2007). In the case of MDS, an additional challenge is that sentences from different documents might contain the same information. Thus, only one of them should be in the summary – although all of them are estimated to be equally important. This is typically handled by adding a redundancy penalty to the objective function, leading to an optimization problem that is also NP-hard (McDonald, 2007).¹¹

Carbonell and Goldstein (1998) proposed a greedy optimization approach called maximal marginal relevance (MMR). Sentences are added iteratively until the length limit is reached, choosing them based on their importance and redundancy with what is already in the summary. That does not necessary yield the optimal subset, but was shown to work well in practice. Other approaches, such as Hatzivassiloglou et al. (2001), rely on sentence clustering to first group redundant sentences together and then use only one sentence per cluster in the summary. Lin and Bilmes (2011) point out that the objective functions discussed here are submodular. For submodular objective functions, greedy optimization al-

¹¹For the easier version without the redundancy term, there is a pseudo-polynomial algorithm (Kellerer et al., 2010). However, it cannot solve the extended MDS problem including redundancy (McDonald, 2007).

gorithms with provable lower bounds exist that guarantee that a greedy solution is at most a constant factor worse than the optimal solution.

Exact solutions can be found by formulating the problem as an integer linear program (ILP), for which a broad range of off-the-shelf solver software exists. McDonald (2007) pioneered this approach, but also showed that it is much more computationally expensive than the greedy alternatives. Gillick and Favre (2009) proposed a new objective function that computes importance and redundancy in terms of included concepts rather than sentences. This has the advantage that the importance and redundancy terms simplify to a single term and yields ILPs that are more efficient to solve.

2.3.2.2 Abstractive Summarization

Extractive summarization methods have several problems. Using just the existing sentences, they might need to include unimportant details in a summary if something more important only occurs in a sentence together with these details. Moreover, extractive summaries can lack fluency and clarity, as the selected sentences might contain unresolvable pronouns or miss important context. Ordering the sentences in the most coherent way is a difficult problem on its own (Nenkova and McKeown, 2011). *Abstractive summarization methods* try to circumvent these problems by going beyond the set of existing sentences.

Sentence Modification Most of the early work has focused on compressing single or fusing multiple of the original sentences. By dropping unimportant parts from the sentences, the length budget of the summary can be used more efficiently. Both rule-based (Jing, 2000, Zajic et al., 2007) and learned (Knight and Marcu, 2002, Clarke and Lapata, 2007) models were proposed to compress sentences. Sentence fusion techniques (Barzilay and McKeown, 2005, Filippova and Strube, 2008) have also been explored since compressing only can lead to having unnaturally many short sentences in a summary. Rather than using these techniques as preprocessing for extractive models, joint models for selection and compression have also been proposed (Berg-Kirkpatrick et al., 2011, Chali et al., 2017).

Traditional Generation A summarization paradigm differing more radically from extraction is the generation of completely new sentences. Such models typically first parse the input documents into a symbolic meaning representation, then summarize that representation and finally generate a realization of the summary from it. While this approach gives a system more freedom to produce a good summary, a crucial point is that the intermediate representation offers enough representational capacity as well as good enough parsing and generation models. An early attempt in this direction was the system of Vanderwende et al. (2004) in DUC 2004. Li (Li, 2015, Li et al., 2016a) proposes an entity-based graph representation well-suited for news documents from which they successfully generate summaries.

Liu et al. (2015) used abstract meaning representation (AMR) as their intermediate representation, but left the generation step for future work. A proposition-based representation was shown to work well for educational texts, covering the full pipeline of parsing, summarization and generation (Fang and Teufel, 2016, Fang et al., 2016).

Neural Generation In recent years, the use of neural network models and large-scale training data led to improved performance in various NLP tasks, including text generation tasks such as language modeling (Mikolov, 2012) or machine translation (Cho et al., 2014, Sutskever et al., 2014). The predominant approach of generating text with word-level RNNs has been first applied to summarization by Rush et al. (2015). Their framework of using RNN encoder and decoder modules with attention was quickly adopted and refined (Nallapati et al., 2016, Chopra et al., 2016, Wang and Ling, 2016). These models are able to produce much more fluent summaries than previous generative models, and thereby substantially renewed the interest in abstractive summarization. Important extensions to this architecture are copy mechanisms that allow a model to include unknown words from the input in the summaries (Gu et al., 2016, See et al., 2017) and strategies to avoid repetitions in the generated sequences (Suzuki and Nagata, 2017, See et al., 2017). The greatest limitation so far is that most work focuses on SDS from a few sentences to short headlines, as training models for bigger inputs and outputs requires huge amounts of computational resources. In addition, no large-scale training corpora are available for MDS. Very recently, strategies such as pre-summarizing documents with extractive methods (Tan et al., 2017, Liu et al., 2018) or hierarchical encoders (Cohan et al., 2018, Celikyilmaz et al., 2018, Zhang et al., 2018) have been proposed to improve the scalability. These neural models are able to handle SDS examples with on average 5,000 input and 220 output words (Cohan et al., 2018) and MDS examples with 10k input and 100 output words (Liu et al., 2018).

2.3.3 Information Extraction

In order to create a concept map from natural language text, concept and relation mentions have to be extracted from the text. In the NLP community, these and other extraction tasks that try to obtain structured data from unstructured text are studied as *information extraction* (Jurafsky and Martin, 2009, Chapter 22). Traditionally, information extraction has been modeled as named entity recognition followed by relation classification (Doddington et al., 2004). Such techniques yield pairs of entities with relations between them, a structure that is already very similar to pairs of concepts and their relations needed for a concept map.

However, a limitation of classic information extraction is that relation extraction is modeled as a classification task, requiring a pre-defined list of supported relations and annotated training data for each of them. As Banko et al. (2007) points out, this renders these approaches impractical when applied to a large body of arbitrary text such as pages on the

web, for which it is impossible to anticipate all types of relations expressed in them. In our use case, a similar problem arises, as we want to create concept maps from documents before we know their content. The set of relations should thus not be constrained a priori. The idea of an open vocabulary for concepts and relations is a feature of a concept map (Novak and Cañas, 2008) that should be supported by the extraction mechanism.

As a solution, Banko et al. (2007) proposed *open information extraction (OIE)*, a variant of information extraction following the open vocabulary paradigm. OIE systems extract tuples such as (*Barack Obama - graduated from - Harvard Law*), consisting of two arguments connected by a relation, in which all parts are taken directly from the text. Every extraction has to be asserted by the text and arguments and relations should be as short as possible. Several extractions can be made from a single sentence.

Existing OIE systems can be classified along several dimensions:

Learned vs. Rule-based All OIE systems derive their extractions from the syntactic structure of a sentence. A key difference is whether the extraction patterns operating on the syntax are learned from data or have been hand-engineered. TextRunner (Banko et al., 2007), the first OIE system, uses a binary classifier to detect patterns. WOE (Wu and Weld, 2010) extends this idea by using Wikipedia infoboxes as supervision. Another learning approach is bootstrapping, employed by OLLIE (Mausam et al., 2012) and NestIE (Bhutani et al., 2016), which iteratively extends a set of initial seed patterns. Recent work by Stanovsky et al. (2018) showed that the task can also be formulated as a sequence tagging problem. Despite these successful attempts at learning patterns, a similarly large amount of OIE systems have been proposed that use carefully hand-crafted patterns based on linguistic insights. ReVerb (Fader et al., 2011), KrakenN (Akbik and Löser, 2012), ClausIE (Del Corro and Gemulla, 2013), Exemplar (Mesquita et al., 2013), OpenIE4 (Mausam, 2016), PropS (Stanovsky et al., 2016b) and Graphene (Cetto et al., 2018) are examples for this line of work.

Parsing vs. Part-of-Speech Tagging Due to Banko et al. (2007)’s initial motivation of using OIE to extract relations from all of the web, scalability has been an important design criterion. Early systems, such as TextRunner (Banko et al., 2007), WOE^{pos} (Wu and Weld, 2010) and ReVerb (Fader et al., 2011), rely only on part-of-speech tags to avoid the more costly dependency parsing used in most other systems. Therefore, these systems can achieve high processing speeds by sacrificing precision. In more recent work, this idea has been largely abandoned, presumably since better parsing algorithms and cheaper computational resources lowered the cost of dependency parsing.

Extraction Format While early work focused only on extractions of verb-mediated relations with two arguments, this was soon deemed to be too narrow to extract all relevant relations from text. KrakenN (Akbik and Löser, 2012) introduced the idea of n-ary extrac-

tions, adopted in most of the following work. OLLIE (Mausam et al., 2012) added context annotations, indicating that an extraction is only valid in a given context. NestIE (Bhutani et al., 2016) achieves the same by nesting extractions. Other extensions are including noun-mediated relations (Yahya et al., 2014) and minimizing argument spans (Angeli et al., 2015).

Language The large majority of work on OIE targets the English language. ExtrHech (Zhila and Gelbukh, 2013), a rule-based system for Spanish, is a first exception. Later work by Gamallo and Garcia (2015) led to ArgOE, a system that can process five different languages with the same set of rules. Following the same idea, PredPatt (White et al., 2016) develops a rule set working with universal dependencies (Nivre et al., 2016), making their system scale to potentially all of the 71 languages that are currently covered by the treebanks annotated with universal dependencies.¹² Their evaluation covers five languages.

More details on the various systems can be found in a recent survey by Niklaus et al. (2018). An important challenge for OIE has been the evaluation of systems due to the lack of annotated data, making most work rely on post-hoc manual evaluations. Recently, efforts by Stanovsky and Dagan (2016b) and Schneider et al. (2017) introduced the necessary evaluation data and tested a range of published systems. They found ClausIE and OpenIE4 to be the best performing systems according to their benchmarks.

OIE systems, in particular rule-based methods working with dependency parses, are very similar to the methods developed for concept and relation extraction in the concept map mining literature (see Section 2.3.1.1 and 2.3.1.2). However, we are not aware of any direct applications of existing OIE systems for the task of concept map mining.

2.3.4 Other Methods

In Section 2.1.3, we compared different structured text representations and pointed out differences between concept maps and alternative representations. While the previous sections introduced computational methods that are directly or indirectly related to concept maps, we now briefly look at automatic methods that have been developed for other structures. In fact, for most alternative representations, much more research than for concept maps has been carried out in the past. Given the large body of work, we focus on pointing out prominent examples for each direction.

As Marchionini (2006) pointed out, pure *information retrieval* methods (Manning et al., 2008) are not sufficient to support the interactive user behavior of exploratory search. To augment search engines for this use case, Crestan and de Loupy (2004) add lists of automatically extracted concepts and named entities to provide an overview of the documents and allow for quick follow-up queries. More generally, the concept of *faceted search* (Hearst

¹²Version 2.2 (July 2018) contains 122 treebanks spanning 71 languages (<http://universaldependencies.org>).

and Stoica, 2009) has been developed to add filtering and navigation capabilities to search results, sometimes automatically extracted from the documents.

The automatic extraction of *keyphrases* has been studied extensively, with the KEA system (Witten et al., 1999) being a prominent early example. Hasan and Ng (2014) provide a comprehensive survey. In general, methods for keyphrase extraction are very similar to concept extraction (see Section 2.3.1.1) and summarization (see Section 2.3.2) methods.

In order to automatically generate a *table-of-contents*, most methods employ a multi-step approach: First, segments in the input document are detected, for which cues such as paragraph breaks can often be used. Second, the hierarchical structure of the segments is determined (Yaari, 1998, Eisenstein, 2009, Erbs et al., 2013, Erbs, 2015, Pembe and Güngör, 2015). And finally, headlines for each segment are extracted from the document (Langer et al., 2004, Branavan et al., 2007, Nguyen et al., 2009, Pembe and Güngör, 2015). We are not aware of work that attempts to do this across multiple documents.

Another line of work studies *concept hierarchies* (also called taxonomies), which are trees — as tables-of-contents — but show concepts, similar to mind maps and concept maps. Early work extracted such hierarchies based on term co-occurrences (Sanderson and Croft, 1999, Sanderson and Lawrie, 2000), language models (Lawrie et al., 2001) or with clustering (Kummamuru et al., 2004). More recent approaches rely on refined notions of co-occurrences (Li et al., 2013, Kang et al., 2016), Hearst patterns (Kozareva and Hovy, 2010) or semantic similarity measures (Yang, 2012, 2015). Adler et al. (2012) pursue a different direction and construct hierarchies of statements and entailment relations between them. Yet another idea is the work by van Ham et al. (2009), who construct graphs of entities related by a specific, user-defined phrase, such as *X at Y* or *X and Y*.

The work of Luo et al. (2012) uses *event extraction* to structure documents and presents the development of events and their importance on a timeline. In a similar spirit, Shahaf and Guestrin (2010) visualize the connections between different news articles, derived from lexical overlap measures, as metro maps (Shahaf et al., 2012b), which can also be applied to scientific papers (Shahaf et al., 2012a). For scientific papers, another interesting structure can be obtained from a *citation analysis*, as done for instance by Lee et al. (2005) or Chou and Yang (2011), which reveals influential papers and authors.

Another, extensively studied approach to structure document collections is *clustering*. Clustering methods group similar documents together, using for instance partitional clustering (Cutting et al., 1992) or self-organizing maps (Kohonen et al., 2000), and often, to increase interpretability, also assign labels to the obtained clusters (Kim et al., 2015). Aggarwal and Zhai (2012) review common text clustering techniques in their survey.

A particularly popular clustering method are *topic models* created via latent Dirichlet allocation (LDA) (Blei et al., 2003), a probabilistic model that detects topics and derives a distribution over these for each document. To make such clusterings easier accessible, visualization tools for topic distributions (Chaney and Blei, 2012), topic development over

time (Dou et al., 2011, Liu et al., 2012) and hierarchical topic models (Griffiths et al., 2004, Smith et al., 2014) have been developed.

2.3.5 Exploratory Search Systems

To support a user during exploratory search in a document collection, the algorithms for text structuring discussed above need to be ultimately integrated into an end-user facing application, an *exploratory search system*. Typically, such systems combine a variety of text structuring and analysis tools with different visualization and interaction capabilities to provide a user all necessary functionality for a complex exploration task.

Since such applications target real world users and are meant to be used in practice, their development is mostly a commercial endeavor. Nevertheless, a few exploratory search systems have been built in academia and described in the literature. After all, the availability of these applications is inevitable to ultimately validate the practical usefulness of text analysis and structuring methods developed and studied by NLP research.

The Jigsaw system (Görg et al., 2013) is a freely available application for exploratory search in documents developed at Georgia Tech. It can load a set of documents and automatically creates summaries, clusters the documents and extracts keyphrases, entities and co-occurrence relations between them. The results of these analysis steps are presented in a range of lists, bar charts and plots that are interactively connected. A user can work with these visualizations and the documents, augmented with highlights, to navigate in the data and look at the content from different perspectives.

In a user study with 16 subjects (Kang et al., 2011), the system was compared against three baselines: the use of just the printed documents, of electronic documents with a search function and of the electronic documents with highlights for entities and keyphrases added. Subjects using Jigsaw performed substantially better during the staged intelligence analysis task, creating higher quality results while reading less, searching less and taking fewer notes. However, other work criticized the system for its complexity, stating that it “received little attention from journalists due to its unintuitive user interface” (Yimam et al., 2016).

New/s/leak (Yimam et al., 2016, Benikova et al., 2014) is a tool targeted at investigative journalists working with large collections of leaked documents. Its central visualization is a graph of named entities, including persons, organizations and locations, connected based on co-occurrence statistics. The graph can be used to interactively explore the content and is accompanied by a search function, document filtering, statistics about metadata as well as timeline and map visualizations. The tool has been developed in collaboration with journalists and found to be intuitive in a first user study.

Event Registry (Leban et al., 2014) is centered around events covered in news articles. In contrast to other systems, users cannot provide their own data, instead, a large and daily updated collection of news articles from around 75k providers is used. From these

articles, events are automatically extracted, clustered and matched across languages. For the resulting unique events, information such as a title, description, date and location is extracted. The web application allows a user to search in the growing event database and offers several visualizations of trending topics, concepts, categories and locations of events, providing an overview of the large news article collection.

NetLens (Kang et al., 2006) is an exploratory search system focusing on scientific publications. Using paper metadata such as authors, institutions, countries, topics and citations, it provides a range of connected lists and bar chart visualizations that can be used to interactively explore collections of papers. Papers can be accessed directly in the tool and interesting papers can be added to a personal list, allowing researchers to explore a research field and discover and collect relevant work.

2.4 Chapter Summary

In this chapter, we introduced and defined exploratory search in document collections, a task that journalists, researchers, intelligence analysts, lawyers and many other people face regularly. Analyzing a set of user studies, we found that users are particularly interested in key elements such as persons, organizations or facts and the relationships between them when trying to answer complex questions from textual data. A variety of computational models have been developed to automatically extract such structured representations from natural language text. When they are combined with visualizations as well as navigation and search functionalities to build exploratory search systems, they can support users during exploratory information seeking activities. In particular, the automatic extraction of relevant elements, concise overviews and easy navigation allow users to process collections far larger than what they could handle without such aids.

Concept maps, labeled graphs depicting concepts and their relations, are a form of structured text representation that is particularly useful. They can provide a concise overview of a collection that reveals key concepts and relationships while also allowing easy access to details. We argued that these properties make them specifically useful to support exploratory search and differentiate them from representations such as summaries, keyphrases, tables-of-content or mind maps. They have been successfully used in many application scenarios in education as well as knowledge and information structuring.

Furthermore, we reviewed existing concept map mining techniques that aim to automate the generation of concept maps from text. The task is usually approached in a step by step manner, starting with the extraction of concept and relation mentions from the text, grouping coreferent mentions together and then selecting a subset of them to construct a concept map. Pattern-based concept and relation extraction from syntactic structures, substring and WordNet-based concept grouping and frequency-based selection of important concepts have been explored. However, as existing work is spread across communities and

uses varying evaluation protocols, it remains unclear which of these techniques perform best. Additionally, we reviewed existing techniques for automatic text summarization and information extraction, two well-studied areas in NLP. Despite their overlap with subtasks of concept map mining, only the most basic methods developed in these areas have been applied to concept maps yet, while the concept map mining task itself has received little attention in the NLP community so far.

CHAPTER 3

Structured Summarization with Concept Maps

In this chapter, we will introduce the central problem studied in this thesis, the automatic creation of multi-document summaries in the form of concept maps. First, we motivate the task based on the review of existing work and user requirements laid out in the previous chapter. A formal definition of the task, a discussion of its challenges for computational models and a comparison to existing tasks follows. We close the chapter by suggesting several methods to evaluate and compare automatic methods for the task.

3.1 Motivation

As we discussed in detail in the previous chapter, supporting users during exploratory search in documents is an important problem. Providing an overview, offering navigation capabilities and revealing important elements and relationships are key functionalities to make exploratory search more efficient. Summaries in the form of concept maps can offer all of these features and are therefore a promising text representation in this scenario.

While there is existing work on concept map mining, reviewed in Section 2.3.1, with papers going back to 2001, we argue that research in this area is still at its beginning:

- The amount of existing work on concept map mining is very limited. In our literature review, we cite 23 distinct papers. In the NLP community in particular, almost no work on the task exists. A search in the ACL Anthology¹³ yields 8 results for “concept map” and 14 for “concept map mining”, of which 3 are papers written by the author of this thesis and only one other is relevant. In contrast, querying for “summarization” returns 668 papers, “named entity recognition” 735 and “event extraction” 172.

¹³<https://aclanthology.info/>

- In addition to the small amount of work in general, there is also no agreed upon evaluation protocol for proposed methods and no common evaluation data. As a result, a range of concept map mining techniques have been proposed, but no information about how they compare to each other in terms of performance exists. This makes it impossible for researchers to make measurable progress and also difficult for practitioners to decide which algorithms to implement in downstream applications.
- The existing work has been published in a range of different communities and venues, including information science (Qasim et al., 2013), expert systems (Zubrinic et al., 2012), learning technologies (Villalon and Calvo, 2009, Zouaq and Nkambou, 2008), knowledge management (Rajaraman and Tan, 2002, Zouaq and Nkambou, 2009) and concept mapping (Aguilar et al., 2016, Kowata et al., 2010, Valerio and Leake, 2006). Only one related paper appeared in an NLP venue (Olney et al., 2011). The fact that there is no single community that “owns” the task seems to contribute to the lack of comparability and common evaluation protocols.
- Although the core challenges of concept map mining are clearly natural language processing problems, little work from the NLP community has been applied to the task. As we showed in Section 2.3.2, for automatic summarization, powerful supervised feature-based and neural network-based models have been developed to determine which parts of a document should be included in a summary. For the related selection problem in concept map mining, only simple unsupervised frequency measures have been explored. Similarly, as Section 2.3.3 has shown, much effort has been invested in designing (open) information extraction systems that can process large and heterogeneous collections of text. Existing work on concept map mining largely ignored these efforts and hand-designed own extraction methods from scratch, often tailored to very specific domains and text types.

Overall, we think that the research on extracting concept maps from text is still in its beginning and that with increased attention, in particular within the NLP community, computational methods for the task can be greatly improved. Given this observation, in combination with the fact that concept maps are a promising representation for exploratory search from a user requirements’ point-of-view, the research of such methods is the goal of this thesis.

In particular, we reformulate the task as a variant of MDS where the summary has the form of a concept map, thus combining traditional MDS with information extraction and coreference resolution challenges. In the context of exploratory search in large document collections, the summarization aspect is important, as the amount of information to process can be huge. Starting with the research described in this thesis, we hope that this new task gains increased attention in the summarization and NLP communities. It is an interesting, application-oriented task at the intersection of several existing NLP tasks that allows researchers to test existing models and to start developing new approaches that target the

unique challenges of the task, all with exploratory search as the real-world use case that motivates the work and that can also serve as a testbed.

3.2 Task Definition

In this section, we precisely define the research problem of this thesis and introduce a few notational conventions. In Section 2.2, we summarized the concept map literature to describe what a concept map is, what it should look like and what makes a good map. Here, we extend this description by adding a more formal definition in terms of graph theory:

Definition 4: Concept Map

A *concept map* is a labeled, directed graph $G = (C, R)$ that has

- nodes C , with each $c \in C$ representing a concept with a unique label $l(c)$,
- and edges R , where each $r \in R$ is a directed edge from a source to a target node with a label $l(r)$ describing the relation between those concepts.

Labels, assigned by l which maps to sequences of words, should be as short as possible and relations should create well-formed propositions (see Section 2.2.1).

If a concept map is extracted from documents D , concepts and relations can be represented as sets of mentions, i.e. subsequences of D referring to them. A concept map is an *extractive concept map* if all labels are taken exclusively from the mentions of the corresponding concept or relation, otherwise, it is an *abstractive concept map*.

In the common definition of a directed graph, there can be at most one edge between two concepts c_1 and c_2 (and one from c_2 to c_1), but no so-called multi-edges. This is also generally assumed to be the case for concept maps, where authors typically choose to represent the most important relation between two concepts.

We further note that the purpose of having directed edges is solely to aid a user interpreting the relations. A triple of two concepts and their relation forms a meaningful proposition when read in that direction, such as (*concept maps - represent - organized knowledge*) (see the concept map in Section 2.2.1). In particular, the direction depends only on the specific surface form used in the relation label. One could easily imagine another concept map in which the same conceptual relationship is expressed as (*organized knowledge - can be represented by - concept maps*), changing the direction of the edge.

Based on this notion of a concept map, we can formally define the task for which we aim to study and develop algorithmic approaches in this thesis:

Definition 5: Concept Map–based Multi-Document Summarization

The goal of *concept map–based multi-document summarization (CM-MDS)* is, given

- documents D and
- size limits \mathcal{L}_C and \mathcal{L}_R ,

to create a concept map $G = (C, R)$ that

- (1) represents the most important concepts and relations in D ,
- (2) has no more concepts and relations than the limit, $|C| \leq \mathcal{L}_C$ and $|R| \leq \mathcal{L}_R$,
- (3) and is a connected graph.¹⁴

We will refer to the output of this task as a *summary concept map*.

The different parts of Definition 5 ensure that the output of the task is a structured text representation that is useful for exploratory search. While (1) requires the concept map to provide a representative overview of the documents, (2) makes sure that it is not arbitrarily large but indeed a summary. Since the summary has the form of a concept map, key concepts and their relations can be shown explicitly. Concept maps have concise labels and are meant to be easily interpretable for users (see Section 2.1.3). We add (3) to ensure that many useful relations are included and each concept can be related to the others by at least one path through the map.¹⁵ Since automatic methods for CM-MDS identify mentions when extracting the map from the documents, navigation to details can be easily implemented by allowing a user to click on concepts and relations to access the parts of the document collection around the mentions, effectively using the concept map similar to a table-of-contents. But in contrast to a normal table-of-contents, the concept map covers the content of multiple documents and can link from concepts and relations to multiple locations. Overall, CM-MDS therefore produces a structured text representation that is in line with all requirements laid out in Section 2.1.2.

Similar to SDS and MDS, one can distinguish an extractive and abstractive variant of CM-MDS based on Definition 4. In this thesis, we mostly focus on the extractive variant, but also explore methods that are capable of producing abstractive concept maps in Chapter 7. In line with most previous work, we do not consider the hierarchical arrangement of a concept map to be part of CM-MDS. We leave this — as well as the connected task of

¹⁴A graph is connected if there is a path between every pair of nodes in the graph. As we pointed out earlier, the direction of an edge in a concept map depends only on the label used for the relation. For the purpose of connectedness, we only care whether there is a relation between two concepts at all, and therefore use the undirected version of the graph to assert connectedness.

¹⁵Note that without (3), a valid summary concept map would be $G = (C, \emptyset)$, i.e. a list of concepts. That would be at odds with the user studies cited in Section 2.1.2 that showed that relationships are what many users are particularly interested in.

visualizing a concept map — as a subsequent step that can be approached independently, in particular within the visualization community.¹⁶

3.3 Subtasks and Challenges

In order to create a summary concept map for a collection of documents, multiple subtasks have to be solved. We will illustrate these tasks and their challenges for computational approaches based on the example shown in Figure 3.1.

3.3.1 Concept Mention Extraction

Given the input documents D , the first task is to identify all mentions of concepts in the text of these documents. We will denote the set of mentions as M .

Mentions of concepts can be of **various syntactic types** as the annotation study by Villalon et al. (2010) (see Section 2.2.3) already showed. Examples shown in the upper part of Figure 3.1 are nouns such as *hypnosis* and *caffeine*, proper nouns such as *ginkgo biloba*, more complex noun phrases like *the core symptoms of ADHD* but also verb phrases that describe activities such as *eating a healthy, nutritious diet*. In addition, concepts are often referred to using **pronouns**, as in sentence (6) of the example.

A main challenge of this subtask is to find an extraction approach with a good **trade-off between precision and recall**. A perfect method would exactly identify the highlighted spans of Figure 3.1. In practice, however, most methods are not perfect. If the precision is high, i.e. all identified spans are indeed mentions of concepts, some constructions are usually missed, lowering the recall. On the other hand, methods trying to obtain a higher recall might extract too many mentions, including false positives. In light of the subsequent steps, a high recall is important, as concepts that are missed here can never make it into the summary, but a very large set of extracted mentions also makes the later selection of summary-worthy elements harder in terms of the number of options to select from. Another challenge is **generalizability**: Extracting spans of a certain syntactic structure might yield only correct mentions on the text it was designed for, but may be too broad and cover many undesired spans on other types of text. Ideally, we want a method that is broadly applicable to many types of text.

3.3.2 Concept Mention Grouping

Having identified all concept mentions M , the next subtask is to group mentions of the same concept together. More formally, we want to find a partitioning C of M such that

¹⁶A vast amount of past and ongoing research exists on visualizing graphs. Surveys such as Di Battista et al. (1998), Herman et al. (2000), Katifori et al. (2007) and Liu et al. (2014) can serve as entry points into this field.

Concept Mention Extraction and Grouping

- (1) Caffeine, which is a mild CNS stimulant, reduces ADHD symptoms.
- (2) Hypnosis has little to no effect on the core symptoms of ADHD.
- (3) Herbal supplements such as ginkgo biloba have been used to treat the symptoms of ADHD.
- (4) Hypnotherapy might help with common ADHD symptoms.
- (5) However, it seems that it is not a very effective means of controlling ADHD symptoms.
- (6) Eating a healthy, nutritious diet can clearly benefit all children.

$M = \{\text{Caffeine, a mild CNS stimulant, ADHD symptoms, Hypnosis, ...}\}$

$C = \{\{\text{Caffeine}\}, \{\text{ADHD symptoms, ...}\}, \{\text{Hypnosis, Hypnotherapy, it}\}, \dots\}$

Relation Mention Extraction and Grouping

- (1) Caffeine, which is a mild CNS stimulant, reduces ADHD symptoms.
- (2) Hypnosis has little to no effect on the core symptoms of ADHD.
- (3) Herbal supplements such as ginkgo biloba have been used to treat the symptoms of ADHD.
- (4) Hypnotherapy might help with common ADHD symptoms.
- (5) However, it seems that it is not a very effective means of controlling ADHD symptoms.
- (6) Eating a healthy, nutritious diet can clearly benefit all children.

$O = \{(\text{Caffeine, is, a mild CNS stimulant}), (\text{Caffeine, reduces, ADHD symptoms}), \dots\}$

$R = \{(c_4, \{\text{has little to no effect on, is not a very effective means of controlling}\}, c_1), \dots\}$

Importance Estimation

$c_1 = \{\text{ADHD symptoms, ...}\}$	0.9	$(\{\text{Hypnosis, ...}\}, \{\text{ADHD symptoms, ...}\})$	
$c_2 = \{\text{ginkgo biloba}\}$	0.3	$\{\text{might help with}\}$	0.3
$c_3 = \{\text{all children}\}$	0.8	$\{\text{has little to no effect on, is not a very ...}\}$	0.8
...		...	

Concept Map Construction

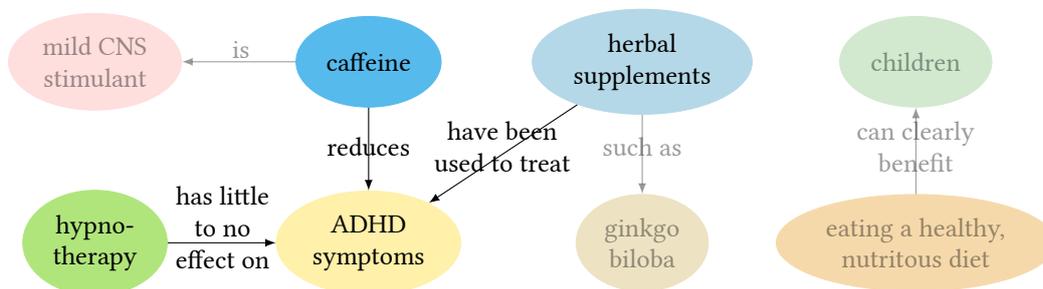


Figure 3.1: Subtasks of CM-MDS illustrated by examples. Six sentences are used as input. Mentions of concepts and relations are highlighted, color indicates groupings. Lightly drawn parts of the graph are not part of the summary. Note that the input sentences were picked to illustrate specific challenges, a real input text would be more coherent and have more complex sentences.

every subset of C contains mentions of a single, unique concept. The main challenge of this subtask is the **variety** of expressions that can be used to refer to the same concept. While several mentions of a concept can use exactly the same words, such as *ADHD symptoms* in (1) and (5) of the example, the same concept can also be mentioned using **synonyms** or **paraphrases** like *the symptoms of ADHD* or using a **pronoun**.

In addition to mentions that clearly refer to the same concept, one might also want to group mentions of slightly different concepts. Examples include concepts with a **hyponymy** relation, such as *the symptoms of ADHD* and *the core symptoms of ADHD*, or a **meronymy** relation, such as *hypnotherapy* and *hypnosis*. Since the ultimate goal is to create a summary of restricted size, the differentiation between them might not be needed. Ideally, such a merging decision has to be made considering the propositions that should be expressed by the final summary map, as they define the necessary **concept granularity**. And of course, one has to avoid that this leads to propositions that are not **asserted** by the text. For instance, if the second concept mention in sentence (6) would be *some children*, grouping it with other mentions to a concept *children* would be problematic, since the (hypothetical) sentence would not assert that eating a healthy diet is good for (all) children.

Finally, a **scalability** challenge arises if the methods that determine whether two mentions refer to the same concept are based on pairwise comparisons, causing a quadratic, i.e. $\mathcal{O}(n^2)$, runtime complexity. Such approaches can therefore only scale to a certain number of concept mentions and consequently only to document collections of a limited size.

3.3.3 Relation Mention Extraction

The goal of relation extraction is to identify all mentions of relations O , where each element $o \in O$ is a tuple (m_1, q, m_2) with q expressing the relationship between the concepts referred to by m_1 and m_2 . We refer to the tuple as a *proposition*.¹⁷

Similar to concept extraction, challenges of this subtask are the **variety** of expressions that are used to describe relationships and finding a good **trade-off between precision and recall** while ensuring **generalizability**. For relation mentions, the dominant class of verb-mediated expressions, such as *reduces* or *might help with* in Figure 3.1, accounts for only 47% of mentions in Villalon et al. (2010)’s annotation study. Other examples are prepositional phrases or adjectival phrases like *such as* in sentence (3). Moreover, some relations might be expressed **across sentence boundaries** and are therefore more difficult to extract. However, if one tries to go beyond sentence boundaries, an additional **scalability** challenge might arise, as any of the n^2 pairs of concept mentions in scope could be the source and target of a beyond-sentence relation mention. Within a sentence, n is typically so small ($n \leq 5$) that this does not constitute a challenge.

¹⁷We use the term proposition both on the document level, referring to a tuple of two concept mentions and a relation mention, and on the concept map level, referring to a tuple of two concepts and a relation.

Furthermore, since this subtask introduces propositions, the **assertedness** of them by the text has to be ensured. As negative examples, consider the following two sentences:

No studies prove that chiropractic helps with symptoms of ADHD.

Some people think diet supplements improve symptoms of ADHD.

Here, because both sentences consist of several nested statements, the inner propositions (*chiropractic, helps with, symptoms of ADHD*) and (*diet supplements, improve, symptoms of ADHD*) should not be extracted, as they are not valid without their context. Using them for CM-MDS would introduce false statements into the concept map. However, as both concept and relation labels of the concept map should be as **concise** as possible, we want to drop unnecessary nested clauses such as, for instance, non-essential relative clauses. A careful understanding of sentence structure is necessary to satisfy both goals.

3.3.4 Relation Mention Grouping

Similar to concept grouping, the goal of this subtask is to partition mentions of the same relation together. In contrast to concepts, we first group all mentions O by the pairs of concepts they were extracted for and then partition the mentions per pair. The result is a set R of relations, containing tuples $(c_1, \{q_1, q_2, \dots\}, c_2)$ that consist of two concepts c_1, c_2 – both sets of concept mentions – together with a set of relation mentions describing a unique relation between the two concepts.

As an example, consider sentences (2), (4) and (5) in Figure 3.1, each of them containing a mention of a relation between the concepts *hypnosis* (which itself has three different mentions) and *ADHD symptoms* (having five mentions in total). During relation grouping, we want the mentions in (2) and (5) to form a relation, as they express the same idea, while mention (4) is a different relation. Figure 3.1 shows the desired result. The challenges here are similar as for concepts, stemming mainly from the **variety** of ways in which the same relation can be expressed. In contrast to concepts, the scope and thereby also the relevance of this subtask is much smaller though because the partitioning is done per pair. That makes the sets of mentions that have to be partitioned substantially smaller, such that in practice comparison-based quadratic approaches are also more feasible.

3.3.5 Concept and Relation Labeling

In C and R , both concepts and relations are represented as sets of mentions. To show them in a concept map, a representative label is needed. Formally, this is a function l mapping from $C \cup R$ to sequences of words. One approach is to select one of the mentions as the representative label. For the concept map in Figure 3.1, *ADHD symptoms* has been chosen

among the different mentions of that concept. And for the relation mention group discussed above, *has little to no effect on* is used as the representative label.

The main goal of labeling is to have **concise** labels. There can be cases where, in order to achieve that goal, we want to go beyond the available mentions, as for example the concept labeled *children*, which has only one, slightly longer mention *all children*. Note that this is a first step towards abstractive concept maps, but a shortened mention is still a sequence of words present in the source text. Similar as for concept grouping, shortening mentions can introduce propositions that are not **asserted** by the text. If we had a concept with the single mention *some children*, labeling it as *children* would make the propositions it participates in more general than sanctioned by the text.

3.3.6 Importance Estimation

Since the summary concept map, being a summary, should typically only contain a subset of all concepts C and relations R found in the previous steps, the importance of these elements has to be determined. The subsequent and final step can then use this information to construct a map that contains only the most important concepts and relations. One way to model this is as a function ν assigning real-valued scores to elements in $C \cup R$.¹⁸ Figure 3.1 shows examples for this approach, giving more importance to the concept labeled *ADHD symptoms* but less to *ginkgo biloba*. Correspondingly, relations also receive scores.

A challenging factor of this subtask is the **size** of C and R . The bigger these sets are, the more options one has to choose from, which tends to make the selection harder. Often, deciding whether a concept is important and should therefore be part of the summary requires **external knowledge** in addition to what is in the source text, e.g. common sense knowledge. In addition, these decisions can also be **user-specific**. Factors such as the specific information need of a user, their personal background knowledge and subjective feelings and opinions can influence which content should be part of a useful summary.

Discussing the challenges of concept and relation extraction earlier, we pointed out that a higher recall might make this subtask more difficult. A similar, but contrary relationship between these tasks exists with regard to precision: If the extraction approach has a non-perfect precision, introducing some erroneously extracted concepts and relations, they can still be filtered out during the selection, compensating the earlier mistakes. Thus, a notion of **quality** can be an additional criterion when scoring concepts and relations.

3.3.7 Concept Map Construction

Finally, having concepts, relations and estimates of their importance, we have to construct the summary concept maps. Due to the grouping of concept and relation mentions before,

¹⁸Note that element-wise scoring is not the only possible approach to this problem. See Section 6.3 and surveys such as Liu (2009) for alternative ways to model this subtask.

propositions that mention the same concept are connected, yielding a graph $G = (C, R)$. According to Definition 5, the summary concept map has to satisfy the size and connectedness constraints while providing as many important aspects of the text as possible. Thus, this subtask amounts to selecting a subgraph $G' = (C', R')$ with $C' \subseteq C$ and $R' \subseteq R$ from G with respect to this goal and the constraints.

Figure 3.1 illustrates some challenges that this subtask entails. First of all, the size-constrained selection problem, i.e. choosing a subset $C' \subseteq C$ such that $|C'| \leq \mathcal{L}_C$, already poses the problem of **combinatorial explosion**. For $\mathcal{L}_C = 3$, there are 92 possible subsets of C . In general, $\sum_{k=1}^{\mathcal{L}_C} \binom{|C|}{k}$ subsets exist among which one would need to find the best according to some criterion, e.g. the sum of importance estimates of the selected elements. Second, the **connectedness** requirement makes the search for the best subset difficult, as it renders some subsets invalid. For instance, subgraphs can only contain *mild CNS stimulant* if *caffeine* is also included. Rather than simply adding the highest-scoring concepts, one might need to add some with a lower importance estimate in order to be able to include other high-scoring ones. Moreover, the graph G can consist of several completely **disconnected components**, as shown in the example. *ADHD symptoms* and *children*, both concepts deemed to be important, cannot be in a connected subgraph with the others.

We can formulate the selection as a constrained optimization problem. Given the graph $G = (C, R)$, importance estimates $\nu : C \cup R \rightarrow \mathbb{R}^{\geq 0}$ and size restrictions $\mathcal{L}_C, \mathcal{L}_R$, we want to find the subgraph $G' = (C', R')$ that maximizes importance:

$$\begin{aligned} \arg \max_{G'} \sum_{c \in C'} \nu(c) + \sum_{r \in R'} \nu(r) \\ \text{s.t. } |C'| \leq \mathcal{L}_C \wedge |R'| \leq \mathcal{L}_R \wedge G' \text{ is connected} \end{aligned} \quad (3.1)$$

Here, the objective function prefers subgraphs containing more important elements while the constraints ensure connectedness and size. As a simpler formulation, one can also just select a subset of concepts C' and let G' be the node-induced subgraph of G , i.e. the graph with nodes C' and all edges that exist between these nodes in G .

$$\begin{aligned} \arg \max_{G'} \sum_{c \in C'} \nu(c) \\ \text{s.t. } |C'| \leq \mathcal{L}_C \wedge G' \text{ is connected} \end{aligned} \quad (3.2)$$

In this second formulation, only the importance estimates for concepts are used in the objective function. While it does not ensure that the number of relations is below the limit, it is still of practical relevance, as we will show later.

The complexity of optimization problems similar to Equation 3.2 has been studied by several authors. An unconstrained version of the problem, i.e. without $|C'| \leq \mathcal{L}_C$, was studied by Álvarez-Miranda et al. (2013) and El-Kebir and Klau (2014) and shown to be NP-

hard. A version with an exact cardinality constraint, i.e. $|C'| = \mathcal{L}_C$, is the subject of Backes et al. (2012)’s work. Conrad et al. (2007) study a more general version of the problem with costs per node and a constraint on the total cost, of which Equation 3.2 is a special case with a cost of one for every node. That problem has also been proven to be NP-hard.

In addition to the subgraph selection problem, a last challenge is **pairwise relation selection**. Given C and R , we can have multiple relations connecting a pair of concepts, making $G = (C, R)$ a directed graph with multi-edges. As discussed earlier, a concept map can only have one relation per pair of concepts, thus, we need to select among them. This could be handled as a preprocessing step to the optimization problem or approached jointly as a part of it. In Figure 3.1, pairwise relation selection is necessary for the concept pair (*hypnosis*, *ADHD symptoms*). Among the two available relations (obtained from merging three relation mentions), we need to select one. Similar to the general concept and relation selection task, importance estimates can be used for that. As indicated by the example, this problem can become particularly relevant if the potential relations **contradict**. Recognizing these cases requires a precise understanding of semantic relationships, specifically entailment, between the relations and the corresponding propositions. Moreover, resolving the conflict to the correct option – if applicable – might require access to **external knowledge**.

3.3.8 Task-Level Complexity

In addition to the challenges of each subtask, further complexity arises from the fact that some subtasks use the output of a previous subtask as their input. Figure 2.2 introduced earlier illustrates these dependencies. If the output of one subtask is imperfect, e.g. it has a non-perfect recall, the achievable performance of a subsequent step is limited. This challenge, known as **error propagation**, can lead to task-level performances much lower than the average per-subtask performance. For instance, if concepts are extracted, grouped and selected with 80% recall each, the final summary will only have a concept recall of 51%.

3.4 Relations to Existing Tasks

The task proposed in Definition 5 is closely related to several NLP tasks discussed in Section 2.3. In this section, we will point out commonalities and differences.

3.4.1 Concept Map Mining

Obviously, CM-MDS is closely related to concept map mining (see Section 2.3.1), as it is also our goal to automatically create concept maps from text. Given the limitations of the existing research, discussed at the beginning of this chapter, studying concept map mining

already seems to be warranted. Yet, we want to emphasize that CM-MDS differs from concept map mining in several aspects, making it an even more relevant task.

First, by formulating the task as a variant of MDS, we add a particular focus on the summarization aspect. In exploratory search scenarios, document collections are typically huge, requiring effective techniques to reduce the content to a small overview. Existing work for concept map mining mostly focused on the extraction subtasks, paying only little attention to summarization. For instance, we are not aware of any previous work that explicitly included a size restriction in their task formulation. Second, the concept map construction subtask has been largely ignored, with only Zubrinic et al. (2015) proposing a first heuristic algorithm. No work has explicitly formulated an optimization problem with an objective function and constraints as described in the previous section.

In addition to this new focus, we want to emphasize that also the other subtasks, namely extraction, grouping and labeling, are far from being solved. With the already explored techniques reviewed in Section 2.3.1, most of the challenges discussed in the last section cannot be handled well enough for the approaches to be broadly applicable in practice.

3.4.2 Text Summarization

Given Definition 5, several obvious parallels to the task of MDS (see Section 2.3.2) are visible: Both tasks work with multi-document inputs, both have a restricted size budget and both require producing the best possible summary within that budget. The main difference is that the summaries have different forms, one being a text, while the other is a concept map. For the latter, applying known MDS techniques alone is thus not sufficient, as the content also has to be structured into concepts and relations. While we are not aware of any work from the summarization community that produces summary concept maps, other types of structured summaries have been explored. Most notably, Haghighi and Vanderwende (2009), Christensen et al. (2014) and Tauchmann et al. (2018) all propose forms of hierarchically organized textual summaries covering different parts of a collection, giving a user, similar as with our concept map-based approach, more navigation capabilities.

Both MDS and CM-MDS require some form of importance estimation. As the existing work on concept map mining in this regard is limited (see Section 2.3.1.5), the application of techniques from MDS is an interesting direction to explore. In contrast to most extractive MDS work, the importance of concepts and relations, rather than sentences, has to be estimated. MDS approaches working with bigrams (Gillick and Favre, 2009, Boudin et al., 2015) or entities (Li, 2015, Li et al., 2016a) are closest to that requirement.

With regard to selection, two main differences are important. First, the notion of redundancy, a central topic in MDS, is less relevant for CM-MDS. As we work with smaller units, concepts and relations, and since redundant mentions of them are already grouped together in previous steps, no redundancy should exist among the elements considered during sub-

graph selection. Explicitly modeling redundancy reduction as an objective is therefore not necessary. The MDS work of Gillick and Favre (2009) goes in a similar direction by making selections based on bigrams instead of sentences. However, the set of bigrams can still contain different bigrams with the same meaning, such that redundancy remains a challenge.

Second, the selection problems are different optimization problems. The MDS selection problem is difficult due to the mismatch of cost (length budget used by a sentence) and utility (contributed important content) per element, resulting in the NP-hard knapsack problem. For CM-MDS, the cost is always one, as only a cardinality restriction is defined, such that one can optimize by simply selecting by utility. However, the additional connect- edness requirement introduces a new restriction, making an optimal selection still difficult. Due to the different constraints, sentence selection techniques discussed in Section 2.3.1.5 cannot be directly applied. More similar is the work on abstractive summarization by Li et al. (2016a) and Liu et al. (2015), who summarize graphs obtained by semantic parsing methods, resulting in a similar subgraph selection problem.

As challenges of the importance estimation, we mentioned the dependence on world knowledge and user-specific information. While these ideas apply to MDS as well, most of the existing work ignores them and deals only with generic summarization, i.e. using only information from the source documents. Notable exceptions are Louis (2014) and Peyrard (2018), who aim to integrate background knowledge, and P.V.S. and Meyer (2017), who propose an interactive summarization system that tailors a summary to a specific user.

3.4.3 Information Extraction

While summarization, as discussed above, has similarities with the importance estimation and concept map construction subtasks of CM-MDS, the remaining subtasks, in particular mention extraction and grouping, are essentially information extraction tasks. As we already pointed out in Section 2.3.3, due to the open vocabulary property of concept map labels, *open* information extraction is closest to the mention extraction subtasks.

In both tasks, we identify relations between two arguments in text and use labels taken from that text to describe that proposition. Conciseness and assertedness are desired in both cases. One difference is that the notion of an OIE argument is typically broader than that of a concept, such that not all propositions extracted by OIE are relations between two concepts.¹⁹ And second, several OIE systems, in particular more recent ones, extract n-ary tuples. A concept map, however, can by definition only represent binary relationships between pairs of concepts and can therefore not use all OIE extractions directly.

To the best of our knowledge, despite the high similarity between OIE and concept and relation mention extraction, no work has yet explored the application of OIE to the latter. Similarly, a large body of work on coreference resolution exists, surveyed for instance

¹⁹For more details and examples, see step 2 of the corpus creation described in Section 4.3.2.

by Zheng et al. (2011), but only rudimentary techniques have been used for concept and relation grouping. In Chapter 5 and Chapter 6, we therefore compare existing information extraction methods against techniques that were specifically designed for concept maps.

3.4.4 Knowledge Graphs

Concept maps have some similarity with more formal knowledge representations, of which many have been developed and used in the past. One example are knowledge bases such as Freebase (Bollacker et al., 2009) or WikiData (Tanon et al., 2016), which are databases of facts about real-world entities like politicians, celebrities, countries, organizations or places. Much work has been done in the NLP community to populate these databases automatically from text, for example as part of the automatic knowledge base construction workshops (Pujara et al., 2016). Projects like Cyc (Lenat et al., 1986), which are mainly driven by the knowledge representation and reasoning community, have a different focus as they aim to formally capture common-sense knowledge that is rarely represented in text. Within the semantic web community, the construction of domain ontologies has been studied extensively (Breitman et al., 2007). Rather than facts about real-world entities, ontologies model general concepts, their properties and relationships. Methods to automatically extract ontologies from text are studied as ontology learning (Maedche, 2002). More recently, the term knowledge graph became popular, encompassing both knowledge bases and ontologies, but it still lacks a clear definition (Ehrlinger and Wöß, 2016).

As pointed out in Section 2.1.3, a main difference to concept maps is that knowledge graphs are more formal. Whereas concept maps follow the open label paradigm and are meant to be interpretable by humans, knowledge graphs are more strictly typed – according to a predefined schema – as their purpose is to be machine-readable. Ontology learning and knowledge base construction methods are therefore not directly applicable. However, the work by Zouaq et al. (2011) showed that the similarities between the tasks can be leveraged by using concept map mining as a preprocessing step for ontology learning.

Another difference is that the ultimate goal of ontology learning and knowledge base construction is to create high-quality knowledge graphs. Therefore, the focus is on making high-confidence extractions or on finding concepts and relations still missing in the graph. Whether all the information present in a specific text is captured by extraction methods is less relevant, as the techniques could also just be applied to additional documents to grow the knowledge graph. In contrast, for CM-MDS, fully covering a given text is much more important, as only then an accurate summary can be produced.

The recently proposed open knowledge graphs (Witvies et al., 2017) are a knowledge representation that is less formal than the representations discussed above and follows, similar to concept maps and OIE, an open label paradigm for its elements. Open knowledge graphs depict entities and propositions describing their relationships, both being the result of an

extraction and grouping process of coreferent mentions. The authors apply the representation to redundant tweets about news topics, achieving great amounts of consolidation by grouping coreferent mentions. As an additional feature, the representation also derives lexical entailment relations between elements of a coreference set.

While this effort is by far the closest to our work from the area of knowledge graphs, there are a few notable differences: First, although the authors position their representation as a form of text summarization (Shapira et al., 2017), its summarization capability stems solely from the consolidation of redundant mentions. Higher degrees of compression that require the selection of a subset of the content, as typically attempted by MDS and also CM-MDS, are not a goal of that work. Second, it has so far only been applied to tweets about events reported in news articles, leaving it unclear how well the representation works for genres with a broader variety of entities and propositions. And third, while a large body of user studies on various usages of concept maps have been carried out in the past and outlined concept maps’ advantages (see Section 2.2.2), the effectiveness of this newer representation is — due to its recent invention — not yet as well understood.

3.5 Evaluation

Evaluation metrics are indispensable to determine which one among alternative computational approaches for a task performs best and whether newly proposed techniques in fact constitute progress. Towards that goal, at least three different types of evaluations are commonly used across the different tasks studied in the NLP community:

- (1) Automatic comparison against references
- (2) Manual (comparative) quality judgments
- (3) Task-based (comparative) evaluations in end user applications

3.5.1 Evaluation Methods for Text Summarization

For MDS, an example for (1) is the ROUGE metric (Lin, 2004). In general, automatic methods require both an appropriate method to quantify the correspondence between the machine-generated output and the reference as well as evaluation datasets that come with high-quality reference outputs. The advantages are cheap, fast and repeatable evaluations, since they can be done fully automatic after the references have been created once.

A problem with (1) is that as long as the problem of automatic natural language understanding is not fully solved, outputs of tasks like summarization can never be completely accurately compared by automatic methods. A metric like ROUGE, which is based on counting lexical overlaps, would give low scores to a summary that contains the same content as the reference but expressed in different words. Methods of category (2) are therefore

regularly used in conjunction with automatic methods. For MDS, one such method is the pyramid method (Nenkova and Passonneau, 2004, Nenkova et al., 2007), which scores summaries after manually matching content units against the reference summary. Other approaches are to show generated summaries to humans, which are then asked to score them or to express a preference between two alternative summaries. Recent examples applying this evaluation include the work of Celikyilmaz et al. (2018), asking for overall summary quality, and Paulus et al. (2018), asking specifically about readability. The DUC conference (Dang, 2005) developed a catalog of dimensions such as grammaticality, non-redundancy, coherence or responsiveness according to which such manual quality judgments are made.

Finally, as none of the aforementioned methods can determine whether generated summaries are actually useful, evaluations of type (3) assess that. Typically, these are user studies in which subjects perform a task for which summaries are supposed to be helpful. Different groups of subjects receive summaries produced by different approaches and the subjects' performance in the task is compared across groups. While only this type of evaluation can show whether automatic methods are ultimately useful in practice, it also has challenges: Compared to other evaluations, type (3) requires much more effort, is not easily reproducible and needs a careful study design to avoid confounds. Examples for MDS are Maña-López et al. (2004), McKeown et al. (2005) and Roussinov and Chen (2001).

3.5.2 Proposed Evaluation Methods for CM-MDS

Given the advantages and limitations of each of the evaluation types, we propose to also use a combination of all of them for CM-MDS. In fact, examples for all three categories have previously been used to assess concept map mining techniques. Villalon (2012) and Aguiar et al. (2016) use automatic comparisons against references (type 1), Kowata et al. (2010), Qasim et al. (2013) and Zubrinic et al. (2015) let experts judge the quality of concept maps (type 2) and Rajaraman and Tan (2002) and Valerio et al. (2012) evaluate them in task-based user studies (type 3). The main problems limiting the comparability of that existing work is that the exact procedures and data differ from work to work.

We hope to reduce that problem in the future by proposing standardized evaluation procedures for CM-MDS. In the following, we outline these procedures for evaluations of type (1) and (2). When results of these evaluations are published, subsequent work can simply run the same evaluations following the description here to compare their approach. For evaluations of type (3), comparisons can typically only be made within one experiment, and we therefore do not give specific recommendations here.

3.5.2.1 Automatic Metrics

In Section 2.2.2, we mentioned the study of McClure et al. (1999), which analyzed different manual scoring methods for concept maps in the context of student testing. They compared

holistically scoring a map, scoring its structure and scoring each proposition independently, all with a master concept map as the reference. The latter method was found to be most reliable and showed the highest correlation with the true assessment of the student maps (obtained through a more laborious manual analysis). Inspired by these findings, we also designed proposition-level metrics to score concept maps.

Given input documents D , a reference summary concept map $G_R = (C_R, R_R)$ and a system-generated summary concept map $G_S = (C_S, R_S)$, we create sets of propositions

$$P_x = \{ l(\text{source}(r_i)) \ l(r_i) \ l(\text{target}(r_i)) \mid r_i \in R_x \} \quad (3.3)$$

where x indicates either the reference or system map and the functions *source* and *target* assign a relation to its source and target concepts. To score a generated concept map, we then calculate the overlap between the sets P_S and P_R .

For evaluation purposes, we distinguish between two different task settings: One where the size of the summary concept map is restricted by both \mathcal{L}_C and \mathcal{L}_R and a second where only \mathcal{L}_C is given. In the first setting, similar to the traditional MDS setting, the summary size is bound and most systems will try to fully use that budget. It is therefore sufficient to compute only recall when comparing P_S and P_R . In the second setting, while the number of concepts is still constrained, the number of relations and thus the size of P_S can differ. By reporting both precision and recall, a fair comparison between systems can be made in this setting. In fact, we think that the second setting is more interesting, as it gives models the flexibility to either focus on high precision or high recall.²⁰

METEOR Our first metric is based on METEOR (Denkowski and Lavie, 2014), which has the advantage that it takes synonyms and paraphrases into account²¹ and does not solely rely on lexical matches (as it is the case for ROUGE). For each pair of propositions $p_s \in P_S$ and $p_r \in P_R$ we use METEOR²² to calculate a match score $\text{meteor}(p_s, p_r) \in [0, 1]$. Then, precision and recall per map are computed as:

$$Pr = \frac{1}{|P_S|} \sum_{p \in P_S} \max\{\text{meteor}(p, p_r) \mid p_r \in P_R\} \quad (3.4)$$

$$Re = \frac{1}{|P_R|} \sum_{p \in P_R} \max\{\text{meteor}(p, p_s) \mid p_s \in P_S\} \quad (3.5)$$

²⁰Note that the size restrictions \mathcal{L}_C and \mathcal{L}_R only limit the size of the graph, but not the size of its content. A possible adversarial attack against the metric would be to create a concept map with very long labels, in the extreme case containing all of D . The recall would be 100%. Obviously, such a concept map is not a summary and the labels would not be concise. It illustrates that automatic evaluation metrics typically have limitations. A comprehensive evaluation should therefore also rely on manual inspections. Note also that this attack does not work in the second setting, where the high recall would come with low precision.

²¹As determined using WordNet (Miller et al., 1990) and PPDB (Pavlick et al., 2015).

²²We use METEOR version 1.5 with default settings.

The F1-score is the equally weighted harmonic mean of precision and recall. Scores per map are macro-averaged over all instances of a dataset.

ROUGE As a second metric, we compute ROUGE (Lin, 2004). We concatenate all propositions of a map into a single string, s_S and s_R , and separate propositions with a dot to ensure that no bigrams span across propositions and the metric is therefore independent of how propositions are ordered. We run ROUGE 1.5.5²³ with s_S as the peer summary and s_R as a single model summary to compute the ROUGE-2 score, the most commonly used variant of the ROUGE metric family.

Significance Testing An important step when using automatic metrics is to determine whether a difference in the average scores that two methods A and B achieve on a dataset is meaningful or only due to chance. The smaller the difference is, the more relevant this question becomes. Statistical tests can be used to determine if a difference is in fact *significant*. Following suggestions for other NLP tasks, we propose to rely on sampling-based tests, which, in contrast to commonly used parametric tests like Student’s t-test, do not make any assumptions on the distribution of the evaluation scores or evaluation data (Dror et al., 2018). Since such assumptions do not hold for many NLP metrics, such as precision, recall and F-scores, using parametric tests is problematic (Yeh, 2000, Dror et al., 2018).

We propose to use a **permutation** or **randomization test** (Noreen, 1989), as also suggested for machine translation by Riezler and Maxwell (2005). Let A and B be two alternative methods with evaluation scores of $E_A = (a_1, a_2, \dots, a_n)$ and $E_B = (b_1, b_2, \dots, b_n)$ over n instances of an evaluation dataset. The average performance difference is then

$$d(E_A, E_B) = \left| \frac{1}{n} \sum_{a_i \in E_A} a_i - \frac{1}{n} \sum_{b_i \in E_B} b_i \right|. \quad (3.6)$$

For the randomization test, we create N samples by swapping evaluation scores between A and B at each position with probability 0.5, yielding N new pairs of score lists such as $E'_A = (b_1, a_2, b_3, \dots)$ and $E'_B = (a_1, b_2, a_3, \dots)$. The p-value is then defined as

$$p = \frac{1 + \# \text{ samples with } d(E'_A, E'_B) \geq d(E_A, E_B)}{1 + N}. \quad (3.7)$$

If p is sufficiently small, we reject the null hypothesis that there is no difference between A and B and conclude that A and B differ significantly. The permutation test, in contrast, checks all 2^n possible ways of swapping scores between E_A and E_B rather than just drawing N samples out of them. While being more accurate, it can be prohibitively expensive to compute for large n , such that in practice one mostly relies on the approximate ran-

²³We run ROUGE with parameters `-n 2 -x -m -c 95 -r 1000 -f A -p 0.5 -t 0 -d -a`

domization test. In this thesis, we make use of the permutation test when possible and otherwise rely on the randomization test. We use thresholds of 0.01 and 0.05 to determine significance, but also report the actual p -values for greater transparency.

3.5.2.2 Manual Quality Dimensions

For manual evaluations of type (2), we experimented with several setups and recommend using pairwise comparisons of different summary concept maps rather than assessing their quality in isolation. Most recent works on MDS, for instance Celikyilmaz et al. (2018), use this pairwise approach, as it makes it easier to discover differences between summaries.

A difficulty that arises is the layouting of concept maps. Different ways of presenting the concept map graph can greatly influence a rater’s perception of the map’s quality, and since layouting is a non-trivial issue, it is difficult to automatically create layouts for different maps that are “equally good”. As the goal is to evaluate the output of CM-MDS approaches, we do not want the layout quality to influence the evaluation. As the solution, we propose to not present the concept maps in a graphical form at all, but instead as lists of propositions. Such lists can be easily shown side by side and allow a fair comparison of the content. In addition, when showing propositions, even raters unfamiliar with concept maps can easily perform the evaluation, as a proposition is essentially just a short sentence.

Given two summary concept maps, we show their propositions side by side and ask raters to pick their preferred list according to the following dimensions:

Meaning The sentences should be understandable and express meaningful facts.

Grammaticality The sentences should be grammatical, without missing or unrelated fragments or other grammar-related problems that make them difficult to understand.

Non-Redundancy There should be no unnecessary repetition within the list, neither of whole sentences nor of partial facts.

Focus The sentences should be focused; sentences should only contain information that is related to the given topic description.

The dimensions are inspired by the (more fine-grained) list used during the DUC competitions (Dang, 2005). The first two dimensions assess the quality of the extraction subtasks by checking whether concept and relation labels form meaningful and grammatical propositions. The third dimension focuses on non-redundancy, assessing whether the grouping subtasks were handled successfully. Finally, the fourth dimension evaluates the summarization aspect, i.e. which content has been selected for the summary concept map.

We note that the automatic evaluation described before mainly focuses on whether a summary contains the same content as a reference summary, whereas the procedure described here captures more aspects of summary quality. However, the evaluation of content

selection is more limited in this setup: Since raters have no access to the full source documents or a reference summary, but see only a topic description, they can only assess how much of a map’s content is relevant, but do not know if there is other more relevant content. Nevertheless, we believe if both evaluation techniques are used in combination, such shortcomings are compensated as both techniques complement each other very well.

3.6 Chapter Summary

In this chapter, we introduced the central problem studied in this thesis. Based on the review of user requirements in the previous chapter, we argued that concept maps are a useful text representation to support users during exploratory search. Given the limited amount of existing work on extracting concept maps from text, more research in this direction is necessary. As a result, we proposed concept map-based multi-document summarization (CM-MDS), a reformulation of concept map mining with a focus on summarization.

We formally defined the different subtasks of CM-MDS, which are mention extraction, grouping and labeling for concepts and relations followed by importance estimation and concept map construction. Moreover, we presented a comprehensive example and illustrated the challenges for computational models for each subtask. For extraction and grouping, these include handling the variety of ways in which mentions can be expressed, finding extraction approaches that generalize to many text types and that successfully trade-off precision and recall, determining appropriate concept granularities, ensuring assertedness, scaling to large inputs and finding concise labels. Further challenges are potentially large numbers of extracted concepts and relations, external and user-specific factors influencing importance, the combinatorial complexity of the subgraph selection, in particular due to its connectedness requirement, and resolving contradictions between propositions.

In addition, we discussed relations to existing work, pointing out that the task is essentially a combination of text summarization with information extraction. Other relations exist to work on knowledge graphs, but clear differences exist. In order to evaluate computational models for the task, we proposed two automatic metrics based on METEOR and ROUGE. To account for the limitations of automatic methods and to better cover aspects beyond content selection, we further suggest to also manually compare different automatically created concept maps against each other with regard to their meaningfulness, grammaticality, non-redundancy and focus and to ultimately also evaluate the usefulness of the created maps in downstream applications by conducting user studies.

Creation of Benchmark Corpora

In this chapter, we will look at the data that is needed to train and evaluate computational methods for CM-MDS. We discuss requirements for suitable corpora and show that corresponding data does not yet exist. Therefore, we describe two different strategies to obtain data – by automatically extending partial annotations and by creating annotations from scratch with scalable methods – and the corpora that we obtained using these strategies.

4.1 Motivation and Challenges

In order to evaluate methods for CM-MDS with the automatic methods discussed in the previous chapter, datasets with appropriate annotations are necessary. As we argued before, being able to run such automatic evaluations is not only important to compare to other work, but also to guide the development of new methods. In addition, such data is also essential to develop any kind of supervised model for the task or to tune hyper-parameters of supervised and unsupervised models.

Specifically, we need datasets of pairs (D, G) where D is a collection of documents and G is a high-quality reference concept map that summarizes D according to Definition 4 and Definition 5. The pairs should satisfy the following requirements:

- G should be a valid concept map and an appropriate summary of D .
- Using the information in D , it should be possible to create a map close to G .
- The dataset should have enough pairs to observe statistically significant differences between evaluated approaches.
- Ideally, to deal with the inherent subjectiveness of summarization, G should be a consensus of several annotators or multiple alternative maps G should be used.

Table 4.1 shows the datasets used in existing work on concept map mining that made use of reference annotations. Unfortunately, none of these datasets can meet all of our requirements. In fact, most of them violate even several: Some are of very small size (Qasim et al.,

Author	Pairs	Per Pair				
		Maps	Concepts	Relations	Docs	Tokens
Valerio and Leake (2006)	80	1	1	–	1	8821
Qasim et al. (2013)	1	1	370	–	65	?
Zouaq et al. (2011)	1	1	1286	1797	36	36008
Aguiar et al. (2016)	1	10	?	?	1	617
Villalon (2012)	42	1	21	12	1	468

Table 4.1: Datasets with reference annotations used for automatic evaluations of concept map mining. Values in the last five columns are averages over all pairs, ? indicates values not reported.

2013, Zouaq et al., 2011, Aguiar et al., 2016), some provide only concept annotations but no relations (Valerio and Leake, 2006, Qasim et al., 2013) and some contain single-document rather than multi-document summaries (Valerio and Leake, 2006, Aguiar et al., 2016, Villalon, 2012). In addition, Villalon (2012)’s corpus, which comes closest to our requirements, is not available for other researchers to use.²⁴ Thus, to enable automatic evaluations for work on CM-MDS, we have to create new corpora.

The manual creation of such a dataset is very time-consuming, as the annotation includes many laborious subtasks. Essentially, an annotator would need to perform exactly the subtasks outlined in Section 3.3 to create a reference concept map for a set of source documents. For a full dataset with many pairs, the corresponding effort would be huge. An additional challenge lies in the nature of summarization: Summaries should be much smaller than their source documents and contain only the most important parts. However, in order to create the summary concept map G , a human annotator has to read all documents D and find and group mentions of concepts and relations in them to obtain the building blocks for G . In the final annotation product G , only a fraction of the identified mentions are used, such that most of the mention annotation work is in a sense “wasted” effort but cannot be avoided. The bigger the size difference between D and G is, i.e. the more we want to summarize, the bigger this problem becomes.

To overcome these challenges, we explore two different directions in this thesis. First, we make use of existing datasets that provide annotations for parts of the task and explore different ways of automatically extending them to a full benchmark corpus for CM-MDS. Due to the automation, no manual effort is needed. As a second option, we develop a corpus creation method that effectively combines automatic preprocessing, scalable crowdsourcing and high-quality expert annotations to make the annotation more efficient. We describe both directions and the resulting datasets in the remainder of this chapter.

²⁴It is not available for download anywhere right now and, in personal communication, the authors of the corpus also pointed out that they are not able to change that in the future.

4.2 Automatic Corpus Creation

To create benchmark corpora automatically, we use two different techniques. We make use of a collection of manually created concept maps and match them with corresponding documents, resulting in the BIOLOGY dataset. And second, we use documents with existing concept annotations and extend them to create the WIKI and ACL datasets.

4.2.1 Using Existing Concept Maps

As the starting point for BIOLOGY, we use a collection of 464 concept maps²⁵ that were manually created by experts as teaching materials for biology. They were previously used by Olney et al. (2011) to evaluate their concept map mining technique. Since the maps have been created independently of a specific text, we have to add appropriate source documents to obtain the desired pairs. We match each of the maps automatically with a corresponding article in Wikipedia. Every map is a star-like graph centered around a central concept, e.g. *protein*, and hence has a similar topical focus as an encyclopedic article. We manually correct wrong assignments and make sure that no article is assigned twice.

To ensure that the concept maps indeed cover the same content as the matched article, we apply several additional measures. We prune all concepts from the maps that are not mentioned in the article and completely ignore maps that have fewer than 4 concepts after this step. This reduces the average number of concepts per map from 8.8 to 7.0. On the text side, we remove all sections from the articles that do not mention a concept other than the central concept of the map. Thereby, we remove aspects from the articles that are not covered by the maps, yet, by working on the section level, we still maintain coherent text.

Unfortunately, a similar approach is not possible to ensure extractiveness of the relation labels, as that would have left us with almost no data. Instead, we manually relabel the relations with mentions that occur together with their concepts in a sentence of the article. Out of 1083 relations in the remaining maps, we assigned a new label to 618 (57%) relations, but could not find appropriate mentions for the remaining 465 (43%).

The resulting dataset has 183 pairs of a concept map and a Wikipedia article. Figure 4.1 shows one of the maps that has been paired with the article *Atom*. Note that in this dataset, maps are centered around a focus concept and concept labels are rather short. The example also shows that relations are sometimes not expressed in the most natural way, which is a consequence of ensuring that all labels are mentions in the paired article.

²⁵Available at <http://web.archive.org/web/20120106232123/http://www.biologylessons.sdsu.edu/ta/toc.html> (select *Lessons SemNet* for a specific topic to access the concept maps).

4.2.2 Using Existing Concept Annotations

As the starting point for WIKI, we use a recently published MDS corpus created by Zopf et al. (2016b). They took the introductory sections of featured Wikipedia articles, which tend to be good summaries of the topic due to the Wikipedia guidelines for featured articles, and matched them with web pages that described different aspects covered by the summary in detail. Their corpus consists of 91 pairs of documents D and a textual summary S .

For our corpus, we make use of the fact that these summaries S , being Wikipedia articles, contain many links to other Wikipedia pages. For each S , we create a set of concepts C by collecting the names of the linked articles and the main article itself. Since they are linked in the summary, they tend to be important concepts for the topic. Further, we run an existing OIE system²⁶ over the source documents D to extract binary propositions. To construct a summary concept map G for a document set D , we then iterate over all extractions and identify those that mention a concept out of C in both of their arguments. We apply a range of rules to filter out spurious matches, e.g. concept mentions that are just a small part of a very long argument or extractions containing unresolved pronouns. If an extraction passes all tests, it is added to a set R , forming the concept map $G = (C, R)$.

Since the map G has been created based on the set C derived from S , it covers similar content as the summary and is thus an adequate summary for D . To ensure that it is also connected, as required by Definition 5, we reduce the obtained graph to its biggest connected component. Finally, we remove all pairs where the resulting concept map has fewer than 7 concepts. After these steps, WIKI consists of 38 pairs of documents and a summary concept map. One of them is shown in Figure 4.2.

For the third dataset, ACL, we use the ACL RD-TEC 2.0 corpus (QasemiZadeh and Schumann, 2016). It consists of 300 abstracts taken from papers in the ACL Anthology in which two annotators marked concepts. As abstracts are good summaries of a paper, these concepts tend to be the central concepts discussed in the papers. We use Apache Tika²⁷ to extract the full texts, excluding the abstracts, from the PDF version of the corresponding papers. We filter out papers where the extraction fails. These texts are then paired with the annotated concepts as the gold concepts. We obtain 255 pairs. Note that we cannot use this corpus to evaluate relation extraction, as such annotations are not available.

4.2.3 Comparison and Limitations

All three datasets, compared in Table 4.2, could be created mostly automatic with minimal manual effort, circumventing the challenges of manual annotation that were discussed at the beginning of this chapter. And, compared to most of the existing datasets presented in

²⁶OpenIE4 (Mausam, 2016), a state-of-the-art system according to Stanovsky and Dagan (2016b).

²⁷<https://tika.apache.org/>

Dataset	Pairs	Concept Map				Source	
		Concepts	Tokens	Relations	Tokens	Documents	Tokens
BIOLOGY	183	6.9 ± 4.0	1.2 ± 0.4	3.5 ± 3.0	1.9 ± 1.2	1.0 ± 0.0	2620.9
WIKI	38	11.3 ± 5.2	1.9 ± 0.4	13.8 ± 8.4	5.0 ± 1.2	14.6 ± 3.1	27065.6
ACL	255	10.9 ± 5.5	1.9 ± 0.9	–	–	1.0 ± 0.0	4987.5

Table 4.2: Corpus statistics for automatically created benchmark corpora. All values are averages over pairs with their standard deviation indicated by \pm . ACL does not contain relations.²⁸

Table 4.1, all three are substantially bigger. But Table 4.2 also reveals that the datasets do not yet satisfy all requirements.

Both BIOLOGY and ACL only provide summaries for single documents, whereas we want to have summaries for document sets. In addition, ACL does not have real concept maps, but only concepts, and thus can only be used to evaluate concept mention extraction, but not the full task. BIOLOGY, while having relations, provides only very small concept maps, with especially few relations. Given the average number of relations (3.5) and concepts (6.9) per map, one can also easily see that the graphs are disconnected.

WIKI comes closest to our requirements because it provides multi-document summaries that are bigger and connected concept maps. Its main weakness are the relations, which have been obtained fully automatically. Since no annotator was involved, we do not have a guarantee that they express relationships in the same way a human would. The large size of their labels, compared to BIOLOGY (5.0 vs. 1.9 tokens), indicates that they follow a different style. The example in Figure 4.2 also reveals that some relations are rather complex clauses. During evaluations, this dataset might also unfairly favor CM-MDS approaches that use similar OIE-based techniques for relation extraction.

In light of these limitations, we explore other techniques to create more high-quality datasets with reasonable effort in the next part of this chapter. That being said, we want to emphasize that the automatically created datasets can still be of use in experiments where their limitations are less relevant or if they are taken into account when interpreting quantitative results. In this thesis, we will use the BIOLOGY and ACL datasets to evaluate concept and relation extraction approaches in Section 5.2 and the WIKI dataset as a second corpus to evaluate pipelines for the full task in Chapter 6.

²⁸The values reported here differ slightly from those in (Falke and Gurevych, 2017c, Table 1) where statistics for the test split rather than the whole dataset are shown.

4.3 Manual Corpus Creation

4.3.1 Importance Annotation via Crowdsourcing

A major bottleneck for the manual creation of summary concept maps is the high effort connected with determining the important elements of a document collection. For document sets of around 40 documents, just reading all of them once takes already more than 8 hours.²⁹ And second, the notion of importance, or in other words, the decision of what to include in a summary, can be very subjective. While this is to some extent the nature of the summarization task, we aim to avoid subjectiveness as much as possible to ensure that a data collection can be reliably repeated, even with different annotators, and that it results in consistent annotations that are useful to train and evaluate models.

We explore *crowdsourcing* as a means to perform this task more efficiently and to collect a broad range of individually (potentially) subjective annotations that can be aggregated to a consensus. Crowdsourcing allows us to distribute the work to a large pool of annotators of which each has to do only a small task. Starting with the seminal work of Snow et al. (2008), annotations for many NLP tasks have been successfully collected in the past. Lloret et al. (2013) describe several experiments to crowdsource textual reference summaries. In their study, workers are asked to read 10 documents and then select 10 summary sentences from them for a reward of \$0.05. They discovered several challenges, including poor work quality and the subjectiveness of the annotation task, and conclude that crowdsourcing is not useful for this type of data collection.

To that end, we introduce a new task design, *low-context importance annotation*, to determine summary-worthy parts of documents. Compared to Lloret et al.’s approach, it is more in line with crowdsourcing best practices, as the tasks are simple, intuitive and small (Sabou et al., 2014) and workers receive reasonable payment oriented at the minimum wage (Fort et al., 2011). Most importantly, it is also much more efficient and scalable, as it does not require a single worker to read all documents in a cluster to create a summary.

4.3.1.1 Task Design

We let crowdworkers perform the task of importance annotation per proposition³⁰ rather than full sets of documents. The goal of that data collection is to obtain a score for each proposition that indicates its importance in a document cluster, such that a ranking according to the score would reveal what is most important and should be included in a summary. In contrast to other work, we do not show the documents to the workers at all, but provide

²⁹For the corpus creation, we use sets of 40 documents that have on average 97,880 words. At a typical reading speed of 200 words per minute, it would take 8.16 hours to read them.

³⁰The propositions used here are the same as in a concept map, a triple of concepts and a relation that form a meaningful statement. In Section 4.3.2, we describe how the data for crowdsourcing is generated.

<p>Imagine you want to learn something about student loans without credit history. How useful would the following statements be for you?</p> <p><i>(P1) students with bad credit history - apply for - federal loans with the FAFSA</i> <input type="checkbox"/> Extremely <input type="checkbox"/> Very <input type="checkbox"/> Moderately <input type="checkbox"/> Slightly <input type="checkbox"/> Not at all</p> <p><i>(P2) students - encounter - unforeseen financial emergencies</i> <input type="checkbox"/> Extremely <input type="checkbox"/> Very <input type="checkbox"/> Moderately <input type="checkbox"/> Slightly <input type="checkbox"/> Not at all</p>

Figure 4.3: Likert-scale crowdsourcing task with topic description and two example propositions.

only a description of the document cluster’s topic along with the propositions. This ensures that tasks are small, simple and can be done quickly (see Figure 4.3).

As another crucial aspect of the crowdsourcing task, we try to make it as intuitive as possible. This is important since crowdworkers typically do not bring specific skills for the task, cannot be trained beforehand and also avoid reading lengthy guidelines (Sabou et al., 2014). Instead of trying to define what is meant by “importance”, we therefore embed the annotation task into a real-world scenario, ask the workers to imagine an information need and to then judge the usefulness of a statement in that scenario.

In preliminary tests, we found that this design, despite the minimal context, works reasonably well. As an example, consider Figure 4.3: One can easily say, just given the topic description and the statements, without reading the documents, that P1 is more useful and should rather be in a summary than P2.

We distinguish two variants of the crowdsourcing task:

Likert-Scale Tasks Instead of enforcing binary importance decisions on whether something should be in the summary or not, we use a 5-point Likert-scale to allow more fine-grained annotations. The responses obtained from the interface shown in Figure 4.3 are translated into scores (5...1) and the average of all scores for a proposition is used as an estimate for its importance. This follows the idea that while single workers might find the task subjective, the consensus of multiple workers, represented in the average score, tends to be less subjective due to the “wisdom of the crowd”. We create a single crowdsourcing task by combining five randomly selected propositions from the same document set.

Comparison Tasks As an alternative, we use a second task design based on pairwise comparisons. Comparisons are known to be easier to make and more consistent (Belz and Kow, 2010), but also more expensive, as the number of pairs grows quadratically with the number of objects.³¹ To reduce the cost, we group five propositions into a task and ask workers to order them by importance per drag-and-drop. From that ordering, we derive pairwise

³¹Even with intelligent sampling strategies, such as the active learning in CrowdBT (Chen et al., 2013), the number of pairs is only reduced by a constant factor according to recent experiments by Zhang et al. (2016).

comparisons and use TrueSkill (Herbrich et al., 2007) to obtain importance estimates for each proposition. TrueSkill is a Bayesian model that induces a ranking (and corresponding scores) from pairwise comparisons of a set of elements. In a recent comparison of models for this task, it was found to be among the best performing ones (Zhang et al., 2016).

4.3.1.2 Pilot Experiment

To verify the proposed approach, we conducted a pilot study on Amazon Mechanical Turk³² using data from TAC 2008 (Dang and Owczarzak, 2008). We collected importance estimates for 474 propositions extracted from the first three document sets³³ using both task designs. Each Likert-scale task was assigned to 5 different workers and we paid each of them \$0.06. For comparison tasks, we also collected 5 labels each, paid \$0.05 per task and sampled around 7% of all possible pairwise comparisons. We submitted them in batches of 100 pairs and selected pairs for subsequent batches based on the confidence of the TrueSkill model given the data collected so far.

Quality Control Following the observations of Lloret et al. (2013), we established several measures for quality control. First, we restricted our tasks to workers from the US with an approval rate of at least 95%. Second, we identified low quality workers by measuring the correlation of each worker’s Likert-scores with the average of the other four scores. The worst workers (at most 5% of all labels) were removed. In addition, we included trap sentences, similar as in Lloret et al. (2013), in around 80 of the tasks. In contrast to Lloret et al.’s findings, both an obvious trap sentence (*This sentence is not important*) and a less obvious but unimportant one (*Barack Obama graduated from Harvard Law*) were consistently labeled as unimportant (1.08 and 1.14), indicating that the workers do the task properly.

Agreement and Reliability For Likert-scale tasks, we follow Snow et al. (2008) and calculate agreement as the average Pearson correlation of a worker’s Likert-score with the average score of the remaining workers.³⁴ This measure is less strict than exact label agreement and can account for close labels and high-scoring or low-scoring workers. We observe a correlation of 0.81, indicating substantial agreement. For the comparison tasks, the majority agreement, i.e. the fraction of the collected pairwise preferences that agree with the majority decision per item, is 0.73. To further examine the reliability of the collected data, we followed the approach of Kiritchenko and Mohammed (2016) and simply repeated the crowdsourcing for one of the three topics. Between the importance estimates calculated from the first and second run, we found a Pearson correlation of 0.82 (Spearman 0.78) for

³²<https://www.mturk.com/>

³³D0801A-A, D0802A-A, D0803A-A

³⁴Because workers are not the same across all items, we create five meta-workers by sorting the labels per proposition.

Peer Scoring	Pearson	Spearman
Modified Pyramid	0.4587	0.4676
ROUGE-2	0.3062	0.3486
Crowd-Likert	0.4589	0.4196
Crowd-Comparison	0.4564	0.3761

Table 4.3: Correlation of manual responsiveness scores on TAC2008 topics 01-03 with ROUGE, Pyramid and peer summary scores derived from crowdsourced proposition importance.

Likert-scale tasks and 0.69 (Spearman 0.66) for comparison tasks. This shows that the approach, despite the subjectiveness of the task, allows us to collect reliable annotations.

Peer Evaluation In addition to the reliability studies, we extrinsically evaluated the annotations in the task of summary evaluation. For each of the 58 peer summaries of participants in TAC 2008, we calculated a score as the sum of the importance estimates of the propositions it contains. Table 4.3 shows how these peer scores, averaged over the three topics, correlate with the manual responsiveness scores assigned during TAC in comparison to ROUGE-2 and Pyramid scores.³⁵ The results demonstrate that with both task designs, we obtain importance annotations that are similarly useful for summary evaluation as pyramid annotations or gold-standard summaries (used for ROUGE).

Based on the pilot study, we conclude that the proposed crowdsourcing setup allows us to obtain reliable importance annotations. Since workers are not required to read all documents, the annotation is much more efficient and scalable as with traditional methods.

4.3.2 Scalable Manual Corpus Creation

This section presents our new corpus creation process, as outlined in Figure 4.4, which combines the scalable crowdsourcing setup described in the previous section with automatic preprocessing techniques and high-quality expert annotations. Using it, summary concept maps can be created even for larger document sets with much less effort than in the fully manual annotation process. For every document of the corpus we created, we spent about \$150 on crowdsourcing and 1.5h of expert annotations, while just a single annotator would already need over 8 hours to read all documents of a topic once.

As a starting point, we used the DIP corpus (Habernal et al., 2016a), a collection of 49 clusters of 100 web pages on educational topics (e.g. bullying, homeschooling, drugs) with a

³⁵Correlations for ROUGE and Pyramid are lower than reported in TAC because we used only 3 instead of all 48 topics in the pilot study.

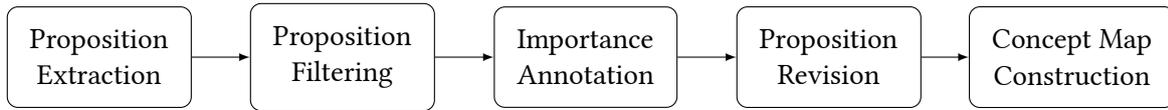


Figure 4.4: The five-step process of our scalable manual corpus creation approach.

short description of each topic. It was created from a large web crawl using state-of-the-art information retrieval. We selected 30 of the topics for which we created summary concept maps as described in the following sections.

4.3.2.1 Proposition Extraction

Since concept maps consist of propositions expressing the relation between concepts, we need to impose such a structure upon the plain text in the document sets. This could be done by manually annotating spans mentioning concepts and relations, however, the size of our document sets makes this a huge effort: 2,288 sentences per topic (69k in total) need to be processed. Therefore, we resort to an automatic approach.

As we pointed out earlier, OIE can extract tuples from sentences that are already very similar to propositions in a concept map. For instance, from

Students with bad credit history should not lose hope and apply for federal loans with the FAFSA.

an OIE system can extract the following tuples

(students with bad credit history - should not lose - hope)
(students with bad credit history - apply for - federal loans with the FAFSA)

While the relation is similar to a relation in a concept map, many arguments in these tuples represent useful concepts. For the corpus creation, we used OpenIE4 (Mausam, 2016) to automatically process all sentences of the 30 document sets. After removing duplicates, we obtained 4137 unique extractions per topic. Since we want to create a gold-standard corpus, we have to ensure that we produce high-quality data. We therefore made use of the confidence assigned to every extraction to filter out low quality ones. To ensure that we do not filter too aggressively (and miss important aspects in the final summary), we manually annotated 500 tuples sampled from all topics for correctness. On the first 250 of them, we tuned the filter threshold to 0.5, which keeps 98.7% of the correct extractions in the unseen second half. After filtering, a topic had on average 2,850 propositions (85k in total).

4.3.2.2 Proposition Filtering

Despite the similarity of OIE extractions, not every extracted tuple is a suitable proposition for a concept map. To reduce the effort in the subsequent steps, we therefore want to filter out unsuitable ones. A tuple is only suitable if it

- (1) is a correct extraction,
- (2) is meaningful without any context and
- (3) has arguments that represent proper concepts.

We created a guideline explaining when to label a tuple as suitable for a concept map and performed a small annotation study. Three annotators independently labeled 500 randomly sampled tuples. The agreement was 82% ($\kappa = 0.60$). We found tuples to be unsuitable mostly because they had unresolvable pronouns, conflicting with (2), or arguments that were full clauses or propositions, conflicting with (3), while (1) was mostly taken care of by the confidence filtering in the previous step.

Due to the high number of tuples we decided to automate the filtering step. We trained a linear support vector machine (SVM) using the majority vote of the annotations as supervision. As features, we used the extraction confidence, length of arguments and relations as well as part-of-speech tags. To ensure that the automatic classification does not remove suitable propositions, we tuned the classifier to avoid false negatives. In particular, we introduced class weights, improving precision on the negative class at the cost of a higher fraction of positive classifications. Additionally, we manually verified a certain number of the most uncertain negative classifications to further improve performance. When 20% of the classifications are manually verified and corrected, we found that our model trained on 350 labeled instances achieves 93% precision for negative classifications on the unseen 150 instances. We found this to be a reasonable trade-off of automation and data quality and applied the model to the full dataset.

The classifier filtered out 43% of the propositions, leaving 1,622 per topic. We manually examined the 17k least confident negative classifications and corrected 955 of them. We also corrected positive classifications for certain types of tuples for which we knew the classifier to be imprecise. Finally, each topic was left with an average of 1,554 propositions suitable to be part of a concept map (47k in total).

4.3.2.3 Importance Annotation

Given the propositions identified in the previous step, we now applied our crowdsourcing scheme as described in Section 4.3.1 to determine their importance. To cope with the large number of propositions, we combine the two task designs: First, we collect Likert-scores from five workers for each proposition, clean the data and calculate average scores. Then,

using only the top 100 propositions³⁶ according to these scores, we crowdsource 10% of all possible pairwise comparisons among them. Using TrueSkill, we obtain a fine-grained ranking of the 100 most important propositions.

For Likert-scores, the average agreement over all topics is 0.80, while the majority agreement for comparisons is 0.78. We repeated the data collection for three randomly selected topics and found the Pearson correlation between both runs to be 0.73 (Spearman 0.73) for Likert-scores and 0.72 (Spearman 0.71) for comparisons. These figures show that the crowdsourcing approach works on this dataset as reliably as on the TAC documents.

In total, we submitted 53k scoring and 12k comparison tasks to Amazon Mechanical Turk for a price of \$4,425.45 including fees to perform the annotation. From the fine-grained ranking of the 100 most important propositions, we select the top 50 per topic to construct a summary concept map in the subsequent steps.

4.3.2.4 Proposition Revision

Having a manageable number of propositions, an annotator then applied a few straightforward transformations that correct common errors of the OIE system. First, we break down propositions with conjunctions in either of the arguments into separate propositions per conjunct, which the OIE system sometimes fails to do. And second, we correct span errors that might occur in the argument or relation phrases, especially when sentences were not properly segmented. As a result, we have a set of high quality propositions for our concept map, consisting of, due to the first transformation, 56.1 propositions per topic.

4.3.2.5 Concept Map Construction

In the final step, we connect the set of important propositions to form a graph. For instance, given the following two propositions

(student - may borrow - Stafford Loan)
(the student - does not have - a credit history)

one can easily see, although the first argument differs slightly, that both labels describe the concept *student*, which allows us to build a concept map with the concepts *student*, *Stafford Loan* and *credit history*. The annotation task thus involves deciding which of the available propositions to include in the map, which of their concepts to merge and, when merging, which of the labels to use. As these decisions highly depend upon each other and require context, we decided to use in-house annotators rather than crowdsource the subtasks.

Annotators were given the topic description and the most important, ranked propositions. Using a simple annotation tool providing a visualization of the graph, they could

³⁶We also add all propositions with the same score as the 100th, yielding 112 propositions on average.

Dataset	Pairs	Source Documents			
		Tokens	Documents	Tokens/Doc	Rel. Std.
EDUC	30	97,880 \pm 50,086.2	40.5 \pm 6.8	2,412.8 \pm 3,764.1	1.56
DUC 2006	50	17,461 \pm 6,627.8	25.0 \pm 0.0	729.2 \pm 542.3	0.74
DUC 2004	50	6,721 \pm 3,017.9	10.0 \pm 0.0	672.1 \pm 506.3	0.75
TAC 2008A	48	5,892 \pm 2,832.4	10.0 \pm 0.0	589.2 \pm 480.3	0.82

Table 4.4: Source documents of EDUC in comparison to classic MDS evaluation datasets. All values are averages over pairs with their standard deviation indicated by \pm . Rel. Std. shows the standard deviation of tokens per document divided by average document length to provide a measure of how much the document length varies independent of the typical document size.

connect the propositions step by step. They were instructed to reach the size of 25 concepts, the recommended maximum size for a concept map (Novak and Cañas, 2007). Further, they should prefer more important propositions and ensure connectedness. When connecting two propositions, they were asked to keep the concept label that was appropriate for both propositions. To support the annotators, the tool used ADW (Pilehvar et al., 2013), a method to compute semantic similarity, to suggest possible connections. The annotation was carried out by graduate students with a background in NLP after receiving an introduction into the guidelines and tool and after annotating a first example. If an annotator was not able to connect 25 concepts, they were allowed to create up to three synthetic relations with freely defined labels, making the maps slightly abstractive. On average, the constructed maps have 0.77 synthetic relations, mostly connecting concepts whose relation is too obvious to be explicitly stated in text (e.g. between *Montessori teacher* and *Montessori education*).

To assess the reliability of this annotation step, we had the first three maps created by two annotators. We casted the task of selecting propositions to be included in the map as a binary decision task and observed an agreement of 84% ($\kappa = 0.66$). Second, we modeled the decision which concepts to join as a binary decision on all pairs of common concepts, observing an agreement of 95% ($\kappa = 0.70$). And finally, we compared which concept labels the annotators decided to include in the final map, observing 85% ($\kappa = 0.69$) agreement. Hence, the annotation shows substantial agreement (Landis and Koch, 1977).

4.3.3 Corpus Analysis and Experiments

In this section, we describe the newly created corpus, EDUC. In addition to having summaries in the form of concept maps, it differs from traditional summarization corpora in several aspects that make it both challenging and unique.

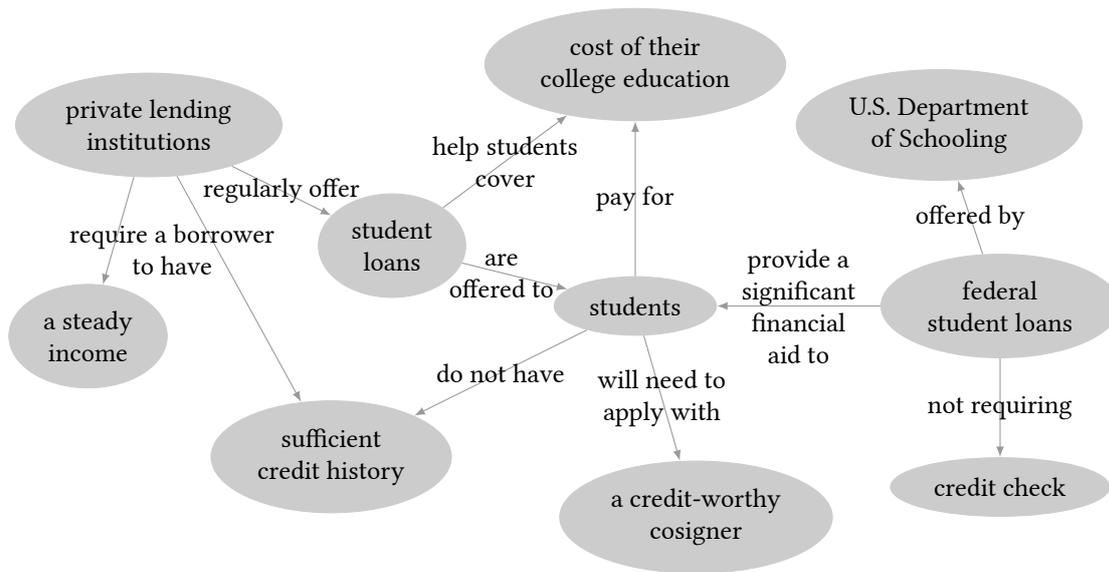


Figure 4.5: Excerpt from a summary concept map from EDUC for the topic “*students loans without credit history*”. The full map has 25 concepts and 28 relations.

4.3.3.1 Source Documents

The corpus consists of document sets for 30 different topics. Each of them contains around 40 documents with on average 2,413 tokens, which leads to an average cluster size of 97,880 tokens. With these characteristics, the document sets are 15 times larger than typical DUC clusters of ten documents and five times larger than the 25-document-clusters (see Table 4.4). In addition, the documents are also more variable in terms of length, as the (length-adjusted) standard deviation is twice as high as in the other corpora. With these properties, the corpus represents an interesting challenge towards real-world application scenarios, in which users typically have to deal with many more than ten documents.

Because we used a large web crawl as the source for the corpus, it contains documents from a variety of genres. To further analyze this property, we categorized a sample of 50 documents. Among them, we found professionally written articles and blog posts (28%), educational material for parents and kids (26%), personal blog posts (16%), forum discussions and comments (12%), commented link collections (12%) and scientific articles (6%). This makes the corpus particularly challenging, as extraction techniques cannot rely on genre-specific properties such as the fact that in news articles, the most important content tends to be at the beginning and the least important at the end of the document.

In addition to the variety of genres, the documents also differ in terms of language. To capture this property, we follow Zopf et al. (2016b) and compute, for every topic, the average Jensen-Shannon divergence between the word distribution of one document and the word distribution in the remaining documents. The higher this value is, the more the language differs between documents. We found the average divergence over all topics to

Part-of-Speech	Concepts		Relations	
	Heads	Tokens	Heads	Tokens
NOUN	81.8	49.3	5.3	10.5
VERB	14.5	8.7	93.8	50.8
ADJ	2.9	14.5	0.1	4.9
ADP	0.1	9.5	0.8	16.1
ADV	0.1	2.1	–	5.4
DET	–	6.0	–	2.9
CONJ	–	3.4	–	0.6
PNCT	–	3.0	–	0.2
PRT	–	1.6	–	7.9
other	0.6	1.9	–	0.7

Table 4.5: Part-of-speech distribution in concept and relation labels of EDUC, shown among all tokens and just head tokens. Part-of-speech according to the universal tagset (Petrov et al., 2012).

be 0.3490, whereas it is 0.3019 in DUC 2004 and 0.3188 in TAC 2008A. Again, this indicates that the new corpus is challenging and requires dealing with more diverse language.

4.3.3.2 Summary Concept Maps

Each of the 30 reference concept maps has 25 concepts and between 24 and 28 relations. Labels have on average of 3.2 tokens. Figure 4.5 shows a subset of one of the maps.

To obtain a better impression of what kind of text spans have been used as labels, we automatically tagged them with their part-of-speech and determined their head with a dependency parser.³⁷ Results are shown in Table 4.5. Concept labels tend to be headed by nouns (82%) or verbs (15%), while they also contain adjectives, prepositions and determiners. Relation labels, on the other hand, are almost always headed by a verb (94%) and contain prepositions, nouns and particles in addition. These distributions are very similar to those reported by Villalon et al. (2010) for their (single-document) concept map corpus.

Analyzing the graph structure of the maps, we found that, in line with the instructions given to annotators, all of them are connected graphs. They have on average 7.2 central concepts with more than one relation, while the remaining ones occur in only one proposition. During the annotation, we found that achieving a higher number of connections would mean compromising importance, i.e. including less important propositions, and decided against it to maintain the summary aspect of the maps.

³⁷Tagging and dependency parsing was performed with spaCy v1.3.0 (<https://spacy.io>).

Metric	Precision	Recall	F1-Score
METEOR	15.12	19.49	17.00
ROUGE-2	6.03	17.98	8.91

Table 4.6: Performance of the baseline on the EDUC test set.

4.3.3.3 Baseline Experiments

Along with the corpus creation, we implemented a first baseline approach for CM-MDS. It is a pipeline that approaches each of the subtasks with a simple method inspired by previous work on concept map mining. For a document set D , it performs the following steps:

- (1) Extract all noun phrases as concept mentions M .
- (2) Group mentions whose labels match after stemming to unique concepts C .
- (3) For each pair of concepts whose mentions co-occur in a sentence, select the tokens in between as a relation mention if they contain a verb. This directly yields R , as no explicit relation grouping is done.
- (4) If a pair of concepts has more than one relation, select the one with the shortest label. This subset $\hat{R} \subseteq R$ is used to build the graph $G = (C, \hat{R})$.
- (5) Assign an importance score $i(c)$ to every concept in C .
- (6) Find a connected subgraph of G that has no more than \mathcal{L}_C concepts with high scores.

For (5), we train a binary classifier to identify the important concepts in the set of all potential concepts. We use common features for summarization, including position, frequency and length, and Weka’s Random Forest (Hall et al., 2009) implementation as the model. At inference time, we use the classifier’s confidence for a positive classification as the importance score $i(c)$ for a concept.

In step (6), we start with the full graph G and use a heuristic to find a subgraph that is connected, satisfies the size limit \mathcal{L}_C and has many high-scoring concepts: We iteratively remove the lowest-scoring concept until only one connected component of \mathcal{L}_C concepts or less remains, which is used as the summary concept map. This approach guarantees that the concept map is connected, but due to its heuristic nature, it might not find the subset of concepts that has the highest total importance score while being connected.

For the experiment, we used a 50:50 split of the EDUC corpus, using one half for development and training the random forest model and the other half for evaluation. Table 4.6 shows the performance of the baseline, evaluated using the METEOR and ROUGE metrics introduced in Section 3.5.2 and the relation-unrestricted setting with $\mathcal{L}_C = 25$, the size of the reference concept maps. In terms of METEOR, the baseline achieves an F1-score of 17%, while it is around 9% for ROUGE-2. For traditional summarization tasks and corresponding corpora, state-of-the-art performance is at around 24% ROUGE-2 recall for SDS on DUC

2002 (Al-Sabahi et al., 2018), at 19% on DUC 2002 and 10% on DUC 2003 for MDS (Peyrard and Eckle-Kohler, 2016) and, measured in ROUGE-2 F1-score, at 19% on CNN/DailyMail and 31% on the New York Times corpus (Celikyilmaz et al., 2018), which are both also SDS corpora. A comparison across datasets, as already illustrated by the performance differences between the different traditional summarization corpora, is not very meaningful. A comparison of our baseline results against these state-of-the-art performances is therefore of only very limited value, as not just the data, but also the task and evaluation protocols differ. Nevertheless, it at least indicates that the task represented by our dataset is neither trivially simple nor extremely hard, but in the range of existing summarization work.

A detailed analysis of the single pipeline steps revealed that there is room for improvement in all the pipeline steps. First, we observed that only around 76% of gold concepts are covered by the extracted mentions (step 1) and after grouping concepts (step 2), showing that better extraction methods are needed. For relations, the recall is considerably lower. After estimating the importance of concepts (step 5), the top 25 concepts contain only 17% of the gold concepts. Hence, content selection is a major challenge, stemming from the large cluster sizes in the corpus. The propagation of errors along the pipeline further contributes to low performance. Overall, the baseline experiments confirm that the task is complex and cannot be solved with simple techniques.

4.4 Chapter Summary

In this chapter, we addressed the lack of suitable corpora to train and evaluate computational models for CM-MDS. We briefly reviewed the limited amount of datasets that have been used for concept map mining in the past and showed that they cannot be used for our purposes. While this shows that the creation of a new corpus is inevitable, we also pointed out that the fully manual annotation of the required summary concept maps is a very time-consuming and complex effort. As alternatives, we explored two different directions: the automatic extension of existing partial annotations and the use of a new scalable annotation process that relies on crowdsourcing.

With regard to the first direction, we showed how existing concept maps can be used by matching them with corresponding documents to obtain the necessary pairs for evaluation, leading to the BIOLOGY corpus. As a second approach, we used documents with existing annotations for concept mentions from two datasets based on Wikipedia and the ACL Anthology to derive additional evaluation data. The WIKI and ACL corpora are the results of these efforts. While these approaches led to corpora of a reasonable size with almost no manual effort, the resulting data does not fulfill all of our requirements. BIOLOGY and ACL provide only single-document summarization scenarios, with ACL also lacking relation annotations, whereas WIKI covers the full task but is not of a high enough quality to serve as a reference for human-level performance on the task.

Dataset	Pairs	Concept Map				Source	
		Concepts	Tokens	Relations	Tokens	Documents	Tokens
EDUC	30	25.0 ± 0.0	3.2 ± 0.5	25.2 ± 1.3	3.2 ± 0.5	40.5 ± 6.8	97880.0
BIOLOGY	183	6.9 ± 4.0	1.2 ± 0.4	3.5 ± 3.0	1.9 ± 1.2	1.0 ± 0.0	2620.9
WIKI	38	11.3 ± 5.2	1.9 ± 0.4	13.8 ± 8.4	5.0 ± 1.2	14.6 ± 3.1	27065.6
ACL	255	10.9 ± 5.5	1.9 ± 0.9	–	–	1.0 ± 0.0	4987.5

Table 4.7: Corpus statistics for all benchmark corpora used in the thesis. Values for BIOLOGY, WIKI and ACL are the same as in Table 4.2 and are repeated for easy comparison to EDUC.

As the second direction, we developed a new corpus creation method that effectively combines automatic preprocessing, scalable crowdsourcing and high-quality expert annotations. Its crucial component is a novel crowdsourcing scheme called low-context importance annotation. In contrast to traditional approaches, it allows us to determine important elements in a large document set without requiring annotators to read all documents, making it feasible to crowdsource the task and overcome quality issues observed in previous crowdworking attempts. We showed that the approach creates reliable data for our summarization scenario and, when tested on traditional summarization corpora, creates annotations that are similar to those obtained by earlier data collection efforts. Using this new corpus creation method, we can avoid the high effort for annotators, which allows us to scale to document sets that are 15 times larger than in traditional summarization corpora. We created a new corpus, EDUC, with 30 topics, each with around 40 source documents on educational topics and a summarizing concept map that is the consensus of many workers.

Table 4.7 summarizes the corpora created in this chapter. All of them will be used throughout the remainder of this thesis. While EDUC, offering the highest-quality annotations, will be the main evaluation corpus, we will use the automatically created BIOLOGY and ACL corpora for concept and relation extraction experiments in Section 5.2 and the WIKI corpus as a second dataset for the task-level experiments carried out in Chapter 6.

CHAPTER 5

Concept and Relation Extraction

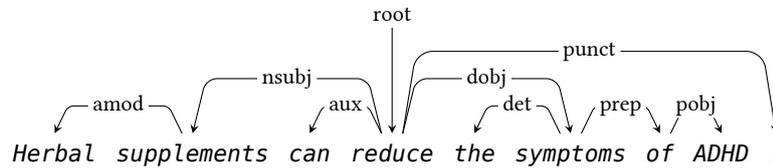
In this chapter, we will focus on the CM-MDS subtasks of concept and relation mention extraction. Using the datasets introduced in the previous chapter, we will present a series of experiments that, for the first time, directly compare different extraction approaches proposed in previous work. Moreover, we will introduce the idea of using predicate-argument analysis for concept and relation mention extraction and include such methods in the experimental comparison. And finally, as most work on concept maps in the past has focused on the English language, we will dedicate the second part of the chapter to studying how such extraction methods can be ported to other languages.

5.1 Motivation and Challenges

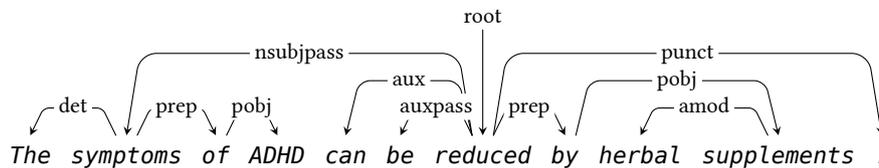
In Section 3.3.1 and Section 3.3.3, we outlined the challenges of concept and relation mention extraction: the variety of expressions that can be used to mention concepts or relations in text, the need to find a good trade-off between precision and recall when trying to cover that variety and the desire to design extraction methods that work well across different types of text. A range of extraction techniques has been proposed for this subtask, which we reviewed in Section 2.3.1.1 and Section 2.3.1.2. They all use a syntactic representation of the input text – in the form of part-of-speech tags, constituency parse trees or dependency parses – and extract sequences of tokens that follow certain patterns. Corresponding sets of patterns have been hand-designed to cover the targeted mentions.

We also mentioned before that a weakness of previous work is the lack of comparative experiments, leaving it unclear which of the proposed extraction methods perform best. As the first contribution of this chapter, we therefore carry out such a comparison by reimplementing representative approaches from previous work and evaluating them on the new corpora introduced in the previous chapter. This will provide valuable insights into the performance of the different approaches and open issues that still have to be addressed.

As a second contribution, we propose to design mention extraction methods based on predicate-argument structures instead of dependency representations. To illustrate this idea, consider the following example and its syntax as given in a dependency tree:



To extract the concept mentions *Herbal supplements* and *the symptoms of ADHD*, patterns extracting *nsubj*- and *dobj*-dependencies can be used to find the relevant spans of tokens in the dependency representation. However, these patterns cannot extract anything from the passive variant of the sentence because the tokens now have other grammatical functions:



Additional patterns would be necessary to identify the same concept mentions in the passive sentence. Due to the variety of natural language, such pattern-based approaches are either limited in coverage or require a very large and carefully designed set of patterns to cover every possible way in which propositions of concepts and relations can be expressed.

To eliminate the high effort associated with the pattern definition, we propose to utilize predicate-argument structures instead of more fine-grained representations such as dependency trees because the former already abstract away from many syntactic variations. Continuing the example, a representation that simply marks *reduce* as a predicate and *herbal supplements* and *the symptoms of ADHD* as its arguments would be desirable. Being independent of a specific realization, it would be an appropriate representation of both the active and passive syntactic variant of the example, requiring no separate handling of the cases. Using such a unified representation based on predicates and arguments, mention extraction approaches no longer need to carefully define large sets of patterns, but can instead make use of existing predicate-argument analysis tools.

Finally, as the third contribution of this chapter, we address the language dependency of concept and relation extraction methods. Most previous work on concept map mining focused on text in English and designed extraction patterns specifically for the syntax of English. Unfortunately, also the predicate-argument analysis tools we propose as an alternative are mainly focused on English. To gain more insight into how difficult and costly it is to also create predicate-argument analysis tools for other languages, we present a case-study of porting an existing system to German. We discuss the different challenges of such

a porting approach that arise from differences between the source and target languages and evaluate the proposition extraction performance of the resulting system for German.

5.2 Extraction with Predicate-Argument Analysis

To assess the usefulness of predicate-argument analysis for concept and relation mention extraction, we test three different representations: PropBank semantic role labeling (SRL) (Hajič et al., 2009), PropS (Stanovsky et al., 2016b) and OIE (see Section 2.3.3). This selection is motivated by the fact that the three representations are examples for different degrees of abstraction from dependency trees: OIE identifies predicates and arguments in the dependency tree, PropS additionally unifies a certain number of syntactic variations and PropBank SRL maps tokens to a syntax-independent lexicon of predicates and arguments.

In the next section, we describe in detail how we derive concept and relation mentions from the three predicate-argument representations. In the subsequent section, we then present experiments that compare extraction performance between these representations and concept map mining approaches from previous work.

5.2.1 Predicate-Argument Structures

We will demonstrate how predicate-argument structures can be leveraged for mention extraction using sentence (1) from the example in Figure 3.1:

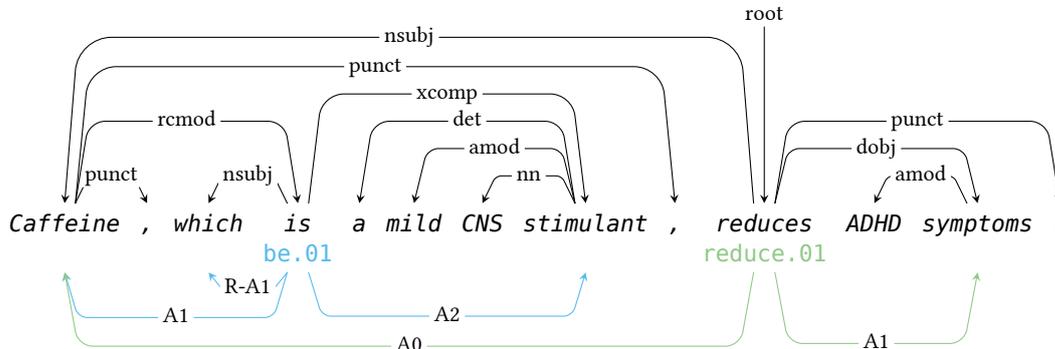
Caffeine, which is a mild CNS stimulant, reduces ADHD symptoms.

If we have a representation that identifies *reduces* as a predicate and *Caffeine* and *ADHD symptoms* as the arguments of that predicate, this predicate-argument-triple could be directly used as a proposition in a concept map. The central idea is that there is a direct correspondence between a predicate and its arguments and a relation and its concepts in a concept map. Thus, we simply use all identified predicates as relation mentions and their arguments as concept mentions. The only restriction is that we need to obtain tuples of a predicate with exactly two arguments from the predicate-argument structure because relations (and the resulting propositions) in a concept map are always binary.

5.2.1.1 Semantic Role Labeling

PropBank SRL takes as input a sentence and its dependency parse tree. It then maps all verbs in the sentence to their correct sense in the PropBank lexicon (Palmer et al., 2005) and identifies the head tokens of their arguments. We prefer PropBank SRL (Palmer et al., 2005) over FrameNet and VerbNet because of its robustness and maturity. The lexicon defines which arguments (fulfilling which roles) a specific verb can have, but not all of them are

necessarily present in a sentence. For our example sentence, the resulting representation looks like this:



Arcs above the tokens show syntactic dependencies. Below the tokens, the two sense-disambiguated predicates *be.01* and *reduce.01* are shown with arcs pointing to the heads of their arguments and role names describing the arguments' function. A small set of role names, such as *A1* and *A2*, exist in PropBank. They can have slightly different meanings for different predicates which are defined in the lexicon. For *reduce.01*, the *A0*-argument describes the agent and the *A1*-argument the logical subject.

This approach is the most sophisticated type of analysis in our study, as it tries to map a natural language sentence to another layer of abstract semantic representation defined by the predicate and argument role inventory. Because that inventory is independent of specific syntactic realizations, predicates and arguments are always mapped to the same symbols. For instance, in both *Caffeine reduces ADHD* and *ADHD is reduced by caffeine*, the predicate *reduce.01* is found and has an argument *caffeine* with role *A0*. The different realizations of the same underlying proposition are thus unified in the SRL representation, allowing us to handle both cases with the same extraction logic. However, the SRL representation also has disadvantages: First, spans of roles are strictly bound to full subtrees in the dependency parse, as shown in the example, where the relative clause becomes part of the *A0*-argument of the predicate *reduce*, although the proposition would also be valid without that part. And second, predicates and arguments that are not covered by the lexicon cannot be represented and thus can also not be extracted for a concept map.

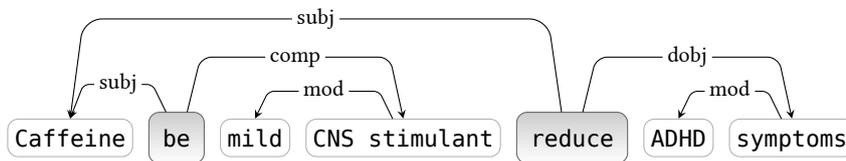
For our experiments, we use the SRL functionality of Mate Tools (Björkelund et al., 2009), a freely available system that was one of the best in the CoNLL 2009 shared task (Hajič et al., 2009). To obtain binary predicates from the representation, we ignore predicates that have just one argument, and if more than two are present, we form propositions with all pairs of them. We further ignore referential arguments (e.g. role *R-A1*), as only a direct mention of an argument is useful for a concept map. For the example sentence, this strategy yields the following propositions:

- (1) (*Caffeine - is - a mild CNS stimulant*)
- (2) (*Caffeine, which is a mild CNS stimulant, - reduces - ADHD symptoms*)

5.2.1.2 PropS

As the second approach, we use PropS, a rule-based converter that turns dependency trees into typed predicate-argument graphs (Stanovsky et al., 2016b). In addition to identifying predicates and arguments, it also canonicalizes the representation of propositions, e.g. by unifying variations such as active and passive or copula and appositive constructions. However, compared to SRL, PropS works exclusively on the lexical level and does not map tokens to any symbols from an external inventory. That has the advantage that the representation can always cover the full content of a given sentence, as opposed to SRL. PropS also classifies predicate-argument relations into a small set of labels such as *subj* and *dobj*.

For the example sentence, PropS yields the following graph representation:



Here, it identified two predicates (dark nodes) that both have two arguments with different roles (*subj*, *dobj* and *comp*). Similar to SRL, arguments can be trees of several tokens, but due to the transformations applied to the dependency tree, its original structure is typically largely simplified. In the example, punctuation and determiners have been dropped and the noun compound has been collapsed into a single graph node. In the graph, predicates are represented by their lemmas, but the original surface form is stored for later use.

To create binary predicates, we traverse the graph from every predicate node and select as its arguments the subgraphs of the directly connected argument nodes. We remove unary predicates and break down higher-arity predicates by creating all possible pairs except if they have the same edge label (e.g. two objects). As mentions, we use the original surface forms of predicates and arguments. We obtain the following propositions for the example:

- (1) (Caffeine - is - a mild CNS stimulant)
- (2) (Caffeine - reduces - ADHD symptoms)

5.2.1.3 Open Information Extraction

As the third approach for predicate-argument analysis, we use OIE. As we already described in Section 2.3.3, OIE systems extract tuples that represent propositions from a given sentence. Every tuple consists of a relation phrase and two arguments, which is already exactly the representation in which we need concept and relation mentions. In contrast to the other two approaches, OIE systems in general do not create an intermediate representation of a specific type. However, they still serve our purpose as the direct creation of propositions

also alleviates the need for mention extraction approaches to deal with different syntactic realizations when looking for concepts and relations.

In our experiments, we use OpenIE4 (Mausam, 2016). It is one of the state-of-the-art OIE systems (Stanovsky and Dagan, 2016b). Using it, the extracted propositions for the example sentence are the same as for PropS:

- (1) *(Caffeine - is - a mild CNS stimulant)*
- (2) *(Caffeine - reduces - ADHD symptoms)*

With regard to the motivation of avoiding laborious definitions of large sets of rules, we note in passing that PropS as well as many OIE systems do indeed make their extractions using hand-written rules. Thus, relying on them for concept and relation mention extraction does not completely remove the need for rules, instead, it shifts the responsibility for rule creation from researchers working on the specific task of concept and relation extraction to the authors of more generally applicable predicate-argument analysis tools.

5.2.2 Experiments

We conduct several experiments to study the usefulness of the three predicate-argument analysis tools in comparison to the techniques specifically developed for concept maps.

5.2.2.1 General Experimental Setup

For the experiments, we use EDUC, our high-quality dataset, as the main evaluation corpus. We also run additional evaluations on BIOLOGY and ACL for concept extraction and BIOLOGY for relation extraction. Being based on Wikipedia articles and scientific papers, they add additional text genres that allow us to assess how well extraction approaches work across domains and genres. As WIKI is from a similar domain as BIOLOGY and of lower size, we do not include it. We use the standard 50:50 split of the EDUC corpus and a 10:90 split into development and test set for the other two. Since no tuned or supervised models are used in these experiments, we only use the training/development sets to verify the implementations and report results on the respective test sets.

For all experiments, we preprocess the documents with DKPro Core (Eckart de Castilho and Gurevych, 2014)³⁸, using tokenization, part-of-speech tagging and constituency parsing from the Stanford NLP tools and Snowball for stemming. Constituency parse trees are converted into collapsed and propagated dependencies (de Marneffe and Manning, 2008), to which we will refer as dependency graphs since this version of dependencies is not guaranteed to be a tree. These preprocessing steps are shared by all approaches that make use of such annotations, while OpenIE4 uses its own internal preprocessing.

³⁸DKPro Core version 1.8.0 (<https://dkpro.github.io/dkpro-core/>).

5.2.2.2 Reimplementation of Previous Work

From the previous work reviewed in Section 2.3.1, we selected and reimplemented³⁹ several representative examples for concept and relation mention extraction techniques. As the descriptions of these methods in the respective papers are often missing relevant details, we had to make a few assumptions on how exactly a method should work. To ensure full reproducibility, we document the exact implementation used for the experiments here.

Valerio et al. (2006) As an example for extraction approaches that rely on constituency parse trees, we reimplement Valerio and Leake (2006). Following them, we extract as concept mentions all NP-constituents that do not contain any smaller NPs and that have at least one token tagged as a noun (tag N*) or adjective (tag J*). Their proposed approach for relation extraction is unfortunately less clear: they write that they extract a relation for “all pairs of concepts that have an indirect dependency link through a verb phrase.” Our specific implementation of that idea looks for NPs followed by a VP containing another NP, where both NPs have previously been identified as concept mentions. In that case, the tokens from the beginning of the VP until the start of the inner NP form the relation mention.

Qasim et al. (2013) We reimplement the method proposed by Qasim et al. (2013) as an example for dependency-based extraction, as they provide the most comprehensive description of their work. For concept mention extraction, we implement Algorithm 1 of their paper (Qasim et al., 2013). It extracts mentions consisting of two tokens that are connected by *nn* (noun compound) or *amod* (adjectival modifier) dependencies and single tokens that are the child of a *nsubj* (noun subject) dependency. In addition, for the two collapsed dependencies *conj_and* (conjunction) and *prep_of* (preposition), both tokens with the additional conjunction or preposition in between are extracted.

For relation extraction, we follow Qasim et al. (2013) and extract all verbs that are parent tokens of a *nsubj*, *nsubjpass*, *xcomp* or *rmod* dependency, including preceding auxiliaries (*aux*, *auxpass*). Then, we determine pairs of concept mentions that co-occur in a sentence with one of the verbs. As an additional criterion, one of the concept mentions has to act as a subject (child token of a **subj** dependency) and the other as an object (child of a **obj** or parent of a *cop* dependency). If all these conditions are satisfied, the verb is used as a relation mention for that pair of concept mentions.⁴⁰

³⁹There is no previous work that makes their implementation available.

⁴⁰If several verbs co-occur with a pair of concepts and all occurrences fulfill the subject-object-criteria, we select among them by frequency, using a VF-ICF metric that favors verbs occurring often but not with many other concepts. We refer to Qasim et al. (2013) for the exact formula. For our experiments, this detail is less relevant as such a selection is rarely necessary. We also note that the full approach by Qasim et al. (2013) uses an additional complex clustering step that further restricts the number of pairs of concepts that are considered for relations at all. It is not part of our reimplementation.

Villalon (2012) Villalon (2012) proposed a different method to extract mentions from dependency parses that we include in our experiments as well. Given the dependency graph, the method first applies the following operations that merge or remove graph nodes:

- If two nodes A and B are connected with an *amod*, *nn*, *number* or *num* dependency and they are directly adjacent in the text, merge them into a single node $A B$.
- If two nodes A and B are connected with a *advmod*, *aux* or *auxpass* dependency, they are directly adjacent in the text and their part-of-speech tags are VB^* , merge them into a single node $A B$.
- If two nodes A and B are connected with a *conj_and* or *prep_of* dependency, they have a distance of 2 in the text and their part-of-speech tags are NN^* , merge them into a single node $A \text{ and } B$ or $A \text{ of } B$.
- If a node has a *det* dependency to its parent, remove it.

After this set of transformations, all nodes that have at least one NN^* part-of-speech tag are extracted as concept mentions. Note that due to the merging operations, these can be multi-token mentions. While the extraction procedure is slightly different from Qasim et al. (2013)’s approach, one can easily see that both methods use a similar set of patterns. What differs more is the relation extraction approach: Here, for each pair of concept mentions in a sentence, the shortest path between them in the transformed graph is calculated and the tokens along that path are used as a relation mention. If another concept is crossed on that path, the method tries to find the shortest path excluding that concept.

5.2.2.3 Concept Extraction

In the first experiment, we test the performance of concept mention extraction for the three techniques from previous work described above and the three approaches based on predicate-argument analysis introduced earlier.

Experimental Setup To evaluate the coverage of concept mention extraction, we compare extracted concepts C with the concepts C_R of the reference concept maps. The two main metrics, recall and yield, are defined as:

$$\text{Recall} = \frac{|C \cap C_R|}{|C_R|} \qquad \text{Concept Yield} = \frac{|C|}{|C_R|}$$

While recall measures the fraction of covered reference concepts, concept yield indicates the amount of over-generation. The more common usage of precision is less useful in this case because the reference for comparison, C_R , are only the concepts included in the summary concept map, not all concept mentions in the text (which would be required for a proper precision calculation). Since the set C is typically much larger than C_R , computing

Approach	EDUC			BIOLOGY			ACL		
	Yield	Recall	Len	Yield	Recall	Len	Yield	Recall	Len
Noun Tokens	167.38	25.07	1.0	51.53	75.61	1.0	48.99	42.48	1.0
Valerio et al. (2006)	406.53	55.73	2.4	69.50	69.60	2.2	77.53	62.21	2.3
Qasim et al. (2013)	467.15	48.13	2.4	74.61	61.46	2.3	78.04	74.39	2.3
Villalon (2012)	351.85	51.20	2.5	60.75	76.37	2.3	69.10	76.09	2.3
OIE	277.70	58.00	5.9	44.53	41.80	4.9	41.99	28.67	5.3
SRL	481.59	46.93	6.5	66.28	50.99	4.2	77.21	44.14	5.0
PropS	451.20	46.27	4.3	76.55	73.50	3.2	55.87	58.41	3.7
Reference			3.2			1.2			1.9

Table 5.1: Concept extraction performance by dataset. Bold indicates best recall per group. Recall is given in percentages, yield and length (number of tokens per concept) are absolute values.

a precision metric against C_R leads to very low scores that are hard to interpret, in particular when combined with recall in an F-score. However, proper precision scores based on C_R can be computed after the selection of a summary subset from all extractions, as we will do in Section 5.2.2.5. All metrics are averaged over all pairs per dataset.

Note that we also use a simple form of mention grouping, based on exact matches of stemmed mentions, to obtain C . This avoids that mentions that can be easily seen to refer to the same concept are counted as separate concepts for the yield metric. As an extraction baseline, we include a strategy that extracts only single tokens tagged as nouns.

Recall and Yield Table 5.1 reports results for the EDUC, ACL and BIOLOGY dataset. Out of the different tools used to obtain predicate-argument structures, we observe the highest recall using PropS on two datasets and OIE on the other. From previous work, Villalon’s method shows the best results on two datasets, while Valerio et al.’s method is best on EDUC. Overall, we conclude that concept extraction based on predicate-argument structures is competitive, giving slightly better (EDUC) or slightly worse (BIOLOGY) results, except for the performance on the ACL dataset. With regard to yield, predicate-argument structures are even more competitive, producing less candidate concepts in most cases. For all approaches, the yield correlates with the size of the input documents, producing most concepts for the multi-document inputs of the EDUC dataset.

One interesting observation is that on EDUC the best performing method, among previous work as well as predicate-argument structures, differs from the one on the other two datasets. The reason is that the concept labels in this corpus tend to be longer (3.2 tokens

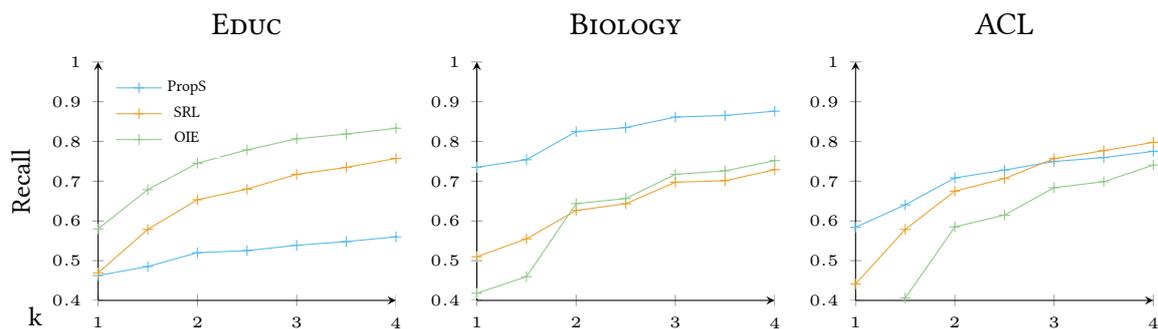


Figure 5.1: Concept extraction recall for inclusive matches at increasing thresholds of k .

vs. 1.2 and 1.9), including more complex noun phrases and also verbal phrases describing activities, while concept labels are mostly single nouns in BIOLOGY or noun compounds in ACL. This explains the generally lower recall and the better performance of approaches focusing on full noun phrases rather than nouns and noun compounds. Of course, the different style of concept mentions in EDUC is to some extent a result of the fact that OIE has been used to create the corpus (see Section 4.3.2), contributing to the good performance of OIE in this experiment. However, we want to emphasize that in later steps of the corpus creation process, all concept labels have been manually verified and often revised, ensuring that they are in fact good references for how humans would label concepts.

Analyzing the results on the ACL corpus, we found that the automatic extraction of text from the PDFs produced very noisy data, causing a lot of the dependency parses to be of low quality due to wrong sentence segmentation. Interestingly, while this reduced the performance of all approaches using predicate-argument structures, it did not influence the methods from previous work. We hypothesize that these approaches are more robust against these parsing errors because they only extract from dependencies locally, while the other approaches globally process the full parse to derive predicate-argument structures.

Added Concepts We further compared the concepts extracted by the best approach from previous work with those produced by predicate-argument structures. To assess whether the latter identified previously uncaptured concepts, we joined both sets and compared the combined recall against the method from previous work alone. In all cases, predicate-argument structures extract at least some concepts that are not covered by previous approaches, with the best approach adding 15.20 (EDUC), 6.90 (BIOLOGY) and 5.81 (ACL) points of recall. This shows that the predicate-argument structures are not only competitive but can also be used to extend the coverage of previous methods.

Concept Length Finally, we looked at the length of extracted concept mentions. As Table 5.1 shows, the extractions made by previous work tend to be around 2.3 tokens long, while the arguments of predicate-argument structures are up to three times as long. In order

to assess whether missing concepts might be subsequences of these longer mentions, we define a new evaluation metric: An extracted concept c and reference concept c_R *match inclusively at k* if c_R is contained in c and c is at most $k \cdot |c_R|$ tokens long.

Figure 5.1 shows the corresponding recall when increasing k . For all three approaches, but especially for SRL and OIE, the recall increases dramatically when considering longer arguments. This indicates that the concept extraction performance could be further improved by learning how to reduce longer arguments to the desired parts. Corresponding techniques have been studied, e.g. by Stanovsky and Dagan (2016a) and Stanovsky et al. (2016a). But as we already mentioned in Section 3.3, the compression of arguments has to ensure that propositions are still asserted by the text, making this direction non-trivial.

5.2.2.4 Relation Extraction

In the second experiment, we perform a similar comparison between the relation extraction techniques from previous work and the ones based on predicate-argument structures.

Experimental Setup To ensure a fair comparison independent of concept extraction, all strategies use reference concepts in this experiment. We compute recall and relation yield as our metrics, counting relations that were extracted with a correct label for the correct pair of reference concepts. To account for the fact that some approaches extract complex phrases, e.g. including prepositions or auxiliaries, while others extract only single tokens, we use a lenient matching criterion, requiring that the stemmed heads of the relation labels have to match. For instance, *is located in* and *located* are considered a match.

Results As shown in Table 5.2, the shortest path method of Villalon is the best performing method from previous work. However, it also creates a comparably large set of relations. Using predicate-argument structures, we see a substantial improvement on both datasets, and again, PropS performs well on BIOLOGY and OIE on EDUC. Note that on both datasets even the other method is on par with Villalon’s approach while producing a substantially smaller number of extracted relations.

We found that the low recall of Valerio et al.’s and Qasim et al.’s methods is due to the small coverage of their patterns, whereas Villalon’s method is very noisy and extracts many meaningless relation mentions. On the other hand, predicate-argument structures benefit from their main advantage in this evaluation: Since all extractions are made on the level of propositions, every concept has at least one meaningful relation to another concept.

With regard to the length of relation labels, we found a similar picture: Villalon’s method provides very long labels (4.5 and 4.4 tokens), as it simply takes all tokens along a path in the dependency graph. SRL yields the shortest labels (1.0), since predicates are restricted to one token by design, PropS finds a bit longer ones (1.5), including auxiliaries and light-verb constructions, and OIE extracts the longest labels (2.9 and 2.3), also containing

Approach	EDUC			BIOLOGY		
	Yield	Recall	Len	Yield	Recall	Len
Valerio et al. (2006)	2.70	8.32	1.8	2.02	17.75	1.8
Qasim et al. (2013)	2.24	3.30	1.7	1.43	8.08	1.4
Villalon (2012)	17.28	21.53	4.5	9.64	32.34	4.4
OIE	6.47	25.76	2.9	3.52	31.63	2.3
SRL	11.60	17.97	1.0	4.22	17.57	1.0
PropS	11.62	21.20	1.5	6.48	40.95	1.5
Reference			3.2			1.9

Table 5.2: Relation extraction performance by dataset. Bold indicates best recall per group. Recall is given in percentages, yield and length (number of tokens per concept) are absolute values.

prepositions as in *was made for*. Overall, considering both recall and the style of labels, we consider OIE to provide the most useful relations for concept maps.

5.2.2.5 Concept Selection

Finally, we present a third experiment on concept selection. While we found predicate-argument structures to be superior for relation extraction in terms of both recall and yield, the picture is less clear for concepts. By scoring the extracted concepts and selecting a subset of summary-worthy ones, we try to shed more light on the usefulness of certain trade-offs between recall and yield for subsequent steps in the pipeline. However, we want to emphasize that the experimental setup used here is not directly a subtask of CM-MDS: For CM-MDS, importance estimates for concepts are not used to simply select a subset of the concepts but to select a subgraph of concepts and relations.

Experimental Setup We use the concept sets C obtained from the different methods studied in the first experiment (see Section 5.2.2.3), assign a score to each concept, select the $|C_R|$ concepts with the highest score and compare them to the reference concepts C_R . This variant of precision is known as r-precision:

$$\text{R-Precision} = \frac{|C_{top-k} \cap C_R|}{|C_R|} \text{ with } k = |C_R|$$

As the score, we use the concept’s frequency, i.e. the number of its mentions in the documents, a metric that has been previously proposed to find important concepts (Valerio and

Approach	EDUC	BIOLOGY	ACL
Valerio et al. (2006)	15.07	29.46	17.75
Qasim et al. (2013)	14.27	24.99	19.47
Villalon (2012)	14.80	32.60	20.34
OIE	15.33	21.52	10.45
SRL	13.20	24.11	14.67
PropS	14.27	28.86	16.61

Table 5.3: Concept selection performance by dataset. Performance is measured in *r*-precision (given as percentages) and bold print indicates the best result per group and dataset.

Leake, 2006). Note that we are mainly interested in the difference between approaches and not the absolute selection performance in this experiment.

Results Table 5.3 shows the selection precision for the concepts obtained with different extraction methods. As expected, the performance closely resembles the picture obtained from the extraction experiment: If the recall is higher after extraction, precision is also higher after the selection. However, an interesting exception is for example PropS and SRL on EDUC: While SRL has a slightly higher recall (46.93 vs. 46.27), PropS selects more relevant concepts (13.20 vs. 14.37), which might be due to the lower yield of PropS.

Summarizing this set of experiments, we conclude that using predicate-argument analysis is a promising approach for concept and relation mention extraction. Comparing them to several previous methods specifically developed for concept map mining, we found that they substantially improve relation extraction while being competitive with regard to concept extraction. PropS representations are particularly good to capture short noun-focused concepts whereas longer and more complex concept labels are more reliably extracted with OIE. Considering the good performance and the ease of use — as opposed to manually defining syntactic patterns — future work on mention extraction for concept maps should rely on ready-to-use predicate-argument analysis.

5.3 Predicate-Argument Analysis for German

As we already pointed out in Section 2.3.1, most work on concept map mining has focused on text in English. The only exceptions that we are aware of are the works of Kowata et al. (2010) for Portuguese and Zubrinic et al. (2012, 2015) for Croatian. Among the predicate-argument analysis tools used in the preceding section, the picture is similar. Both PropS and OpenIE4 are designed for English and no version for another language exists. As most

proposed methods for concept and relation extraction as well as a large part of the existing OIE systems rely on sets of hand-designed rules, an important question is how these rule sets can be created for additional languages.⁴¹

In order to make rule-based systems available in new languages, several approaches can be used. First, one can create a new set of rules from scratch that is specifically designed for that language. *ExtrHech* (Zhila and Gelbukh, 2013), an OIE system for Spanish, was created this way. While leading to high-quality rules for the new language, a problem is that this approach is as laborious as the creation of the rules in the initial language, making scaling to many languages a huge effort. A second approach is to transfer the existing rules for English manually to the new language. For the task of temporal tagging, this approach resulted in a competitive system for French (Moriceau and Tannier, 2014). With the existing rules as a starting point, it typically takes less effort than creating rules from scratch while the manual process still allows to address all idiosyncrasies of the new language.

A third alternative is to fully automate the process by automatically “translating” the existing rules from English to the target language. In the case of temporal tagging, this idea allowed Strötgen and Gertz (2015) to obtain baseline taggers for more than 200 languages. However, the more the source and target languages differ from each other, the more difficult it is to derive rules that cover all specifics of the new language by automatic means only. Differences in syntax and dissimilarities in the corresponding part-of-speech and dependency representations are challenges for automatic rule translation. And if certain phenomena only exist in the target language, no rules will cover them. Strötgen and Gertz (2015) compare their automatically derived rule sets against manually created ones on 11 languages, observing F1-scores that are between 9 and 79 points lower.

To better understand the challenges and the effort associated with porting a set of rules for concept and relation mention extraction, we perform a case-study of porting the PropS system to German. For the purpose of that study, we follow the second approach, i.e. the manual transfer of existing rules. That allows us to assess the difficulty of porting rules between English and German and to evaluate whether such a process could be automated in the future. We select PropS as an example for rule-based predicate-argument analysis because an open-source implementation of the system is available and we were able to directly work with the authors of PropS for that case-study. At the time the study was carried

⁴¹In contrast, for SRL, all existing systems that we are aware of, including the work of Björkelund et al. (2009), are supervised models trained on annotated data. Thus, making such systems available in different languages means creating annotated corpora for those languages rather than designing new, language-specific rules. As part of the 2009 CoNLL shared task (Hajič et al., 2009), SRL data for German based on the SALSAs corpus (Burchardt et al., 2006) has been made available, such that SRL systems exist for German. However, in this section, we focus on rule-based OIE systems since they showed better performance in the extraction experiments and are also very similar to extraction methods proposed for concept maps, allowing us to gain insight into the effort associated with porting such methods to other languages.

out, there was also no existing predicate-argument analysis tool for German,⁴² making not just the study but also its resulting artifact — a new tool for German — interesting.

For the sake of completeness, we want to note that there is also a fourth approach to the problem of making rule-based systems available in new languages: If the same syntactic representation on which the rules operate can be used across multiple languages, different rules per language are not needed at all. For instance, ArgOE (Gamallo and Garcia, 2015), a rule-based OIE system, relies on a dependency parser that uses a common tagset for five European languages. After the case-study reported in the remainder of this section had been carried out, White et al. (2016) published PredPatt, an OIE system that extracts predicate-argument structures from universal dependencies. Universal dependencies (Nivre et al., 2016) are a recently developed, language-independent syntactic representation with annotated treebanks for over 70 languages, allowing PredPatt to apply their ruleset to all of them. A later evaluation (Zhang et al., 2017) showed very promising results for English, while the performance on other languages has yet to be tested.

5.3.1 PropS

PropS (Stanovsky et al., 2016b), which we already briefly introduced in Section 5.2.1.2, is a rule-based converter that turns dependency graphs for English into typed graphs of predicates and arguments. Compared to the original dependency representation, these graphs differ in several aspects: First, they mask non-core syntactic details, such as tense or determiners, by removing corresponding tokens and instead encoding this information as features of the remaining nodes. Second, they unify semantically equivalent constructions, such as active and passive alternations of a sentence. And finally, they explicate implicit propositions, such as those indicated by possessives (*Peter's car*) or appositions (*Trump, the US president*), by introducing additional predicate nodes. As input, PropS expects the syntactic structure of a sentence in the collapsed and propagated version of Stanford dependencies (de Marneffe and Manning, 2008). The example in Figure 5.2 shows a German sentence with its dependency representation and the derived PropS graph.

At this point, it is important to note that PropS can be seen in two different ways. First, it is a graph-based sentence representation that makes predicate-argument structures easier to access compared to dependency graphs. This can be useful for a range of applications that have to identify predicates and arguments in sentences. And second, PropS can also be seen as an OIE system because OIE-like extractions can be easily made from the graph-based representation. The open-source implementation⁴³ provides both graphs and OIE tuples as

⁴²Later, an OIE system for German, GerIE (Bassa et al., 2018), has been published. It also uses hand-written rules based on dependencies, but in contrast to our case-study here, they are not direct translations of rules of an existing system for English.

⁴³<https://github.com/gabrielStanovsky/props>

output. Therefore, we mention PropS in our review of OIE systems (see Section 2.3.3) while also including it as a more general predicate-argument analysis tool in Section 5.2.

OIE tuples can be extracted from the PropS graph in a straightforward way. Every non-nested predicate node p in the graph, together with its n argument-subgraphs a_1, \dots, a_n , yields a tuple $(a_1 - p - a_2 - \dots - a_n)$. Due to additional nodes for implicit predicates, for instance those indicated by possessive constructions, PropS can also make extractions that go beyond the scope of other OIE systems, such as *(Michael - has - bicycle)* from *Michael's bicycle is red*. In line with more recent OIE systems, PropS extracts tuples that are not necessarily binary, but can be unary or of higher arity.

5.3.2 Porting Rules to German

In our analysis, we assess for each rule of the converter that transforms a dependency graph to the PropS graph whether and how it can be used on German text. A rule transforms a part of the graph if it fulfills conditions referring to dependency types, part-of-speech (POS) tags and lemmas. The following are two simple rules used in the English version of PropS:

if graph has edge *det*(X, Y) then delete Y
 if graph has edge *nn*(X, Y) then merge X and Y

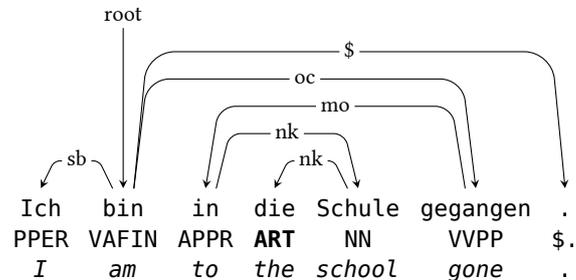
In line with the English system, which works on collapsed and propagated Stanford dependencies (de Marneffe and Manning, 2008), we assume a similar input representation for German that can be obtained with a set of collapsing and propagation rules provided by Ruppert et al. (2015) for TIGER dependencies (Seeker and Kuhn, 2012).

Overall, we find that most rules can be used for German, in particular because some syntactic differences, such as freer word order (Kübler, 2008), are already masked by the dependency representation (Seeker and Kuhn, 2012). We identified three groups of rules that can be transferred to German with different amounts of effort and also identified the need for some additional rules. In the next sections, we discuss these groups.

Directly Applicable Rules About 38% of the rule set can be directly ported to German, solely replacing dependency types, POS tags and lemmas with their German equivalents. As an example, the PropS rule removing negation tokens looks for *neg* dependencies in the graph, for which a corresponding type *ng* exists in the German tag set. We found similar correspondences to remove punctuation and merge proper nouns and number compounds. In addition, we can also handle appositions and existentials with direct mappings.

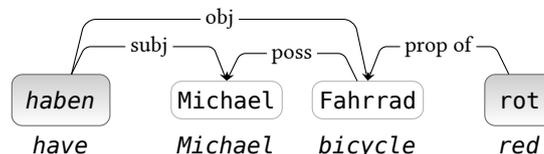
Rules Requiring Small Changes For 35% of the English rules, small changes are necessary, mainly because no direct mapping to the German tag set is possible or the annotation style differs. For instance, while English has a specific type *det* to link determiners to their

governor, a more generic type *nk* (noun kernel modifier) is used in German that also occurs in other cases. Instead, determiners can be easily detected by part-of-speech, tagged as *ART*, as the following example illustrates:



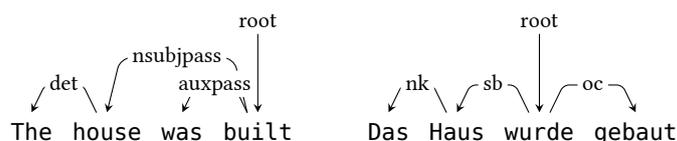
Another type of difference exists with regard to the representation of auxiliary verb constructions. In Stanford dependencies, main verbs govern all auxiliaries, whereas in TIGER dependencies, an auxiliary governs the main verb. The above example shows this for *gone* and *am*. Therefore, all rules identifying and removing auxiliaries and modals have to be adapted to account for this difference.

With similar changes as discussed for determiners, we can also handle possessive and copular constructions. The graph for *Michael's bicycle is red*, for example, features an additional predicate *have* to explicate the implicit possessive relation. The copular verb *is* is omitted and *red* becomes an adjectival predicate in the graph representing this sentence:



Moreover, conditional constructions can be processed with slight changes as well. Missing a counterpart for the type *mark*, we instead look for subordinating conjunctions by part-of-speech. In fact, we found conditionals to be represented more consistently across different conjunctions, making their handling in German easier than in English.

Rules Requiring Substantial Changes More substantial changes are necessary for the remaining 27% of the rules. To represent active and passive in a uniform way, PropS turns the subject into an object and a potential by-clause into the subject in passive clauses. For English, these cases are indicated by the presence of passive dependencies such as *nsubjpass*. For German, however, no direct counterparts exist and instead passive constructions use the same types as active ones. The following example illustrates this for a short sentence:



As an alternative strategy, we instead look for past participle verbs (by part-of-speech) that are governed by a form of the auxiliary *werden* (Schäfer, 2015). Instances of the German static passive (Zustandspassiv) are, in contrast, handled like copulas.

Another deviation from the English system is necessary for relative clauses. PropS heavily relies on the Stanford dependency converter, which propagates dependencies of the relative pronoun to its referent. The German collapser does not have this feature, and we therefore implement it as an additional transformation. As an example, consider Figure 5.2, where the *sb* dependency from *liegen* to *die* is propagated to the referent *Orte* in the PropS graph (and is labeled as *subj* in PropS instead of *sb*, which is used in TIGER).

To abstract away from different tenses, PropS represents predicates with their lemma, indicating the original tense as a feature, as detected with a set of rules operating on POS tags. For German, no tense information is contained in POS tags, but instead, a morphological analysis can provide it. Determining the overall tense of a sentence based on that requires a new set of rules, as the grammatical construction of tenses differs between German and English. PropS also tries to heuristically identify raising constructions, in which syntactic and semantic roles of arguments differ. In German, this phenomenon occurs in similar situations, such as in *Michael scheint zu lächeln* (*Michael seems to smile*), in which *Michael* is not the semantic subject of *scheinen*, though syntactically it is. To determine these cases heuristically, an empirically derived list of common raising verbs, such as done by Chrupala and van Genabith (2007) for English, needs to be created for German.

Additional Rules An additional step that is necessary during the lemmatization of verbs for German is to recover separated particles. For example, a verb like *ankommen* (*arrive*) can be split in a sentence such as *Er kam an* (*He arrived*), moving the particle to the end of the sentence, with a potentially large number of other tokens in between. We can reliably reattach these particles based on the dependency parse. Another addition to the rules that we consider important is to detect subjunctive forms of verbs and indicate the mood with a specific feature for the predicate. A morphological analysis provides the necessary input. Compared to English, the usage of the subjunctive is much more common, usually to indicate either unreality or indirect speech (Thieroff, 2004).

Table 5.4 summarizes our analysis of PropS' portability to German. With 38% of the rules being directly transferable and 35% requiring only small changes, the necessary effort to create a German version of PropS seems to be substantially smaller than creating a complete rule set for German from scratch.

Category	Size	Description
Directly applicable	38%	Only replacement of POS tags and dependency types with German equivalent, one-to-one mapping exists.
Small changes	35%	No one-to-one mapping exists, but alternative conditions, e.g. using POS instead of dependencies, or vice-versa, can be easily found.
Substantial changes	27%	Rules cannot be used, complex alternative rules or resources have to be created instead.

Table 5.4: Analysis of the portability of PropS rules from English to German.

Following that analysis, we implemented a German version of PropS, named PropsDE. It uses Mate Tools for POS tagging, lemmatizing and parsing (Bohnet et al., 2013). Dependencies are collapsed and propagated with JoBimText (Ruppert et al., 2015). The rule set covers 89% of the English rules, lacking only the handling of raising-to-subject verbs and more advanced strategies for coordination constructions and tense detection. Similar to PropS, the implementation provides both PropS graphs and OIE extractions as its output. For the latter, similar to other OIE systems, PropsDE assigns confidence scores to extracted tuples. It uses a logistic regression model that has been trained on 410 extractions annotated for correctness. Model features are the length of the input sentence, length of the extraction, its number of arguments, whether it contains punctuation and which dependency types and PropS edge labels are present in the corresponding part of the graph.

Based on correspondence with the authors of the English system and their estimation of the effort they put into building it, we can conclude that we implemented the German version with roughly 10% of the effort they reported, including both the conceptual analysis and the technical implementation. This shows that our approach of manually porting a rule-based system to a new language is a valid approach to overcome the lack of tools in a specific language with reasonable effort in a short amount of time. However, the analysis also pointed out that there are a range of challenges, specifically the rules requiring substantial changes, that make a fully automatic porting difficult. For target languages that are more different from English than German, these cases are presumably even more prevalent.

5.3.3 Experiments

In addition to the more qualitative analysis of PropS portability in the previous section, we close this chapter with a final experiment aiming to evaluate the performance of the new PropsDE system quantitatively. While the direct evaluation of PropS graphs is difficult, as there is no dataset with gold graphs, neither for English nor German, we instead make use

of the fact that PropS (and PropsDE) can be used as OIE systems and follow the standard evaluation protocol for such systems.

Experimental Setup We manually label extractions made by PropsDE to assess its performance. For this purpose, we created a new dataset consisting of 300 German sentences, randomly sampled from three sources of different genres: news articles from the TIGER treebank (Brants et al., 2004), German web pages from CommonCrawl (Habernal et al., 2016b) and featured Wikipedia articles. For the treebank part, we use the text with both gold and parsed dependencies to analyze the impact that parsing errors have on PropsDE.

Every tuple extracted from the set of 300 sentences was labeled independently by two annotators as correct or incorrect. In line with previous work, they were instructed to label an extraction as incorrect if it has a wrong predicate or argument, including overspecified and incomplete arguments, or if it is well-formed but not entailed by the sentence. Unresolved co-references were not marked as incorrect. We observed an inter-annotator agreement of 85% ($\kappa = 0.63$). For the evaluation, we merged the labels, considering an extraction as correct only if both annotators labeled it as such. Results are measured in terms of precision, the fraction of correct extractions, and yield, the total number of extractions.⁴⁴ The latter is commonly used as an indicator of recall, which cannot be determined directly in this kind of evaluation. We also plot precision-yield curves obtained by gradually decreasing a threshold for the extraction confidence. The confidence prediction model was trained on a separate development set.

Results From the whole corpus of 300 sentences, PropsDE extracted 487 tuples, yielding on average 1.6 per sentence with 2.9 arguments. 60% of them were labeled as correct. Table 5.5 shows that most extractions are made from Wikipedia articles, whereas the highest precision can be observed for newswire text. According to our expectations, web pages are most challenging, presumably due to noisier language. These differences between the genres can also be seen in the precision-yield curve (Figure 5.3).

For English, state-of-the-art systems show a similar performance. In a direct comparison of several systems carried out by Del Corro and Gemulla (2013), they observed precision scores of 58% (Fader et al., 2011, ReVerb), 57% (Del Corro and Gemulla, 2013, ClausIE), 43% (Wu and Weld, 2010, WOE) and 43% (Mausam et al., 2012, OLLIE) on datasets of similar genres. The reported yield per sentence is higher for ClausIE (4.2), OLLIE (2.6) and WOE (2.1), but smaller for Reverb (1.4). However, we note that in the evaluation, all systems were configured to output two-argument-tuples. For example, from a sentence such as

⁴⁴Note that the yield metric here, commonly used for OIE systems, measures the number of extractions per sentence, while the concept and relation yield metrics used in Section 5.2.2 measure extractions per reference concept or relation.

Genre	Sentences	Length	Yield	Precision
News*	100	19.3	1.42	78.87
News	100	19.3	1.44	70.83
Wiki	100	21.4	1.78	61.80
Web	100	19.2	1.65	49.09
Total	300	20.0	1.62	60.16

Table 5.5: Tuple extraction performance of PropsDE by text genre. News* indicates extraction from gold parses, not included in Total. Precision in percentages, length in tokens per sentence.

The principal opposition parties boycotted the polls after accusations of vote-rigging.

OLLIE can either make two binary extractions

- (1) *(the principal opposition parties - boycotted - the polls)*
- (2) *(the principal opposition parties - boycotted the polls after - accusations of vote-rigging)*

or just a single extraction with three arguments. PropS always extracts the combined tuple

- (1) *(the principal opposition parties - boycotted - the polls - after accusations of vote-rigging)*

which is in line with the default configuration of more recent OIE systems.

For the sake of comparability, we conjecture that the yield of our system would increase if we broke down higher-arity tuples in a similar fashion: Assuming that every extraction with n arguments, $n > 2$, can be split into $n - 1$ separate extractions, our system's yield would increase from 1.6 to 3.0. That is in line with the numbers reported above for the binary configuration for English. Overall, this indicates a reasonable performance of our straightforward porting of PropS to German.

Extractions were most frequently labeled as incorrect due to false relation labels (32%), overspecified arguments (21%) and wrong word order in arguments (19%). Analyzing our system's performance on the treebank, we can see that the usage of gold dependencies increases the precision by 8 percentage points, making parsing errors responsible for about 28% of the incorrect extractions. Since the Mate Tools parser is trained on the full TIGER treebank, including the news part of our experimental data, its error contribution on unseen data might be even higher.

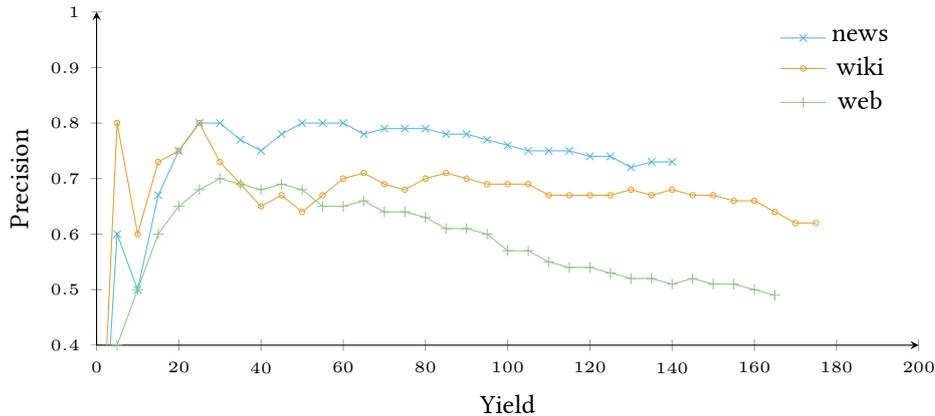


Figure 5.3: Extraction precision of PropsDE at increasing yield by genre.

As we mentioned earlier, a second OIE system for German, GerIE (Bassa et al., 2018), has been published recently. Using their own dataset consisting of sentences from news articles and encyclopedic articles, they performed an evaluation of both GerIE and PropsDE. They report precision scores of 91% for GerIE and 85% for PropsDE on news data as well as 88% and 68% on the encyclopedic articles, indicating that after our initial work presented here, OIE systems for German are now being actively developed and further improved.

5.4 Chapter Summary

In this chapter, we focused on the concept and relation mention extraction subtasks of CM-MDS. As the first contribution, we addressed the lack of a clearly established state-of-the-art among previously proposed extraction methods by carrying out a first direct comparison of such methods. For that purpose, we selected three representative approaches, reimplemented them and compared their performance on our new benchmark corpora. The most interesting finding of these experiments is that previously proposed methods for relation mention extraction perform poorly on the datasets, with some showing a recall below 10% while the best approach finds about a third of all relations. For concept mentions, the results are better, but still, between a fourth and half of the reference concepts are missed.

As our second contribution, we proposed to extract concept and relation mentions from predicate-argument structures instead of syntactic representations. We tested this idea using three different representations, namely SRL, PropS and OIE, and compared their performance to the aforementioned methods. We found that this novel approach substantially improves the relation extraction performance, identifying more and higher-quality relation mentions, while performing comparable to previous work for concept mention extraction. Given that the use of such readily available predicate-argument analysis tools greatly reduces the effort of designing concept and relation extraction methods by alleviating the

need of hand-designing extraction rules, we argue that this direction is a very attractive alternative to existing extraction methods.

Despite the fact that our newly proposed method improves upon previous work, the experiments also showed that there are still many concept and relation mentions that are not identified by current methods. To further improve methods for this subtask, we point out two directions for future work: First, our experiments showed that many desirable mentions are in fact contained in longer mentions extracted by current methods. Towards that end, models that can reliably reduce mentions to more concise ones without introducing unasserted propositions could be helpful. And second, more generally, one could replace using hand-written extraction rules on dependencies or predicate-argument structures with supervised models that treat the mention extraction task as a sequence tagging problem, as it is, for instance, commonly done for named-entity recognition. While such a data-driven approach could close the recall gap, we are not aware of any existing datasets providing the necessary token-level annotations of concept and relation mentions for concept maps.

As the third contribution, we carried out an extensive analysis of the rule set of PropS, one of the predicate-argument analysis tools, to assess how difficult it is to port it to another language, in particular, to German. As most previous work for concept and relation extraction as well as OIE focused on English, it is important to know how challenging and laborious it is to also obtain such systems for other languages. The analysis revealed that a large fraction of the PropS rules can be easily ported to German, requiring only small adaptations. With roughly 10% of the effort that went into the English system, we could build a variant for German covering 89% of the rules. However, we also observed that some rules are more challenging, making fully automatic approaches to transfer rules difficult in practice. As a result of that analysis, we presented PropsDE, a predicate-argument analysis tool for German that is the first system available to make OIE-style extractions from German text. In an evaluation of its extraction capabilities, we showed that its performance is comparable to existing OIE systems for English.

Pipeline-based Approaches

After the previous chapter looked in detail at concept and relation mention extraction, we focus on the remaining subtasks of CM-MDS – mention grouping, importance estimation and concept map construction – in this chapter. We first study each subtask in isolation and propose new techniques to address its challenges. In the final part of the chapter, we then combine techniques for all subtasks into a pipeline and evaluate its overall task performance, the quality of the generated concept maps and the scalability of the pipeline.

6.1 Motivation and Challenges

Most of the existing work on concept map mining, reviewed in Section 2.3.1, focuses mainly on the concept and relation mention extraction subtasks. Consequently, for the remaining subtasks of CM-MDS, only a small number of fairly simple techniques have been suggested and evaluated in the past.

For mention grouping, as we discussed in Section 3.3.2 and Section 3.3.4, the main challenge is the variety of expressions that can be used to refer to the same concept. For instance, in the example in Figure 3.1, the mentions *ADHD symptoms* and *the symptoms of ADHD* as well as *hypnosis* and *hypnotherapy* refer to the same concepts. Morphological variants, synonyms and paraphrases have to be recognized to identify all mentions of a concept. If mentions are not grouped correctly, this can have several negative effects on the quality of a summary concept map: First, failing to identify all mentions leads to incorrect counts of how often a concept is mentioned, reducing the quality of the frequency signal to estimate importance. Further, if several ungrouped mentions of the same concept are selected for the summary concept map, these redundant concepts make it harder for a user to look for relations of that concept, as they are spread among the duplicates. And in addition, the duplicates waste valuable space that could otherwise be used to include additional concepts.

We reviewed how existing work approaches these challenges in Section 2.3.1.3. Methods that have been explored include stemming (Villalon, 2012), matching substrings (Valerio and Leake, 2006) and looking up synonyms in WordNet (Oliveira et al., 2001, Villalon, 2012). Other methods group mentions based on frequency-based term-vector representations (Rajaraman and Tan, 2002) or compute similarities based on the WordNet structure (Aguiar et al., 2016). While there is no fundamental problem with this direction, the main limitation is that all applied methods are rather old and far behind the current state-of-the-art in NLP for measuring semantic similarity. For instance, dense word vectors (Mikolov et al., 2013b, Pennington et al., 2014) have been shown to be superior to one-hot or frequency based representations as used by Rajaraman and Tan (2002). In order to improve methods for this subtask, we thus study the performance of more recent approaches to measure semantic similarity and design concept grouping methods based on them.

The selection of a summary-worthy subset of all extracted concepts and relations was largely ignored in previous work, as many studies did not have a focus on summarization. Consequently, the importance estimation subtask did not receive much attention. However, when dealing with larger document clusters, as in our EDUC benchmark corpus, this step becomes inevitable. In previous work on concept map mining, the use of term frequencies (Valerio and Leake, 2006), concept-based TF-IDF (Zubrinic et al., 2015), LSA (Villalon, 2012) and concept map-specific scoring models (Aguiar et al., 2016) has been explored. As we showed in the review in Section 2.3.2, many more techniques for importance estimation have been proposed in existing work on MDS. To improve methods for CM-MDS, we thus transfer such techniques to our problem. In particular, we study the effectiveness of a large set of features and, for the first time for concept maps, learn supervised models from annotated data to estimate the importance of concepts.

The final subtask, concept map construction, is least frequently addressed in previous work. In fact, no previous work on concept map mining has explicitly formulated a size restriction and connectedness constraint as we do for CM-MDS and, consequently, that work has also not formalized the subgraph selection problem as in Equations 3.1 and 3.2. The only work in that direction is the graph reduction heuristic of Zubrinic et al. (2015), which selects a connected graph of the target size, but not necessarily the best one according to our objective function (or any other explicitly formulated objective). Towards this end, we present an ILP to solve the selection problem optimally.

We finally combine our newly proposed methods for mention grouping, importance estimation and concept map construction with extraction approaches from the previous chapter in a pipeline that can handle the full scope of CM-MDS. On EDUC and WIKI, we compare its performance against the baseline described in Chapter 4 and analyze the created summary concept maps. Since EDUC is a corpus with sets of around 40 documents to summarize, which, while being a realistic real-world application scenario, is 10 to 15 times larger than traditional MDS corpora (see Table 4.4), scalability is another important chal-

lenge. In contrast to previous work, we therefore pay particular attention to the runtime complexity of proposed methods and evaluate their feasibility on the large corpus.

6.2 Concept Mention Grouping

Given the set of concept mentions M extracted from the text, the goal of concept grouping is to partition M into subsets that refer to the same concept. We model this task with a two-step approach that first, for any pair of mentions m_1, m_2 , predicts if they refer to the same concept, and second, given these pairwise predictions, derives the desired partitioning. By using a binary classifier, we can easily combine different features indicating coreference in a single model, allowing us to use the signals used in existing work on concept maps discussed in the previous section as well as more recent measures of semantic similarity.

This two-step approach is a common model for coreference resolution (Zheng et al., 2011). However, we would like to point out that concept grouping is different from the general coreference resolution task. First, we have to perform the resolution across a set of documents, while coreference resolution typically works within one document. A typical feature such as the recency of a mention (Jurafsky and Martin, 2009) is not available when working with mentions from different documents. Second, we are mainly interested in coreferences between noun phrases⁴⁵, whereas general coreference resolution methods also spent considerable efforts on resolving pronominal anaphora. Typical features for this subclass, such as person, number and gender agreement, are not useful in our scenario. And third and most importantly, the partitioning step is, due to the number of mentions faced when working with a set of documents, crucial for our task. For coreference resolution, it is less relevant, e.g. the recent state-of-the-art model by Lee et al. (2017) on the English OntoNotes coreference benchmark (Pradhan et al., 2012) simply uses the partitioning induced by the transitive closure over its predictions. Despite these differences, we also evaluate the performance of existing coreference resolution methods to concept grouping and compare it to our method in the end-to-end experiments in Section 6.5.

6.2.1 Pairwise Mention Classification

As potential features for the coreference of concept mentions, we use a set of lexical and semantic similarity measures that compare the tokens in both mentions. Given a pair of mentions (m_1, m_2) , each of these measures returns a real-valued number $sim(m_1, m_2) \in [0, 1]$,

⁴⁵Note that the extraction methods studied in Chapter 5 target noun phrase-like structures for concept mentions. Pronouns are either ignored, or, as in some previous work (Oliveira et al., 2001, Qasim et al., 2013, Aguiar et al., 2016) and our pipeline presented in Section 6.5, resolved before mention extraction. No handling of pronouns is therefore necessary during the grouping.

where 1 indicates a high similarity. We study their individual and combined effectiveness in experiments in Section 6.2.3.

Lexical Similarities We include an **exact match** measure that is 1 if m_1 and m_2 are exactly the same and a **lemma match** measure that is 1 if m_1 and m_2 are equal after reducing all tokens in both mentions to their lemmas. Further, we use the same lemmatized tokens to compute the overlap of the mentions as a **Jaccard coefficient**

$$sim_{Jaccard}(m_1, m_2) = \frac{|m_1 \cap m_2|}{|m_1 \cup m_2|}, \quad (6.1)$$

treating both mentions as sets of tokens, and we use Levenshtein’s **edit distance** to compute a length-adjusted, character-based similarity defined as

$$sim_{edit}(m_1, m_2) = \max \left(0, 1 - \frac{edit_dist(m_1, m_2)}{\max(\#character(m_1), \#character(m_2))} \right). \quad (6.2)$$

All similarity measures ignore case. Note that both the lemma match and edit distance similarity can detect *kid* and *kids* to be very similar, while they are unrelated mentions for the exact match. The Jaccard similarity on the other hand can find similarities between multi-token mentions if a certain number of tokens is shared.

Word-Net–based Similarities To also capture coreferences with no lexical overlap, such as *child* and *kid*, we experiment with a range of similarity measures based on WordNet (Miller et al., 1990). We use the methods deriving word-level similarity measures from the WordNet structure proposed by Wu and Palmer (1994), Resnik (1995), Jiang and Conrath (1997), Leacock and Chodorow (1998) and Lin (1998) in combination with a phrase-level aggregation of these similarities proposed by Rus and Lintean (2012). Note that the last method, (Lin, 1998), has also been applied to concept map mining by Aguiar et al. (2016). Further, we use an alternative phrase-level aggregation developed by Corley and Mihalcea (2005) and a more recent method proposed by Pilehvar et al. (2013), called **ADW**, that uses alignments, word-sense disambiguation and random walks in WordNet.

Vector-based Representations A problem of WordNet is that, being a manually created lexicon, it is of limited size and does not contain all words one encounters in text. As an alternative, approaches that derive dense vector representations for words from large corpora have been developed. Based on the fact that words with similar meanings tend to occur in similar contexts, the distance between such vectors is strongly connected with semantic similarity. For our experiments, we use word vectors trained with the **Word2Vec** (Mikolov et al., 2013a) and **GloVe** (Pennington et al., 2014) models. For a mention m , we

compute a mention vector as the sum⁴⁶ of its word vectors

$$v(m) = \sum_{t_i \in m} v(t_i) \quad (6.3)$$

and measure the similarity between two mentions by cosine similarity

$$sim_{vector}(m_1, m_2) = \frac{v(m_1) \cdot v(m_2)}{\|v(m_1)\| \|v(m_2)\|}. \quad (6.4)$$

In addition, we compute a third vector-based similarity using the older approach of LSA (Deerwester et al., 1990) to obtain word-level similarities in combination with Rus and Lintean (2012)’s sentence-level aggregation method.

We combine the aforementioned semantic similarity measures as features in a binary classification model that determines whether a pair of mentions is coreferent. Since the classifier has to be applied to all pairs of mentions, whose number is in $\mathcal{O}(|M|^2)$ and can easily become very large (see Section 6.2.3 for concrete problem sizes), designing a practically useful classifier requires a trade-off between classification performance and runtime. Therefore, we resort to a simple log-linear model

$$P(y = 1 \mid m_1, m_2, w) = \frac{1}{1 + e^{-w^T \phi(m_1, m_2)}}. \quad (6.5)$$

Here, w denotes the learned weights for the features $\phi(m_1, m_2)$ that indicate the similarity of two mentions. In the experiments presented in Section 6.2.3, we examine the classification performance and computation time of the set of similarities discussed before and determine a subset of them that offers a reasonable trade-off of both metrics.

6.2.2 Mention Partitioning

The second step of our concept grouping approach is to derive a partitioning of all mentions from the pairwise classifications. Given mention pairs $(m_1, m_2) \in M^2$ and their coreference probabilities denoted as $cr(m_1, m_2)$, we are interested in the relation

$$Pos = \{(m_1, m_2) \mid (m_1, m_2) \in M^2 \wedge cr(m_1, m_2) \geq 0.5\} \quad (6.6)$$

of all pairs predicted to be coreferent. The main challenge to solve in this step is that Pos does not necessarily induce a valid partitioning of M because the pairwise classifications can contradict each other, leading to no clear partitioning.

⁴⁶Note that while taking the average instead of the sum is a more common way to represent sequences of tokens with word vectors, whether to use the sum or the average does not matter in our case as it is canceled out in the later computation of cosine similarity.

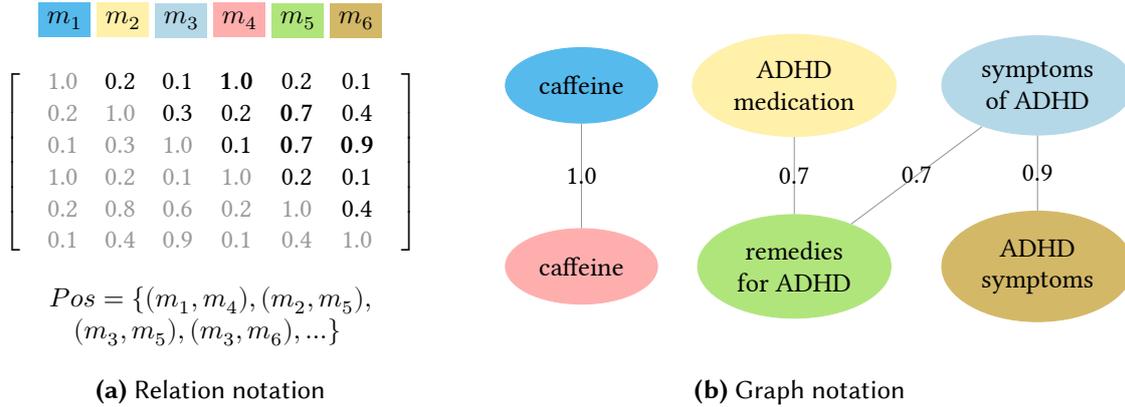


Figure 6.1: Partitioning example with six mentions and coreference predictions for all 15 mention pairs. In (a), the matrix of all pairwise predictions is shown as well as the resulting relation of positive classifications. In graph notation (b), positive classifications are represented by edges.

As an example, consider the mentions and their classifications given in Figure 6.1. Here, the mention pair (*ADHD medication*, *remedies for ADHD*) has been classified as coreferent as well as the pair (*symptoms of ADHD*, *remedies for ADHD*). However, the pair (*ADHD medication*, *symptoms of ADHD*) was found not to be coreferent. If we group all three mentions together, that would conflict with the third prediction, whereas all groupings of two mentions (with the third being another group) would conflict with at least one of the two positive classifications. Resolving such conflicts is the main task of this second step of concept grouping.

More formally, in order to induce a valid partitioning of M , we need a relation that is reflexive, symmetric and transitive. Known as an equivalence relation, the set of its equivalence classes then yields the desired partitioning C . By design, Pos is already guaranteed to be reflexive and symmetric.⁴⁷ The task thus boils down to deriving a transitive relation from Pos . Figure 6.1 also shows an alternative representation of the problem in terms of graph theory: Given an undirected graph in which nodes represent mentions and edges connect coreferent mentions, a valid partitioning would be a graph in which all connected components are cliques, i.e. complete subgraphs. Clearly, the graph in the example violates this requirement, lacking three more edges in the right component.

Transitive Closure A simple approach to obtain a transitive relation from Pos is to compute its transitive closure Pos^+ , i.e. the smallest transitive relation containing Pos . For our example, the transitive closure is (omitting reflexive and symmetric pairs for brevity)

$$Pos^+ = Pos \cup \{(m_2, m_3), (m_2, m_6), (m_5, m_6), \dots\} \quad (6.7)$$

⁴⁷Note that our model ensures symmetry, i.e. $cr(m_1, m_2) = cr(m_2, m_1)$, by using only pairwise, symmetric features. Also note that due to the symmetry, the classifier has to be applied to only the $\frac{|M|(|M|-1)}{2}$ unordered pairs rather than all $|M|^2$ pairs, as the remaining predictions are given by symmetry and reflexivity.

and the partitioning induced by that relation is

$$C = \{\{m_1, m_4\}, \{m_2, m_3, m_5, m_6\}\}. \quad (6.8)$$

In the graph notation, the transitive closure amounts to adding the missing edges to connected components or, since we are only interested in the resulting equivalence classes, simply to identifying connected components. Computationally, the transitive closure can be computed in $\mathcal{O}(|M|^2)$ time and with $\mathcal{O}(|M|)$ additional space to store the resulting partitioning. Using a disjoint-set data structure (Galler and Fisher, 1964) with (practically) constant-time merge operations (Tarjan and van Leeuwen, 1984) and one pass over the positive pairs, the runtime can be improved in practice. However, that is only beneficial if positive predictions *Pos* are available in a data structure that allows to directly iterate over them instead of iterating over all $|M|^2$ pairs of mentions. The worst-case complexity still remains $|M|^2$, since even then, *Pos* could theoretically be equal to M^2 .

While being conceptually simple and comparably cheap to compute, the transitive closure has the problem of lumping together many mentions to large groups. In the example, all four mentions, *ADHD medication*, *remedies for ADHD*, *symptoms of ADHD* and *ADHD symptoms*, end up in the same equivalence class and thus become a single concept. This happens because the approach is biased towards positive classifications, as it only extends them to be transitive but completely ignores negative classifications (and their probabilities). As a more serious example, consider two concepts with 100 mentions each. If only one out of the 10k pairwise mention comparisons across the two concepts would be classified as coreferent, the transitive closure would group all 200 mentions together to a single concept. The more mentions are available, the more often such undesired behavior happens in practice, as the experiments in Section 6.2.3 will show.

Optimization As an alternative, Barzilay and Lapata (2006) propose to formulate set partitioning based on pairwise scores – in their case for aggregating sentences – as an optimization problem, in particular, as an ILP. Denis and Baldridge (2007) successfully applied that approach to coreference resolution. For every mention pair (m_i, m_j) let x_{ij} be a binary decision variable that indicates whether the pair is part of the desired relation. Then, finding the optimal variable assignments

$$\begin{aligned} \arg \max \quad & \sum_{m_i, m_j \in M^2} cr(m_i, m_j) x_{ij} + (1 - cr(m_i, m_j)) (1 - x_{ij}) & (6.9) \\ \text{s.t.} \quad & x_{ik} \geq x_{ij} + x_{jk} - 1 \quad \forall i, j, k \in [1, \dots, |M|] \text{ and } i \neq j \neq k \\ & x_i \in \{0, 1\} \end{aligned}$$

leads to a relation that maximally agrees with the pairwise predictions and is transitive due to the constraints. Note that the objective function gives high rewards if a pair with high

probability is included and if a pair with low probability is not, thus enforcing the solution to agree with both positive and negative classifications as much as possible.

For the example in Figure 6.1, the equivalence relation

$$\{(m_1, m_4), (m_2, m_5), (m_3, m_6), \dots\} \quad (6.10)$$

has an objective function value of 29.2, while the equivalence relation that is the transitive closure of Pos has a value of 28.4. Optimizing according to Equation 6.9 would therefore prefer the former, yielding the desired partitioning

$$C = \{\{m_1, m_4\}, \{m_2, m_5\}, \{m_3, m_6\}\} \quad (6.11)$$

in which three unique concepts are formed by the mentions.

Computationally, the optimal solution to an ILP can be found using cutting plane or branch and bound algorithms (Cormen et al., 2009) as implemented in off-the-shelf ILP solvers.⁴⁸ While the general problem of solving arbitrary ILPs is NP-hard (Cormen et al., 2009), the subclass of ILPs with totally unimodular constraint matrices, to which Equation 6.9 belongs, can be solved in polynomial time (de Belder and Moens, 2012). In our setting, a more serious problem than runtime complexity are the space requirements: The ILP in Equation 6.9 needs $\mathcal{O}(|M|^2)$ decision variables and $\mathcal{O}(|M|^3)$ constraints to ensure transitivity. For our data, where more than 10k mentions can be extracted from a document set, that leads to 100 million variables and 1 trillion constraints, which makes it difficult to fit the representation of the ILP into memory.

Delayed column generation (Desrosiers and Lübbecke, 2005) is a technique that can overcome memory limitations by using only a subset of the variables in an ILP. It iteratively adds an additional variable (column) based on the solution of a subproblem that determines which additional variable will improve the main problem's solution the most. De Belder and Moens (2012) apply this technique to coreference resolution. For that purpose, they use an alternative ILP formulation

$$\begin{aligned} \arg \max \quad & \sum_{i=1}^{2^{|M|}} v_i x_i \\ \text{s.t.} \quad & \sum_{i=1}^{2^{|M|}} b_{ji} x_i = 1 \quad \forall 1 \leq j \leq |M| \\ & x_i \in \{0, 1\} \end{aligned} \quad (6.12)$$

⁴⁸Popular implementations are CPLEX (<https://www.ibm.com/analytics/cplex-optimizer>), Gurobi (<http://www.gurobi.com>) or GLPK (<https://www.gnu.org/software/glpk>).

Algorithm 1 Beam Partitioning Search

Input: mentions M , positive predictions Pos , beam size k , max. depth d , max. breadth b
Output: equivalence relation $S \subseteq M^2$

```

1: function BEAMSEARCH( $M, Pos, k, d, b$ )
2:    $S_{best}, v_{best} \leftarrow Pos, \text{SCORE}(Pos)$  ▷ initial solution
3:    $B \leftarrow \{(S_{best}, v_{best})\}$ 
4:   for  $i$  in  $1, \dots, d$  do ▷ search until max depth
5:      $B' \leftarrow \emptyset$ 
6:     for  $(S, v) \in b$  do ▷ for each solution in beam
7:       for  $(m_i, m_j) \in \text{GETSMALLESTB}(S, b)$  do ▷ generate b neighbors
8:          $S' \leftarrow S \setminus \{(m_i, m_j)\}$ 
9:          $B' \leftarrow B' \cup \{(S', \text{SCORE}(S'))\}$ 
10:       $B \leftarrow \text{KEEPBESTK}(B', k)$  ▷ keep k best in beam
11:      if  $\text{BEST}(B) > v_{best}$  then ▷ remember the overall best solution
12:         $S_{best}, v_{best} \leftarrow \text{BEST}(B)$ 
13:   return  $\text{TRANCLOSURE}(S_{best})$ 
14: function SCORE( $S$ )
15:   return compute objective function of Eq. 6.9 for  $\text{TRANCLOSURE}(S)$ 

```

where decision variables x_i represent the $2^{|M|}$ possible subsets of M . b_{ij} denotes whether mention m_j is part of subset i and v_i is the sum of all predicted pairwise probabilities for mentions in subset i . Setting some decision variables to 1 amounts to combining different subsets of M , while the constraints ensure that these subsets are disjoint and thus form a partitioning of M . This ILP has $\mathcal{O}(2^{|M|})$ variables and $\mathcal{O}(|M|)$ constraints, but can be efficiently solved as only a small subset of the variables are typically needed in practice. The subproblem that determines these variables has $\mathcal{O}(|M|^2)$ variables and constraints and needs to be solved in turn with the growing main problem for every iteration until a solution is found. We refer the reader to de Belder and Moens (2012) for more details on the procedure and the formulation of the subproblem.

Local Search While the partitioning approach using the ILP has the advantage that it finds an optimal solution, applying it to large sets of mentions is challenging in terms of computation time and memory requirements. As an alternative, we therefore propose two local search algorithms that use the transitive closure partitioning as a starting point and try to improve that solution. A problem of the transitive closure over all positive classifications Pos , as we discussed earlier, is that it tends to group too many mentions together. Therefore, the basic idea of our search algorithms is to improve the partitioning by removing some pairs from Pos and then computing the transitive closure over the remaining pairs.

Algorithm 2 Greedy Partitioning Search**Input:** mentions M , positive predictions Pos **Output:** equivalence relation $S \subseteq M^2$

```

1: function GREEDYSEARCH( $M, Pos$ )
2:    $S_{best}, v_{best} \leftarrow Pos, SCORE(Pos)$ 
3:   for  $(m_i, m_j) \in TRANSREDUCTION(Pos)$  do ▷ edges potentially removed
4:      $S' \leftarrow S_{best} \setminus \{(m_i, m_j)\}$ 
5:      $v' \leftarrow SCORE(S')$ 
6:     if  $v' > v_{best}$  then ▷ if improved, continue with that solution
7:        $S_{best}, v_{best} \leftarrow S', v$ 
8:   return  $TRANSCLASURE(S_{best})$ 
9: function SCORE( $S$ )
10:  return compute objective function of Eq. 6.9 for  $TRANSCLASURE(S)$ 

```

As an example, consider the transitive closure solution given in Equation 6.11, which has an objective function value of 28.4, as the starting point for the search. Since Pos in our example has four positive classifications, the following four neighbor solutions can be created by removing one pair from Pos and then deriving the partitioning from the transitive closure of the remaining three pairs:

without (m_1, m_4)	$\{\{m_1\}, \{m_4\}, \{m_2, m_3, m_5, m_6\}\}$	26.4
without (m_2, m_5)	$\{\{m_1, m_4\}, \{m_2\}, \{m_3, m_5, m_6\}\}$	28.8
without (m_3, m_5)	$\{\{m_1, m_4\}, \{m_2, m_5\}, \{m_3, m_6\}\}$	29.2
without (m_5, m_6)	$\{\{m_1, m_4\}, \{m_2, m_3, m_5\}, \{m_6\}\}$	27.6

In this case, two improved partitionings are among the four neighbors, including the desired optimal solution. In general, we can recursively continue this search. Here, for each of the four solutions, three new neighbor solutions can be created by removing yet another pair. The full search tree spanned by this approach has a depth of $|Pos|$ and a branching factor starting at $|Pos|$ that decreases by 1 at each level.

Clearly, an exhaustive search of the full search space is not feasible for large sets of M . We therefore propose a **beam search** algorithm shown in Algorithm 1. Instead of traversing the whole search tree, it follows only the k highest-scoring solutions found at every level, it searches at most until a maximum depth d , i.e. it removes at most d pairs from Pos , and for every solution, it only considers the neighbors obtained by removing one of the b pairs with the lowest predicated probability. The best solution encountered in this subset of the search tree is returned. The runtime complexity is $\mathcal{O}(dkb|M|^2)$, as $d \cdot k \cdot b$ solutions are explored for which the transitive closure and scoring takes $\mathcal{O}(|M|^2)$ time. For the beam and the final partitioning, $\mathcal{O}(dk + |M|)$ additional space is required.

Partitioning	Optimization	Runtime	Space
Transitive Closure	no explicit	$\mathcal{O}(M ^2)$	$\mathcal{O}(M)$
Greedy Search	approximate	$\mathcal{O}(M ^4)$	$\mathcal{O}(M)$
Beam Search	approximate	$\mathcal{O}(dkb M ^2)$	$\mathcal{O}(dk + M)$
ILP w/ or w/o Column Generation	exact	polynomial	–

Table 6.1: Comparison of mention partitioning algorithms with regard to optimization behavior and time and space complexity. For ILPs, runtime and space depend on the solver’s implementation.

Since the beam search can still be prohibitively expensive, we propose a second, **greedy search** algorithm shown in Algorithm 2. It neither checks all possible neighbors to find the best pair to remove nor does it keep a beam of k -best solutions. Instead, starting with the initial solution, it iterates over all pairs in Pos only once, continuing with the revised solution if removing the pair is beneficial. Thus, it explores only a single path in the search tree. Further, it computes a transitive reduction of Pos in the beginning and only considers neighbors from these pairs to avoid removing edges that do not change the transitive closure. But, since multiple transitive reductions of a relation exist, the arbitrary choice of one of them also influences the explored search space. Algorithm 2 has a worst-case runtime complexity of $\mathcal{O}(|M|^4)$ and needs $\mathcal{O}(|M|)$ additional space for the solution.

Table 6.1 compares the different partitioning algorithms discussed in the preceding section. From top to bottom, the quality of the solution with regard to the objective function improves, while the runtime and space requirements increase. We note that the complexity behavior is a worst-case analysis. In practice, the set Pos tends to be much smaller than M^2 . We present an empirical analysis of this behavior in the next section.

6.2.3 Experiments

We conduct several experiments to evaluate the performance of the coreference classification and partitioning methods discussed in the previous section in order to determine which of them is most useful for CM-MDS.

Pairwise Mention Classification To train and test mention classification models, we derived a dataset of mention pairs with binary coreference labels from the training set of the EDUC corpus. In particular, we use the arguments of the top-50 most important propositions identified per training topic⁴⁹ as concept mentions and build all pairs. If two mentions have been connected by our annotators during the creation of a summary concept map, the

⁴⁹See Section 4.3.2 for details on the corpus creation process.

Features	Performance				Computation
	Pr	Re	F1	AUC	sec/100k
Exact Match	98.6	22.8	37.1	29.3	0.11
Lemma Match	98.0	24.5	39.2	30.6	0.82
Jaccard Coefficient	88.3	27.8	42.2	49.2	0.88
Edit Distance	83.4	32.1	46.4	53.0	0.46
Word2Vec	81.8	34.7	48.8	57.5	8.23
Rus-LSA	81.2	36.8	50.6	51.6	2.16
WN Rus-Resnik	82.1	36.9	50.9	54.2	5.40
WN Corley-Mihalcea	85.9	36.5	51.2	53.2	21.79
ADW	84.2	42.0	56.0	59.0	2057.03
All	81.8	45.2	58.2	63.6	2096.87
All w/o ADW + Corley	79.3	44.7	57.1	62.6	18.05

Table 6.2: Classification performance and feature computation time for pairwise mention classifications. The upper part reports results for models trained on just a single feature.

pair receives a positive label, all other pairs are negative examples. The resulting dataset consists of 17,500 mention pairs of which 1,218 (7%) are coreferent and 16,283 (93%) are not.

We train the log-linear classifier using Weka’s (Hall et al., 2009) implementation of logistic regression⁵⁰ and report metrics obtained with stratified 10-fold cross-validation on the training data. As features, the similarity measures discussed in Section 6.2.1 are used. We use Pilehvar et al. (2013)’s original implementation⁵¹ to compute ADW and the SEMILAR library⁵² (Rus et al., 2013) to compute the other WordNet-based similarities, both relying on WordNet 3.0. We also use SEMILAR to compute the LSA-based similarity with a 300-dimensional model provided with the library. As word embeddings, we use 300-dimensional Word2Vec embeddings trained on GoogleNews⁵³ as well as 50, 100, 200 and 300-dimensional GloVe embeddings trained on Wikipedia and Gigaword⁵⁴.

Table 6.2 shows the classification performance observed in this experiment in terms of precision, recall and F1-score on positive classifications and the area under the precision-recall-curve. In the upper part, we report results for models that were trained on only a single feature to understand the contribution of each similarity measure. Exact and lemma

⁵⁰We use the default regularization constant of 10^{-8} , as we have not seen an effect of using more or less regularization, presumably due to the small number of features.

⁵¹Commit 3298b6b, available at <https://github.com/pilehvar/ADW>.

⁵²Version 1.0, available at <http://deeptutor2.memphis.edu/Semilar-Web>.

⁵³Available at <https://code.google.com/archive/p/word2vec/>.

⁵⁴Available at <https://nlp.stanford.edu/projects/glove/>.

Dimension	Topic 1001	Topic 1042
Mentions	1,135	16,822
Unique Mentions	858	10,317
Pairs	367,653	53,215,086
Positive Pairs	699	22,074

Table 6.3: Size of the partitioning problem on the smallest and biggest document sets of the training part of EDUC. Unique mentions are mentions after applying exact and lemma matching.

matches show, as one would expect, a high precision but low recall. All other measures add additional recall by trading off some portion of precision, but generally improve the overall performance in terms of F1-score and AUC. Interestingly, WordNet-based similarities work surprisingly well on our dataset and outperform the more recent approach of word embeddings. The ADW measure outperforms all others by a substantial margin. Note that we did not include results for GloVe embeddings, which are outperformed by Word2Vec, and also not the alternative WordNet-based measures described in Section 6.2.1, which are outperformed by Resnik (1995)’s approach.

The last column of Table 6.2 reports the necessary time to compute the similarity measures for 100,000 mention pairs. In general, better performing features tend to be more expensive to compute. As the number of mention pairs that have to be classified grows quadratically with the length on the input documents, long computation times can severely limit the applicability of a classifier to our task. Therefore, the ADW similarity, despite showing the best performance, is not very useful due to its orders of magnitude larger computation time. Corley and Mihalcea (2005)’s similarity, while being much cheaper than ADW, still adds a large amount of computation. A good trade-off between classification performance and feature computation time seems to be a model combining all features except the two expensive ones. As the lower part of Table 6.2 shows, excluding them leads to a drop in F1-score and AUC of just one point.

Mention Partitioning To compare the different partitioning algorithms discussed before, we apply them to the concept mention sets obtained for document sets of the training part of Educ. We extract them using an OIE-based method described in Section 6.5. For efficiency, we first group mentions by exact and lemma matching⁵⁵ and denote with M this revised mention set. All mention pairs for M are then classified with the log-linear model discussed in the previous section using only five similarities — neither the two expensive

⁵⁵Note that for exact and lemma matching, a mention can be directly mapped to its subset without comparisons to other mentions. Thus, using a hash map, this partitioning can be implemented in linear time. The resulting reduced set of mentions then also decreases the number of pairs in the subsequent steps.

Algorithm	Topic 1001			Topic 1042			
	Time	Value	$ C $	Time	Value	$ C $	
Trans. Closure	0.2s	0.9261	495	33.7s	0.7383	4078	
Greedy Search	3.4s	0.9659	664	2h 33m	0.9612	7741	
Beam Search	k=1, d=10, b=100	10s	0.9367	505	4h 30m	0.7394	4085
Beam Search	k=1, d=500, b=100	3m 39s	0.9660	669	9d 10h	0.7458	4130
Beam Search	k=5, d=500, b=100	17m 3s	0.9660	669	44d 13h	0.7458	4130
ILP w/ CG		1d 8h	0.9661	621	<i>Out of Memory</i>		
ILP	<i>Out of Memory</i>			<i>Out of Memory</i>			

Table 6.4: Runtime and optimization results for mention partitioning on the smallest and biggest document sets of the training part of EDUC. Objective function values are normalized by pairs.

ones nor the already applied ones — as features. The resulting pairs and their classification probabilities are the input for the partitioning algorithms. We solve ILPs using CPLEX⁵⁶ and use our own implementations of the other approaches. All experiments are carried out on a compute server with 500 GB of memory and Intel Xeon ES-2620 2.1GHz processors of which CPLEX uses 24 cores and the other algorithms only a single core.

Table 6.3 shows the size of the concept grouping problem for the smallest and biggest topics in the training part of EDUC. Note that while topic 1042 has only ten times as many mentions as 1001, the resulting number of pairs is larger by a factor of 150, illustrating the scalability problem of pairwise comparisons. However, as we showed before, our partitioning algorithms have theoretical runtimes that grow even worse than quadratically, which is in line with the partitioning results in Table 6.4. On the smaller document set, we could obtain an optimal solution with the column generation variant of the ILP, whereas the plain ILP was not solvable with the available memory. On the bigger document set, neither variant succeeded. The transitive closure partitioning, on the other hand, while being fast, leads to the aforementioned lumping effect as illustrated by the small number of resulting concepts and the low objective function values. Our two heuristic search algorithms make a better trade-off between runtime and solution quality, finding partitionings close to the optimal solution on the smaller topic substantially faster than the ILP. Comparing between them, we observe that even the fastest instantiation of beam search, which searches to a depth of only 10 and thus yields only marginally better solutions than transitive closure, takes longer than the greedy search. Thus, overall, finding a mention partitioning with greedy search appears to be the best approach on our data.

⁵⁶Version 12.7, available at <https://www.ibm.com/analytics/cplex-optimizer>.

Conclusion Based on the experimental results reported in this section, we conclude that a pairwise coreference classifier using the five selected similarities as features in combination with the greedy partitioning search offer the best combination of performance and runtime. Thus, we will include that setup in our CM-MDS pipeline described in Section 6.5 and assess them in an end-to-end task-level evaluation. With regard to mention partitioning, the experiments also showed that even the greedy search still requires more than 2.5 hours to process the biggest topic in our dataset, limiting the applicability to concept grouping in practice. In Section 6.5.5, we therefore revisit these scalability issues.

6.3 Importance Estimation

The goal of importance estimation is to distinguish between more and less important concepts and relations to be able to later select the best subset of them for the summary concept map. In this section, we focus on estimating importance for concepts. Following our formalization of the subtask given in Section 3.3.6, we want to have a function $\nu : C \rightarrow \mathbb{R}$ that assigns a score to every concept, with higher values indicating higher importance.

As we pointed out in the introduction of this chapter, previous work on concept map mining approached importance estimation, if at all, in an unsupervised fashion relying on a single feature indicating importance. However, many different strategies for supervised importance estimation have been explored for traditional MDS (see Section 2.3.2). In this section, we therefore explore how such methods can be transferred to CM-MDS and which of them perform best. In particular, we compare different formalizations of the supervised machine learning problem and evaluate a large set of features for importance. In the end-to-end experiments in Section 6.5, we then compare the supervised model developed in this section against the unsupervised methods suggested in previous work.

6.3.1 Modeling Approaches

Learning the scoring function ν can be approached with different types of machine learning that learn by minimizing different loss functions and require, in addition to the inputs C , different types of labels. For all approaches, we assume that a feature representation $\phi(c)$ of each concept is available.

Regression The most natural approach is to treat the problem as a regression problem. For MDS, recent works pursuing that direction are for example Zopf et al. (2018b), Cao et al. (2015) or Hong and Nenkova (2014). Pairs $(c, \nu(c)^*)$ of concepts with their true importance scores are needed in such approaches. One can then learn a linear model $w^T \phi(c) + b$ with feature weights w and a bias term b by minimizing the squared distance between predictions and true scores, i.e. using linear regression, or by minimizing a hinge loss as in SVMs.

Kernelized SVMs or neural networks also allow learning non-linear regression models. In our setting, the predicted regressand can be used as the importance estimate.

Binary Classification Less natural, but also very common, is to approach importance estimation as a binary classification problem. Recent examples for MDS are Cheng and Lapata (2016) and Yang et al. (2017). A binary label for every object, typically denoting “important” and “not important”, is necessary. From standard MDS datasets, such labels can be derived by comparing source documents against the reference summaries and assigning a positive label to every element also occurring in the summary. Any binary classification model, such as a log-linear model, an SVM or a neural network, can be used. Since classifiers predict classes instead of scores, their probabilities can serve as importance estimates.⁵⁷

Ranking A third approach to the supervised importance estimation problem is to learn to rank concepts in the correct order. Such methods, often referred to as *learning to rank*, are very popular in information retrieval (Liu, 2009), but have not received much attention in the summarization community so far. The main motivation behind this approach is that for many problems, including summarization, we are only interested in a ranking, i.e. whether one object is more important than another, but not in specific importance scores. In contrast to regression methods, ranking methods therefore minimize a loss expressing the errors in the ranking that the model derives for a set of objects. For instance, Ranking SVMs (Joachims, 2002) learn a linear model $w^T\phi(c) + b$ by minimizing the number of incorrectly ordered pairs obtained when ordering elements according to the model. Shen and Li (2011) apply Ranking SVMs to MDS and observe improvements over SVM-based regression models. Such predictions can also be used as importance estimates for concepts.

Structured Prediction Importance estimation can also be modeled as structured prediction. The idea here is that we are ultimately not just interested in predicting the importance of concepts but in predicting a summary concept map, i.e. a structured output. Prediction problems with structured outputs are common in NLP and many models to approach them have been developed (Daumé III, 2006, Chang et al., 2017). In the area of MDS, Liu et al. (2015) and Li et al. (2016a) applied them to graph-based abstractive summarization (see Section 2.3.2). As the supervision signal, references for the structured outputs are needed and, given the reference and a prediction, a structure-level loss such as the structured perceptron, hinge or ramp loss (Liu et al., 2015) can be computed. While such models still learn to score single concepts, learning that function considering the later use of the scores can lead to models that perform better end-to-end.

⁵⁷Note that for binary classifiers predicting probabilities, importance estimates will be in $[0, 1]$, whereas non-probabilistic binary classifiers, such as SVMs, predict unbounded real-valued scores as in regression.

In the next section, we discuss possible feature representations $\phi(c)$ for concepts and then use them to carry out a range of experiments to analyze the different possibilities to model the importance estimation problem in Section 6.3.3.

6.3.2 Features

For importance scoring, we explore a broad range of features that are commonly used for summarization. We describe the groups of features and their motivation in the following:

6.3.2.1 Common Summarization Features

Frequency Frequencies, starting with the work of Luhn (1958), are the most popular features to determine important elements. To estimate the importance of concepts, we evaluate the following frequency features:

- absolute frequency of a concept, i.e. the number of its mentions $|C|$
- relative frequency of a concept with regard to the total document set
- the fraction of documents containing a concept

In addition, we use inverse document frequencies as a measure of how distinctive terms are as done by TF-IDF. However, instead of only relying on the documents given for a topic, we use term counts computed on a much larger background corpus⁵⁸ which were found to correlate well with document frequencies (Klein and Nelson, 2009). Using them, we obtain the following additional features:

- minimum, maximum and average of the log-scaled inverse document frequency of the terms in a concept's label
- CF-IDF, i.e. the relative concept frequency as defined above multiplied with the either the min, max or average variant of inverse document frequency defined before

Position The second most popular feature for summarization, introduced by Edmundson (1969), is the position of an element in the source documents. We compute the relative position of a concept mention m in a document d independent of document length as $rel_pos(m) = 1 - pos(m)/|d|$ such that a mention at the beginning is close to 1 and at the end close to 0. Based on that, our position features are:

- minimum, maximum and average relative positions among all mentions
- relative position of the mention used as the label
- position spread, i.e. the difference between the first and the last position

⁵⁸We use the Google Web 1T 5-gram corpus, available through LDC at <https://catalog.ldc.upenn.edu/products/LDC2006T13>, which provides n-gram counts computed on a corpus of one trillion web pages.

Length The length of a concept can also be indicative of a concept’s importance and has been used for instance by Li et al. (2016a). In addition, length can also be an indicator of concept extraction quality in the case of overly long mentions or labels, which we would not want to be part of the summary. Similar to the position features, we use the minimum, maximum and average length among all mentions, the length of the selected label and the spread between minimum and maximum, all computed on a token and character level.

Part-of-Speech Similar to length, the presence of different part-of-speech categories can be indicative of importance and quality. In MDS, part-of-speech features are used regularly, for instance by Li (2015) and Hong and Nenkova (2014). We use two binary features for each part-of-speech category in the PennTreebank tagset that indicate whether a token of that category occurs in the concept’s label and whether the token is the label’s head (i.e. the root node in a dependency parse of the label).

Annotations Similar to previous work on summarization, such as Berg-Kirkpatrick et al. (2011), Li et al. (2013), Hong and Nenkova (2014) and Li et al. (2016a), we include additional features based on automatic linguistic annotations of the concept label:

- whether named entities of type person, organization or location are present in the concept’s label or in its head token
- the depth of the concept label’s head token in the dependency parse of the sentence it was extracted from in the source documents
- whether all or some of the characters in the concept label are uppercase
- the absolute and relative number of stopwords in the concept label

Topic Relatedness As some concepts can be prominent in a document cluster but unrelated to its topic, we introduce additional features capturing the relatedness between a concept and the topic. Note that all corpora created for CM-MDS in Chapter 4 have short descriptions of the topic of each document set, such as “alternative ADHD treatments” or “student loans without credit history”.⁵⁹ We compute three features that measure the semantic similarity between a concept’s label and the topic description, using the Word2Vec, WordNet Rus-Resnik and Jaccard measures introduced in Section 6.2.1.

6.3.2.2 Extended Features

In addition to the set of features described before, which have been commonly used in many summarization systems, we also explore a few less common features as well as features specifically designed for our concept map task.

⁵⁹Despite having a topic description, our corpora do not constitute a classic query-focused MDS scenario, since there are no documents in the document sets that are not related to the topic.

Graph TextRank (Mihalcea and Tarau, 2004) and LexRank (Erkan and Radev, 2004), which are popular MDS approaches, build a graph representing the relations of sentences or words to derive an importance metric from that structure. In our case, we already have a graph available, formed by the grouped concepts C and relations R between them. Therefore, we can easily use its structure to derive additional features for concepts such as how centrally a concept is positioned in the graph. Interestingly, the abstractive MDS systems that build graphs as intermediate representations (Liu et al., 2015, Li, 2015, Li et al., 2016a) use only little or none of such features.

Given the graph $G = (C, R)$, we compute the following features for a concept $c \in C$:

- in and out degree of the node c
- fraction of nodes to which c is connected
- PageRank (Page et al., 1999), the centrality measure used in TextRank and LexRank
- betweenness, closeness, eigenvector and katz centrality (Koschützki et al., 2005), alternative standard graph metrics to characterize the centrality of a node in a graph

As we mentioned in Section 2.3.1.5, Reichherzer and Leake (2006) evaluated models specifically designed for the structural analysis of concept maps. The first, HARD, is a linear combination of upper and lower node scores proposed by Cañas and Leake (2001), characterizing a concept’s distance to the main concept of a map, and hub and authority scores as provided by Hyperlink-Induced Topic Search (HITS) (Kleinberg, 1999), an alternative link analysis algorithm to PageRank proposed at the same time. The second measure, connectivity root-distance (CRD), combines a concept’s in and out degree with its distance to the main concept. We include HARD and CRD as well as the underlying authority, hub, lower and upper node scores in our feature set.

And finally, we follow the work of Tixier et al. (2016) who show that graph degeneracy is a useful feature to identify keyphrases. We use the graph core number and core rank as proposed by them, two metrics based on the membership of a node to a k -core of the graph, the maximal subgraph in which all nodes have a degree of at least k .

Extraction Since the concept and relation mentions are extracted from the source documents in preceding steps, we also rely on features obtained from the extraction process. In the pipeline described in Section 6.5, the extraction is done using OIE. We include the confidence with which an extraction was made as well as the type of argument a concept mention is part of – simple, spatial or temporal – as additional features. However, intuitively, they offer little signal for importance but are rather indicative of extraction quality.

Word Categories As suggested in recent work by Yang et al. (2017), dictionary-based features that capture general properties of words such as concreteness, familiarity or imagery can also be helpful for summarization. In particular, whereas all other features discussed

so far are mainly based on the content of the source documents, these features bring in additional external knowledge about how certain words are generally used and perceived by people. As we argued in Section 3.3.6, such information can be crucial to fully capture the human notion of importance. We use the MRC Psycholinguistic Database (Coltheart, 1981), which scores words for concreteness, familiarity, imagability, meaningfulness and age of acquisition, the LIWC dictionary, which groups words into 65 different categories, and an additional, bigger list of concreteness values for words by Brysbaert et al. (2014).

6.3.3 Experiments

To assess which of the different machine learning approaches and features are most useful to estimate the importance of concepts for CM-MDS, we compare their performance on identifying the most important concepts among all extracted concepts.

Experimental Setup Similar to the partitioning experiment in Section 6.2.3, we use a dataset obtained by running the previous steps of CM-MDS on EDUC. Specifically, we use the training set of EDUC, run concept and relation mention extraction and group concepts using exact matches followed by the greedy search partitioning based on coreference predictions. For the 15 training topics, that process yields on average 3,600 concepts per topic.

For classification experiments, we assign binary labels to the concepts based on whether they match one of the 25 concepts in the reference concept map for their topic, leading to dataset of 54,033 instances of which 366 are positive. To obtain regression labels, we use the crowdsourced importance scores (see Section 4.3.2) and assign a concept the maximal importance score its propositions received. Since not all propositions were part of the crowdsourcing, we can only do that for a subset of the concepts, resulting in a dataset of 15,966 concepts with importance scores from 1 to 5. For ranking, we can reuse the binary labels to train a model to rank all positive concepts before the negative ones for each of the topics. Thus, we also have 54,033 instances available.

We estimate the performance of different models with leave-one-out cross-validation over the 15 training topics. For each topic, we first run a grid search over different hyper-parameters on the remaining 14 training topics, determining their performance with an inner 10-fold cross-validation. We then retrain a model with the best hyper-parameters⁶⁰ and use it to estimate the importance of the test topics' concepts. Similar to concept selection in Section 5.2.2.5, we compute r-precision by comparing the 25 concepts found to be most important against the reference concepts and average over all 15 test folds.

As models, we include linear and logistic regression from Weka (Hall et al., 2009), SVMs for classification (C-SVC) and regression (ϵ -SVR) using LibLINEAR (Fan et al., 2008) for

⁶⁰We select the hyper-parameters with highest F1-score (classification), lowest root mean squared error (regression) or highest ranking accuracy (ranking) as estimated by the inner cross-validation.

linear kernels and LibSVM (Chang and Lin, 2011) for a version with radial basis function kernels⁶¹. We further include Ranking SVMs as implemented in Dlib (King, 2009).⁶² We also performed preliminary experiments with structured prediction, using a structured perceptron (Collins, 2002), but did not observe competitive results, presumably due to too little training data, as we can only train on 15 (graph-level) instances in that setup instead of 54k and 16k (concept-level) instances in the other settings. For classification, we subsample negative instances, as the dataset is highly skewed, and, similar to previous work on summarization (Li et al., 2016a), discretize the mostly Zipf-distributed continuous features into bins. We use five equal-frequency bins per feature in the regression setup and rely on Weka’s implementation of the minimum description length principle to find optimal bins in the classification setup. All models use L2-regularization. During grid search, we tune the regularization constant of each model, the SVM’s kernel parameter, the ϵ of its regression variant and the amount of subsampling.⁶³ All source documents are preprocessed with Stanford CoreNLP⁶⁴ (Manning et al., 2014) to obtain the part-of-speech, named entity and dependency annotations required for some of the features. Graph metrics were computed with standard algorithms as implemented in networkx.⁶⁵

Features Before looking at importance estimation results, we analyze the usefulness of different features. In Table 6.5, we show features with the highest Pearson correlation, both positive and negative, per group measured against the crowdsourced importance scores that we use to train regression models. The strongest features correlating with importance are, in line with previous work on summarization, from the frequency group. But the graph-based features, in particular core numbers, degrees and centrality measures, are almost as good. The HARD model and its components show low correlations (< 0.1) on our dataset. Note that position features are less useful than suggested by Table 6.5, as only position spread, which correlates strongly with the number of mentions of a concept and thus frequency, has a high correlation in that group. Another useful feature seems to be concreteness values obtained from external lexicons.

With regard to identifying unimportant concepts, the highest negative correlation with importance can be observed for the length of a concept’s label. This is in line with our intuition, since concepts with very long labels tend to be very specific and detailed and are therefore less useful for a summary. The number of stopwords and the presence of certain parts-of-speech, e.g. determiners, also correlate with unimportance. We believe that this is

⁶¹Substantially higher cross-validation runtimes did not allow us to use RBF-SVMs in the regression setup.

⁶²WEKA version 3.8.1, LibLINEAR version 2.20, LibSVM version 1.0.10 and Dlib version 18.17.100.

⁶³Ranges used for grid search: subsampling to 2%, 5%, 10%, 30%, 50% or 100% (no sampling) of full size, regularization constants of 1E-2, 1E-1, 1, 1E+1, 3E+1, 1E+2, 3E+2, 1E+3, γ -values for radial basis function kernels of 1E-3, 3E-3, 1E-2, ϵ -values for ϵ -SVR of 1E-4 to 1E-0.

⁶⁴Version 3.6.0

⁶⁵networkx version 1.11 (<https://networkx.github.io/>).

Group	#	Most Positive		Most Negative	
		Feature	r	Feature	r
Frequency	9	document frequency	0.258	max.token IDF	-0.054
Graph	17	core number	0.246	lower node score	-0.088
Position	5	position spread	0.218	min. position	-0.098
Lexicon	85	min. concreteness	0.135	has function word	-0.071
Length	10	token spread	0.098	number of tokens	-0.121
Part-of-Speech	78	head has NNS tag	0.064	label has DET token	-0.085
Topic	3	jaccard similarity	0.059	word2vec similarity	-0.021
Extraction	4	is temporal argument	0.019	is simple argument	-0.016
Annotations	13	head is person NE	0.014	number of stopwords	-0.097

Table 6.5: Pearson correlation between features and true importance scores. Shown are the two features with highest positive and negative correlation per group, using the regression dataset.

due to the fact that there are some concepts with noisier labels in the dataset among the less important ones. Using these features, we can lower the chance that they will be included in the summary concept map.

Modeling Approaches Table 6.6 shows the concept selection performance using different classification, regression and ranking models. The list-wise evaluation compares, as described before, the 25 concepts with the highest estimated importance against the concepts of the reference summary concept map. Unfortunately, as the differences between the modeling approaches are marginal and not significant, we cannot draw any conclusions regarding the superiority of any approach from these results. We also performed a second, graph-wise evaluation in which the importance estimates for all concepts are used to select the best summary subgraph with 25 concepts, using the methods proposed in Section 6.4, and then compared the concepts in that map against the ones in the reference map. Such an evaluation can potentially show different results, as importance estimation functions learned with a classification or ranking loss, as opposed to regression, do not necessarily produce scores whose differences are meaningful and comparable, which can be problematic when these scores are used to calculate aggregated measures during selection. However, the results are similar and differences between approaches are even smaller.

Overall Performance and Errors While the primary purpose of this experiment is to compare features and modeling approaches, it also gives insights into how difficult the importance estimation task is in general. With precision scores between 10% and 13%,

Model	List		Graph	
	R-Precision	p-value	R-Precision	p-value
Classification				
Logistic Regression	10.93	.0967	11.73	.1826
Linear SVM	11.47	.2100	11.47	.1284
Kernel SVM	10.13	.0157	10.40	.0381
Regression				
Linear Regression	13.07		13.33	
Linear SVM	11.73	.0703	12.27	.2422
Ranking				
Ranking SVM	12.27	.3877	12.80	.5176
Upper Bound	56.00		46.67	

Table 6.6: List- and graph-wise concept selection performance with different importance estimation models evaluated with leave-one-out cross-validation. The upper bound is determined by the recall of concept extraction. P-values are computed with a permutation test.

only 3 out of the 25 selected concepts match a concept in the reference map, which leaves a lot of room for improvement. We expect the achievable performance for this subtask to be in the range of 30% to 40%, since 47% is the hard upper bound due to extraction performance (see Table 6.6) and the human agreement on importance annotation is around 0.8 (see Section 4.3.2).⁶⁶ Following that reasoning, improvements of 20 to 30 percentage points should still be possible on this subtask.

Looking at the selections made by the models, we found a large fraction of the correct selections (which are only 3 per topic on average) are a small number of concepts that occur in several reference concept maps. For instance, maps for many topics have the concepts *parents* and *children*, which the model also consistently selects. Important concepts which the model misses to select are in particular those that occur in only one topic. During the manual inspection of the selections, we also observed that some selected concepts do not match any reference concept exactly but are very close, e.g. *ADHD remedies* might have been selected while the reference has *ADHD medication*. While this is not rewarded by the exact matching used to compute precision here, it is considered by the ROUGE and METEOR metrics used in task-level evaluations. In that sense, the precision and upper bound reported here characterize importance estimation performance rather pessimistically.

⁶⁶Note that the human agreement for the importance annotation was measured by correlation. In contrast to percentage agreements, which are often directly compared to F-scores to define upper bounds, there is no direct connection between correlation and the precision metric used in this experiment.

Conclusion Based on our experiments, we observe that there is no significant difference between modeling importance estimation as regression, classification or ranking on our data. While one could probably find significant differences using larger datasets, the fact that they can only be observed — if at all — with more data shows that they are presumably rather small and will thus have only a small impact on the overall task-level performance. With regard to features, the graph-based measures that we included in addition to traditional summarization features seem to be particularly useful for our task. Nevertheless, performance on importance estimation is still limited and should be further improved. We suspect that adding more external knowledge on what people generally consider to be important can be particularly helpful. In the task-level experiments in Section 6.5, we also assess how well the supervised model with the current features performs against the unsupervised methods suggested in previous work.

6.4 Concept Map Construction

Given the graph $G = (C, R)$ of extracted and grouped concepts and relations together with estimates of their importance, the final subtask in CM-MDS is to create a summary concept map. In Section 3.3.7, we formalized this as a subgraph selection problem under constraints ensuring connectedness and the size limit, leading to the optimization problem in Equation 3.1 and its simplified version in Equation 3.2. The space of possible solutions is large for most problem instances and there is no straightforward way that directly determines the best solution. In addition, almost none of the previous works on concept map mining focused on this subtask and proposed any approaches for it, with the only exception being Zubrinic et al. (2015)’s heuristic subgraph selection method.

To close this gap for CM-MDS, we propose a formulation of the subgraph selection problem as an ILP. That has the advantages that available ILP solvers can be used in practice and that we are guaranteed to find the optimal solution. In the following section, we will present our ILP formulation. As we will show in the subsequent experiments, solving the ILPs for our problem instances is in this case, as opposed to the partitioning problem discussed earlier, computationally feasible.

6.4.1 Integer Linear Programming Approach

We focus here on the optimization problem from Equation 3.2 which expresses the selection solely in terms of concepts and their estimated importance. While one could also include the selection of relations into the ILP in order to solve the problem defined in Equation 3.1, we leave that extension for future work.

Let x_i be a binary decision variable that represents whether concept $c_i \in C$ is part of the selected subgraph. Then, the objective function can be written as⁶⁷

$$\max \sum_{i=1}^{|C|} x_i \nu(c_i) \quad (6.13)$$

$$x_i \in \{0, 1\} \quad \forall i \in C \quad (6.14)$$

while the following constraint ensures that the subgraph obeys the size limit:

$$\sum_{i=1}^{|C|} x_i \leq L \quad (6.15)$$

Ensuring that the selected subgraph is also connected is a bit more intricate. A common approach to express such a constraint in an ILP are so-called commodity flow variables and, more specifically, the single commodity flow formulation for the minimum spanning tree problem proposed in the operations research community by Magnanti and Wolsey (1994). It has been successfully used in ILPs addressing dependency parsing (Martins et al., 2009), sentence compression (Thadani and McKeown, 2013) and abstractive summarization (Liu et al., 2015, Li et al., 2016a). Let f_{ij} be a non-negative integer variable capturing the flow from concept c_i to c_j . We introduce flow variables for concept pairs with a relation in R . The constraints

$$f_{ij} \leq x_i \cdot |C| \quad \forall (i, j) \in R \quad (6.16)$$

$$f_{ij} \leq x_j \cdot |C| \quad \forall (i, j) \in R \quad (6.17)$$

$$\sum_i f_{ij} - \sum_k f_{jk} - x_j = 0 \quad \forall j \in C \quad (6.18)$$

$$f_{ij} \in \mathbb{N} \quad \forall (i, j) \in R \quad (6.19)$$

enforce that flow can only move between concepts that are selected (6.16 and 6.17) and a selected concept consumes one unit of flow (6.18). Further, let $i = 0$ be a virtual root node and e_{0i} a virtual edge from the root to each concept. The additional constraints

$$|C| \cdot e_{0i} - f_{0i} \geq 0 \quad \forall i \in C \quad (6.20)$$

$$\sum_{i=1}^{|C|} e_{0i} = 1 \quad (6.21)$$

$$\sum_{i=1}^{|C|} f_{0i} - \sum_{i=1}^{|C|} x_i = 0 \quad (6.22)$$

$$e_{0i} \in \{0, 1\} \quad \forall i \in C \quad (6.23)$$

$$f_{0i} \in \mathbb{N}_0 \quad \forall i \in C \quad (6.24)$$

ensure that only one virtual edge can be active (6.21), that the virtual node can only send flow over this active edge (6.20) and that the total amount of flow sent from the root cannot

⁶⁷To simplify the notation, we write $i \in C$ instead of $i \in \{1, \dots, |C|\}$ and correspondingly for R .

exceed the number of selected concepts (6.22). As a consequence, if n concepts are selected, n units of flow are sent from the virtual root over the edges of the graph and each selected concept consumes one of them. This is only possible if the selected subgraph is connected. Equivalently, one can think of it as the edges with flow larger than zero forming a spanning tree of the selected subgraph that is rooted in the additional virtual node.

An important detail for the optimization is the range of the importance estimates. If some concepts receive negative scores, the objective can be improved by excluding them from the subgraph. As a result, some part of the size budget might remain unused although additional connected concepts would be available. In order to avoid that, we can simply shift all importance scores into the positive range, formally, by deriving ν' as

$$\nu'(c_i) = \nu(c_i) - \min\{\nu(c_j) \mid c_j \in C\} \quad (6.25)$$

and then using ν' in the ILP. However, if negative scores are only assigned to concepts that should in no case be part of the summary, the default behavior might actually be desired.

We take several measures to ensure that the ILP can be efficiently solved for the problem instances of CM-MDS. First, the above ILP formulation is already much more efficient than the one proposed by Li et al. (2016a) for MDS, which is the most similar ILP in related work. While ours requires $\mathcal{O}(|C| + |R|)$ variables and constraints, their formulation uses two variables per pair of nodes for the connectivity constraint, resulting in $\mathcal{O}(|C|^2)$ variables and constraints. For sparse graphs, where $|R| \ll |C|^2$, this leads to much smaller ILPs.

Second, we leverage the fact that G is typically disconnected. Since a connected subgraph has to be completely in one of the connected components of G , we first identify these components and solve separate ILPs for each of them. These smaller ILPs can usually be solved faster than a single large one. And third, processing G component by component also allows us to completely skip some of them. Starting with the biggest component, we can keep track of the best objective function value so far. If the next component has a total concept score less than that value, none of its subgraphs can be a better solution. And if the component consists of less concepts than the limit, we can also directly use the component instead of selecting a subset. With these measures, as we show in the experiments, the ILP can be efficiently solved for the problem sizes in the EDUC corpus.

6.4.2 Experiments

To verify the effectiveness and efficiency of our proposed subgraph selection, we conduct an experiment that compares it against heuristic selection and alternative ILP formulations.

Experimental Setup We use the same data as for the concept importance estimation experiment (see Section 6.3.3), namely concepts extracted and grouped from the training topics of EDUC. To evaluate subgraph selection independent of importance estimation, we do

	METEOR				ROUGE			
	Pr	Re	F1	p	Pr	Re	F1	p
EDUC								
ILP	23.32	27.52	25.16		26.09	23.93	24.74	
Heuristic	18.28	25.15	21.13	.0003	17.52	21.97	19.34	.0014
WIKI								
ILP	29.04	26.76	27.73		29.08	18.79	22.54	
Heuristic	24.45	24.46	24.83	.0051	24.06	17.39	19.57	.0093

Table 6.7: Evaluation of summary concept maps obtained with the proposed ILP and heuristic selection. Inputs are graphs created by automatic extraction and grouping in combination with gold importance scores. P-values are computed with a permutation test comparing F1-scores.

not use a trained model but the gold scores derived as training labels in Section 6.3.3. We create a second dataset based on WIKI with the same approach. On both datasets, we evaluate the selected subgraphs by comparing them against the reference concept maps with the metrics proposed for CM-MDS in Section 3.5.2. ILPs are solved with CPLEX⁶⁸ on a compute server with 500 GB of memory and 24 Intel Xeon ES-2620 2.1GHz cores.

As a baseline for our proposed approach, we implement a greedy **heuristic** similar to Zubrinic et al. (2015): Given the graph of scored concepts, it starts with the most important one and selects the best neighbor (by score, breaking ties by the node’s degree) until the size limit is reached. While this procedure ensures that the selected subgraph is valid, i.e. not too big and connected, it is not necessarily, in contrast to the ILP, the best subgraph with regard to our objective function. As a second baseline, we include an alternative formulation of the subgraph selection ILP obtained by transferring Li et al. (2016a)’s ILP for MDS to our task. The main difference is that it uses a quadratic number of variables to represent the presence or absence of all possible edges and the flow along them. While that has implications for its efficiency, it does of course also find an optimal subgraph.

Results Table 6.7 shows the results of this experiment. As expected, our ILP approach selects better subgraphs as summaries and the results on both datasets and in both metrics show that the difference between them and the summaries obtained with the heuristic are substantial and significant. Note that while the ILP finds the best solution to the optimization problem by definition and is in that sense already known to be superior to the heuristic, this experiment verifies that the best solution to the optimization problem is also indeed a good solution for the CM-MDS task in terms of being closer to the reference map.

⁶⁸Version 12.7, available at <https://www.ibm.com/analytics/cplex-optimizer>.

Method	ILP Size		Runtime
	Variables	Constraints	sec
(Li et al., 2016a)	37,273,062	74,530,095	2670.61
by component	25,810,465	51,607,172	999.25
Our ILP	21,596	31,129	7.31
by component	17,973	26,484	5.61

Table 6.8: Comparison of ILP sizes and runtimes on average per topic for subgraph selection on EDUC with our ILP and the alternative formulation of Li et al. (2016a).

In table Table 6.8, we compare ILP sizes and the time required to solve them. Although the differences between our ILP formulation and the one by Li et al. (2016a) are small, they have a large effect in practice, resulting in orders of magnitude smaller problems and faster runtimes. Identifying connected components and selecting subgraphs for each of them separately further improves the efficiency of both ILP approaches. On the document sets of EDUC, with on average over 100,000 tokens, that allows us to select a summary subgraph in just a few seconds, which is not possible with Li et al. (2016a)’s formulation.

Conclusion Based on these experimental results, we conclude that selecting summary subgraphs with our proposed ILP is effective and can also be done efficiently on our copora. We will therefore include it in our CM-MDS pipeline described in the next section and assess it in an end-to-end task-level evaluation.

6.5 Full Pipeline Experiments

Based on the proposed methods and presented experiments for different subtasks of CM-MDS in the previous sections and chapters, we now present a pipeline that can approach the task in its full scope. We carry out an experimental comparison against a range of baselines to assess the effectiveness of our proposed methods and identify remaining challenges.

6.5.1 Pipeline Overview

Concept and Relation Mention Extraction To extract concept and relation mentions from text, we rely on OIE and use, as in the experiments in Section 5.2.2, OpenIE4 (Mausam, 2016). In contrast to the earlier experiments, in which we evaluated the usefulness of OIE extractions in its raw form, we also apply a set of simple post-processing rules to ensure that all extractions consist of mentions of concepts and relations as needed for a concept map. Specifically, we filter the set of all extractions with two constraints to ensure that

the arguments of the extractions are meaningful concept mentions: First, an argument has to contain at least one noun token, and second, it cannot be longer than ten tokens. This removes overly long arguments that are full clauses rather than mentions of concepts and reduces the total number of candidates from which the summary-worthy ones need to be selected in the subsequent steps.

In addition, we apply three rule-based post-processing steps that refine the extractions in order to increase recall. First, using coreference chains found by off-the-shelf coreference resolution software, in our implementation Stanford CoreNLP 3.6.0 (Manning et al., 2014), we resolve pronominal anaphora in arguments of extractions and replace the pronoun with the representative mention of the coreference chain. Without that step, such extractions are not meaningful without context and could therefore not be used in a concept map.

Second, if an argument is a conjoining construction, as indicated by *conj*-edges in a dependency parse, we introduce separate extractions for each conjunct:

(Caffeine - works with - young children and teens)

would be split into two extractions

(Caffeine - works with - young children)

(Caffeine - works with - teens)

And third, if the second argument starts with a verb, as in the following example,

(Herbal supplements - have been used to - treat the symptoms of ADHD)

we move that verb and subsequent prepositions to the predicate. In the example, the predicate is extended to *have been used to treat*, reducing the second argument to *the symptoms of ADHD*. Given the resulting set of filtered and revised OIE extractions, we use all their arguments as concept mentions M and the predicates as relation mentions O .

Concept Mention Grouping and Labeling Based on the findings of the concept grouping experiments in Section 6.2.3, we apply the greedy search optimization to find a good partitioning of concept mentions. It avoids the lumping problems of a simple transitive closure while still being efficient enough for our problem sizes. As for the experiments in Section 6.2.3, we first group mentions in linear time if their lemmas match, then make pairwise coreference classifications for the pre-grouped mentions and finally search for their best partitioning, resulting in unique concepts C . For each concept, we select one mention as its label. We experimentally found that using the most frequent mention, breaking ties by choosing the shortest, is a good heuristic to choose generic and representative labels.

The concept coreference classifier uses the similarities based on edit distance, Jaccard coefficient, Word2Vec, WordNet Resnik and LSA as features. For EDUC, we train it on the

17,500 pairs of mentions derived from the corpus creation process that we introduced earlier. For WIKI, we use a dataset of 4,500 pairs of mentions that we collected from Wikipedia page titles and their (linked) mentions on other pages. Note that both datasets are derived from only the training parts of the two corpora.

Relation Mention Grouping, Labeling and Selection In contrast to concepts, we handle relation mentions with a simpler approach as the number of relation mentions extracted for a pair of concepts is typically not too big. We group together mentions O to obtain R only if the mentions match exactly after lemmatizing their tokens. Among the multiple unique relations that might remain for a pair of concepts, we select the one extracted with the highest confidence by the OIE system. That ensures that the graph $G = (C, R)$ is a directed graph without multi-edges, as required by the task (see Section 3.3.7).

Importance Estimation For importance estimation, we train a Ranking SVM. Given the inconclusive findings of our importance estimation experiments with regard to modeling approaches (see Section 6.3.3), we expect that choosing other types of models would not have a major impact on the overall performance of the pipeline. We train the model on the same data as in those experiments, using concepts extracted on the training topics of EDUC or WIKI, the same set of features and the labels derived from comparing them against the reference maps. We tune the regularization parameter of the SVM by testing values in $\{0.1, 1, 10, 30, 100\}$ with leave-one-out cross-validation on the training topics. The final models are trained on the full training set. We do this separately for all ablations that produce different training data, i.e. different sets of concepts. The best parameter for lemma-based grouping on EDUC and all models on WIKI is 10, and 30 for CoreNLP-based grouping and our grouping model on EDUC (see next section for explanations of ablations).

Concept Map Construction For the final subtask, we use our proposed ILP (see Section 6.4.1) to select the optimal subgraph of G as the summary concept map.

6.5.2 Experimental Setup

For evaluation, we use the EDUC and WIKI corpora and evaluate all tested approaches with the metrics proposed for CM-MDS in Section 3.5.2, including both automatic metrics and manual analysis. We report results for the test sets of the two corpora. The training of all supervised models of the pipeline — the coreference classifier and the concept importance estimator — as well as the tuning of their hyper-parameters and the manual development of rule-based pipeline parts were performed on the training data only.

For the full task of CM-MDS, the only proposed method against which we can compare our pipeline is the baseline that we presented together with the EDUC corpus in Section 4.3.3.

To obtain more insights into the strengths of our pipeline and the effectiveness of the different proposed components, we additionally include a set of ablations of our pipeline in the experiments in which some components are replaced with alternative methods:

Lemma-Grouping Instead of using our pairwise classifier and partitioning, we group mentions only if their lemmatized versions match exactly.

CoreNLP-Grouping Instead of using our pairwise classifier and partitioning, we group mentions if they are part of the same coreference chain as determined by the coreference model of CoreNLP. Since CoreNLP only builds chains within documents, we merge them across documents with the lemma-matching strategy.

PageRank-Scoring Instead of the trained Ranking SVM, we use PageRank scores computed on the full graph G to score concepts.

CF-IDF-Scoring Instead of the trained Ranking SVM, we use the CF-IDF metric proposed by Zubrinic et al. (2015) to score concepts.

Frequency-Scoring Instead of the trained Ranking SVM, we use the frequency of a concept as its score as proposed by Valerio and Leake (2006).

Heuristic-Selection Instead of the subgraph selection ILP, we use the heuristic selection method inspired by Zubrinic et al. (2015) introduced in Section 6.4.2.

Apart from the described differences, all other steps of the ablations use exactly the same methods as used in the pipeline described before.

6.5.3 Results

Overall Results Tables 6.9 and 6.10 show METEOR and ROUGE scores for our pipeline and all baselines and ablations on both datasets. Compared to the corpus baseline, our model improves performance in terms of ROUGE on both datasets and according to METEOR on WIKI. These task-level results show that the newly proposed components for CM-MDS, which we already demonstrated to be effective in isolation in the previous sections, also form an effective pipeline for the full task of CM-MDS.

However, the lower METEOR scores for our pipeline on EDUC contradict with that general observation. We looked into these results in detail and found that the high scores of the baseline are due to heavy overgeneration during relation extraction, introducing many rather meaningless relations into the concept map. Since METEOR scores can be improved by incorrect relations if they are between a correct pair of concepts, leading to a partial match of the proposition, that undesired overgeneration behavior is rewarded by

EDUC	METEOR				ROUGE			
	Pr	Re	F1	p	Pr	Re	F1	p
Corpus Baseline	15.12	19.49	17.00	.1937	6.03	17.98	8.91	.1156
Improved Pipeline	15.14	17.34	16.12		9.37	11.93	10.38	
<i>Ablations</i>								
Grouping								
Lemma	13.93	15.42	14.57	.0023	8.21	8.59	8.25	.0017
CoreNLP	14.14	15.21	14.54	.0077	7.99	6.78	7.26	.0095
Scoring								
PageRank	11.78	16.21	13.61	.0005	7.14	11.66	8.66	.0052
Frequency	11.89	16.12	13.65	.0002	7.33	12.09	8.97	.0124
CF-IDF	12.48	16.44	14.15	.0002	7.68	12.08	9.25	.0235
Selection								
Heuristic	15.29	17.46	16.26	.6250	9.38	11.88	10.38	.9688

Table 6.9: End-to-end results on the EDUC test set for our pipeline and several baselines. P-values are computed with a permutation test that compares F1-scores against our pipeline.

the metric. Hence, the baseline only obtains higher scores by sacrificing the quality of the propositions, introducing many rather uninformative ones.

As suggested in Section 3.5.2, we carried out an additional human evaluation between the two systems to also assess aspects beyond the content-oriented automatic metrics, including differences in quality discussed above. Following our proposed evaluation protocol, the concept maps for each test topic generated by both approaches were shown to crowdworkers on Amazon Mechanical Turk who were asked for their preference. We collected five preferences for each of the 15 pairs of maps. Table 6.11 presents the results, showing that our concept maps tend to have more meaningful and topic-focused propositions and are especially more grammatical and less redundant than those generated by the baseline.

Concept Grouping To analyze the contribution of our concept grouping approach to the overall result, we compare our pipeline’s performance against the *Lemma* and *CoreNLP* ablations in Tables 6.9 and 6.10. Both alternatives cause a drop in both metrics on EDUC and WIKI, showing that our approach is important for the pipeline’s performance. The alternative methods are more conservative, merging much fewer concept mentions than necessary, but at the same time — in particular the *CoreNLP* variant — tend to lump too many different mentions together. In contrast, our model can make many more merges relying on differ-

WIKI	METEOR				ROUGE			
	Pr	Re	F1	p	Pr	Re	F1	p
Corpus Baseline	14.30	23.11	17.46	.5024	6.77	23.18	10.20	.2610
Improved Pipeline	19.57	18.98	19.18		17.00	10.69	12.91	
<i>Ablations</i>								
Grouping								
Lemma	18.32	17.24	17.59	.2050	13.99	9.53	11.07	.3270
CoreNLP	16.81	16.63	16.59	.1084	13.09	9.16	10.29	.1554
Scoring								
PageRank	13.27	14.13	13.62	.0062	8.35	6.17	7.01	.0097
Frequency	13.44	13.79	13.55	.0071	8.57	7.16	7.61	.0205
CF-IDF	14.63	14.92	14.72	.0189	10.50	7.91	8.87	.0450
Selection								
Heuristic	18.22	17.80	17.94	.3008	14.73	9.74	11.51	.3594

Table 6.10: End-to-end results on the WIKI test set for our pipeline and several baselines. P-values are computed with a permutation test that compares F1-scores against our pipeline.

ent types of semantic similarity and at the same time manages to avoid lumping effects by relying on the global partitioning approach.

Importance Estimation The contribution of our supervised scoring model based on Ranking SVMs can be seen in Tables 6.9 and 6.10 when comparing it to the unsupervised ablations *PageRank*, *Frequency* and *CF-IDF*. Note that all variants use the same concepts and relations as input and the same ILP-based subgraph selection. Our model clearly outperforms all alternatives. Looking into the learned weights for our set of features, we observed that the most helpful features are frequencies, in particular document frequency and CF-IDF, and topic relatedness as well as PageRank scores. The fact that these highest-weighted features coincide with the metrics used by the ablations, which we chose based on previous work, shows that we indeed compare our model against the most competitive unsupervised alternatives. To identify unimportant concepts, i.e. assigning low scores, the model makes use of concreteness values and the label’s length.

Concept Map Construction To analyze the effectiveness of our proposed ILP subgraph selection approach, we compare the pipeline against the *Heuristic* ablation. That comparison is essentially equivalent to the experiment carried out in Section 6.4.2 with the only

Dimension	Corpus Baseline	Improved Pipeline
Meaning	0.44	0.56
Grammaticality	0.31	0.69
Focus	0.44	0.56
Non-Redundancy	0.21	0.79

Table 6.11: Human preference judgments between concept maps generated on EDUC.

difference that predicted importance scores are used instead of gold scores. However, in this end-to-end evaluation, the effectiveness of the ILP approach is less pronounced: While scores on WIKI are higher for our approach, the difference on EDUC is only marginal and even slightly prefers the heuristic. Since the ILP is guaranteed to find an optimal solution for the optimization problem, these results seem to be counter-intuitive.

The reason for this observation are errors in the preceding importance estimation step: The optimal subgraph according to the estimated scores is in this case not the best with regard to the reference summary concept maps, explaining the slightly higher METEOR scores on EDUC for the heuristic selection. Despite that behavior on EDUC, the ILP is still the superior approach. Looking into the detailed optimization results, we observed that the heuristic found the optimal subgraph for only 35% of the test topics, selecting a subgraph with an on average 0.63% (EDUC) and 1.30% (WIKI) lower objective function score in the other cases. To ensure that the optimal subgraphs selected by the ILP also lead to better results in the final evaluation, one would need to improve the importance scoring model.

6.5.4 Error Analysis

In order to illustrate the type of errors our pipeline makes, Figure 6.2 shows a part of the summary concept map created for the document cluster on “*students loans without credit history*”. Figure 4.5 shows a similar part of the corresponding reference concept map. The color-coding reveals that the automatically created map includes many of the concepts present in the reference map, such as *student*, *credit check* and *federal student loans*. Even more parts of the created map overlap with the reference map in terms of content, but are expressed with different labels. Examples are slightly different labels, such as *good credit history* instead of *sufficient credit history*, or different propositions, such as expressing the need for a cosigner via the *parents* concept, while the reference map has a direct proposition for it. Since such alternative representations are difficult to reward with automatic evaluation metrics, the examples underline that manual evaluations as proposed in Section 3.5.2 and performed in the previous section are also necessary.

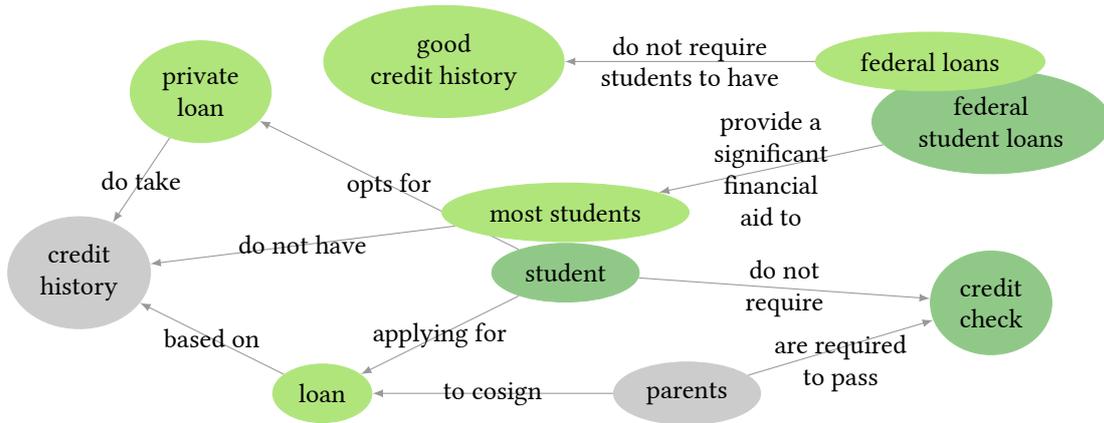


Figure 6.2: Excerpt from the summary concept map created with the improved pipeline for the topic “students loans without credit history”. The corresponding reference map is shown in Figure 4.5, but both figures show only excerpts. We manually selected a part with high overlap but also containing representative errors. The full summary has 25 concepts and 47 relations. Concepts in dark green match the reference exactly, the ones in light green use alternative labels.

With regard to errors, the example illustrates three important categories: First, the extraction step of the pipeline sometimes yields propositions that are not very meaningful. For instance, in (*private loan - do take - credit history*), the relationship has a label that does not make much sense for the two concepts. Second, while we showed earlier that our method groups substantially more concept mentions than previous work, there are still several mentions left that should have been grouped, such as *federal loans* and *federal student loans*. However, as we discussed earlier, finding the right concept granularity is challenging. The third, less frequent but severe error category is the presence of factually wrong propositions, such as (*student - do not require - credit check*). For only a subset of loans, in particular federal loans, credit checks are not required, but in this general manner, the proposition is not correct. Incorrect propositions can be due to extraction errors, too aggressive concept mention grouping and combinations of both.

In order to better quantify the relevance of different types of errors happening in our pipeline, Table 6.12 shows the number of concepts and their recall at different steps. The recall of concept mentions shows that performance is already lost during extraction, suggesting that better approaches would be beneficial. We observed that the problem is mainly the identification of correct spans rather than missing some concepts completely. A custom extraction model instead of relying on existing OIE systems could resolve this issue, but would require training data with annotated mention spans.

With regard to concept grouping, we found that even more coreferent mentions should be grouped together than done by our current model. However, while the current model only rarely incorrectly merged mentions of different gold concepts, in particular in only two cases across all test topics, a more aggressive grouping could introduce more of these

Subtask	EDUC		WIKI	
	Count	Recall	Count	Recall
Mention Extraction	8630	73.87	2549	88.93
Mention Grouping	4022	60.27	1315	82.38
Subgraph Selection	25	16.53	11	30.71

Table 6.12: Average number of concepts and recall per topic at different pipeline steps.

errors. Please also note that the drop in recall in Table 6.12 is mainly due to the exact match metric used here to compute recall, which misses concepts for which the selected label is not exactly the one used by the corresponding concept in the reference concept map. Of course, the label selected instead could still be a good one and our main evaluation metrics based on METEOR and ROUGE can better account for that.

Finally, Table 6.12 again reveals that the main bottleneck is to determine the important concepts. On both datasets, but especially on the bigger document sets of EDUC, a substantial amount of recall is lost during this challenging step. We already made similar observations when evaluating the corpus baseline in Section 4.3.3 and importance scoring models in isolation in Section 6.3.3. For future improvements of the pipeline, the biggest improvements seem to be possible in this subtask.

6.5.5 Runtime Complexity and Optimizations

At various points in this thesis we pointed out the challenge and importance of scalability. In Section 6.2, we compared runtimes of semantic similarity measures, introduced a hashing-based pre-grouping to reduce problem sizes and developed efficient partitioning algorithms as alternatives to expensive ILP formulations. In Section 6.4, we introduced more efficient ILP formulations for subset selection and further improved runtimes by breaking the selection problem down to connected components.

While all of these measures are necessary to enable the application of our proposed methods to problem sizes as present in the EDUC corpus, the scalability of the pipeline as described in this chapter is still limited. A bottleneck is concept mention grouping. As we showed in Section 6.2.3, finding a good partitioning already takes more than 2.5 hours on the largest document set. Computing the features for all pairs of concept mentions takes even longer. Due to the inherent quadratic runtime behavior of pairwise comparisons, the computation time for this step quickly explodes with larger inputs. For sets of several thousands or more documents, the application of the pipeline already becomes infeasible. In this section, we therefore describe a few possible directions to improve the runtime behavior of that step and report preliminary experiments.

As an alternative to our greedy partitioning search, an efficient graph clustering algorithm proposed by Biemann (2006), called Chinese whispers, can be used. It initially assigns every node in a graph to its own cluster and then repeatedly iterates over all of them and changes the cluster assignment of a node to the majority cluster of its neighbors. While not formally guaranteed, this process typically converges to a stable clustering. When run for k iterations, it takes $\mathcal{O}(k|M|^2)$ time, with even better average case runtime for sparse graphs. To our mention grouping problem it can be applied by setting a cut-off threshold for coreference probabilities and then running Chinese whispers on the graph with nodes for all mentions and edges for pairs with probabilities above the threshold. In preliminary experiments, we saw runtimes similar to greedy search on our smallest topic but much faster on the big ones while finding partitionings scoring equally well. However, since this algorithm does not actively optimize our objective function, obtaining good partitionings amounts to carefully tuning the cut-off threshold. And more importantly, it still requires the computation of all pairwise features and predictions before it can be applied, and thus does not fully solve the scalability problem.

To fully overcome the need of making all pairwise comparisons of mentions, one possibility is to rely on locality sensitive hashing (LSH). Given points in a vector space, the central idea of LSH is that one can compute binary hash representations of such vectors which approximately preserve the distance between the vectors. More specifically, the hamming distance between binary hashes for two points can be used to approximate their cosine similarity in the vector space (Charikar, 2002). If we represent concept mentions by vectors in a vector space, e.g. using word embeddings, instead of computing pairwise features, the idea of LSH can be applied to our problem in at least two different ways.

The first option is to rely on the fact that the shorter the binary hashes computed for LSH are, the more vectors get mapped to the same hash, automatically clustering them into groups. As we explained before, the similarity of hashes directly corresponds to the cosine similarity between vectors, and the vectors that get mapped to the exact same hash are therefore the most similar ones. The mention grouping approach is therefore simple: We represent every mention by its word embedding, compute hashes for the embeddings and group those mentions together that receive the same hash. The whole process can be performed in $\mathcal{O}(|M|)$ time. However, despite being conceptually appealing, this approach is difficult to use in practice. The length of the computed hashes has to be carefully tuned to control the size of the created groups and has to be adapted when the number of mentions becomes substantially smaller or larger.

As a second option, one can use LSH together with fast hamming search (Charikar, 2002), a probabilistic algorithm that can find the objects closest to a given object by comparing their hashes (computed from their vectors) to only a subset of all other objects. It has been applied by Ravichandran et al. (2005) to the NLP problem of efficiently finding, for a given noun, the most similar other nouns in a large list. Applied to our problem, we can use

it to find for each concept mention all other mentions that have a cosine similarity above a certain threshold in $\mathcal{O}(|M|\log|M|)$ time. While this approach is therefore an efficient alternative to pairwise mention classification, it still leaves the partitioning problem to be solved. Since the optimization problem to find globally consistent partitionings defined in Section 6.2.2 relies on pairwise scores for all pairs, especially also low ones, it cannot be directly applied in this case. Nevertheless, developing partitioning approaches for this case seems to be a worthwhile direction for future work on improving scalability.

6.6 Chapter Summary

In this chapter, we focused on the CM-MDS subtasks for concept mention grouping, importance estimation and concept map construction. Our contributions include improved techniques for these subtasks and a new state-of-the-art pipeline for the task as a whole.

For concept mention grouping, we proposed a solution based on pairwise mention classification and a subsequent partitioning step. Compared to previous work on concept map mining, this approach can capture more types of coreferences as it relies on a variety of different semantic similarity metrics. Although its inherent quadratic runtime behavior makes the approach challenging to apply to large corpora, we carefully analyzed several trade-offs between computation time and grouping performance to design an approach that can be successfully applied to the problem sizes in our evaluation corpora.

With regard to importance estimation, we studied a broad set of more and less commonly used features as well as different machine learning approaches to determine the importance of concepts. While we could not observe clear advantages of modeling the problem as regression, classification or ranking, we did design supervised models that clearly outperform the exclusively unsupervised methods suggested in previous work on concept maps, therefore contributing to an overall improved performance on CM-MDS.

For concept map construction, we proposed an ILP formulation that allows us to find an optimal solution to the subgraph selection problem of CM-MDS. We experimentally demonstrated that these optimal subgraphs are superior to heuristically selected ones on our evaluation corpus. In addition, we designed the subgraph selection process to be particularly efficient such that it can scale to problem instances of a reasonable size.

Based on the proposed methods for the three subtasks and the mention extraction methods explored in the previous chapter, we then presented a full pipeline for CM-MDS. The pipeline was shown to be superior to a range of baselines based on previous work in both automatic and manual evaluations on two corpora, establishing a new state-of-the-art for the task of CM-MDS. We finally pointed out directions for further improvements of the pipeline, including in particular the need for better importance estimation models, and sketched possibilities to improve its scalability.

End-to-End Modeling Approaches

In this chapter, we focus on alternative models for CM-MDS that try to approach the task end-to-end rather than with a pipeline of multiple steps. For various tasks in NLP, such approaches have recently been very successful. We first discuss how sequence transduction models can be applied to CM-MDS. Then, we propose a new alternative architecture which we name a sequence-to-graph network. We evaluate both approaches experimentally to assess the applicability of end-to-end modeling for CM-MDS.

7.1 Motivation and Challenges

In recent years, the use of *neural network* models — mostly under the name *deep learning* — has led to substantial performance improvements in many NLP tasks. Prominent examples are the pioneering work by Collobert et al. (2011) and subsequent work on language modeling (Mikolov, 2012), text classification (Socher et al., 2013, Kim, 2014) and machine translation (Sutskever et al., 2014, Cho et al., 2014). A lot more work than we can reference here has been done in that area, with large fractions of the papers published in NLP venues in the last three to four years focusing on the development and analysis of neural models. Goldberg (2017) provides an overview of these efforts, whereas Goodfellow et al. (2016), LeCun et al. (2015) and Schmidhuber (2015) give a more general overview of types and applications of deep learning across different fields.

That large body of work on neural models has repeatedly confirmed several advantages that neural networks have over traditional, feature-based machine learning approaches. First, they make it possible to model many complex tasks, e.g. machine translation, as a single end-to-end problem. That allows learning powerful task-specific representations, such as word vectors, directly from the task-level supervision signal and in addition avoids error propagation problems that typically occur in multi-step pipelines where individual models are not trained jointly. Second, distributed word representations and architectures

such as CNNs or RNNs that can combine word vectors into higher-level representations have made it possible to learn models directly from raw text. The design and selection of features, as done in traditional machine learning approaches, is no longer necessary. And third, as a consequence of the previous, many tasks in NLP can nowadays be approached with the same kind of model and no task-specific knowledge or techniques are needed to reach competitive performance.

In light of these advantages, we are interested in also applying such models to CM-MDS, where we expect several benefits. First, the end-to-end modeling and training can overcome the error propagation problems of the pipeline approach observed in Section 6.5. Second, task-specific word representations learned in this end-to-end fashion can improve performance on the difficult concept coreference subtask, where we already found generically pre-trained embeddings to be helpful in Section 6.2.3. And third, parts of our pipeline rely on off-the-shelf software such as OIE systems which have not been explicitly trained to extract concepts and relations. With an end-to-end model, we would be able to directly learn extraction approaches as needed for CM-MDS.

However, this direction also poses the following new challenges:

Large Inputs In CM-MDS, and in particular on our EDUC corpus, the size of the input is large, consisting of multiple full documents with a total of about 100,000 tokens. Most other NLP tasks to which neural models have been successfully applied work with only single documents, sentences or individual words as inputs. For the common sequence processing architecture, RNNs, long sequences can be challenging in terms of training time, memory requirements and successful backpropagation of the training signal.

Graph Outputs The output in CM-MDS is a labeled, directed graph. Thus, it is a structured prediction rather than a simpler classification or regression task. While much work exists on the most prominent structured prediction task in NLP, the prediction of word sequences as in machine translation or text generation, to the best of our knowledge, no neural network components have been proposed to directly predict a labeled, directed graph.⁶⁹

Little Data Due to the structure of the output and the size of the input, manually creating reference concept maps is expensive as we showed in Chapter 4. Only with a complex pipeline of preprocessing, crowdsourcing and manual annotations, we could create the high-quality EDUC benchmark corpus of 30 document-summary pairs. While that size is useful to evaluate CM-MDS approaches, it is far less than what is needed to train neural models. The work on neural SDS, for instance, relies on hundreds of thousands of pairs.

⁶⁹Note that tasks such as dependency parsing or semantic parsing to AMR also produce graphs (or trees), but in a simpler fashion: Exactly every input token is a graph node and only edges need to be determined.

In the following sections, we discuss several solutions to these issues, including methods to generate synthetic training data, pre-summarizing inputs, linearizing target graphs and using our novel sequence-to-graph architecture. We study the effectiveness of these ideas in a range of experiments.

7.2 Synthetic Training Corpora

To successfully train neural networks to perform complex structured prediction tasks, a large number of training examples is necessary. For neural SDS, existing work relies on the 460k pairs of news articles and summaries in the Annotated New York Times corpus (Sandhaus, 2008), the 290k article-summary pairs of the CNN/DailyMail corpus (Hermann et al., 2015) or the 10 million article-headline pairs of the Gigaword corpus (Napoles et al., 2012). Very recently, an additional corpus with 1.3 million pairs of news articles and summaries from various sources has been published by Grusky et al. (2018). As we already mentioned in Section 2.3.2, no datasets of these sizes exist for MDS. Similarly, for our CM-MDS task, we also lack suitable training data for neural models, as EDUC and WIKI have only 30 and 38 and BIOLOGY only 183 examples. Note that while the corpus creation method presented in Section 4.3.2 has been specifically designed to make the annotation scale to the size of the benchmark corpus, the number of pairs required for neural models are far beyond its scalability: With on average \$150 crowdsourcing cost per document set, creating a dataset of 100k examples with the approach would cost at least 15 million dollars.

To be able to carry out experiments with neural models despite the lack of training data, we pursue two directions to obtain synthetic training examples automatically:

Extending CM-MDS Corpora The idea of our first data generation approach is to use the concept maps of the training part of EDUC and pair them with alternative inputs to obtain additional pairs. As we discussed during mention extraction (see Chapter 5), there are many different ways to express the same concepts and relations and multiple alternative inputs can therefore lead to the same summary concept map.

Specifically, we process each of the 15 training topics and create additional examples. Given the summary concept map with concepts C and relations R , we first collect alternative expressions for them in addition to their labels. Towards that end, we use data collected during the corpus creation (see Section 4.3.2) in which alternative mentions of concepts have been merged and further rely on PPDB 2.0 (Pavlick et al., 2015), a database of paraphrases. Looking up our concepts and relations in it, we can find alternatives such as *manifestations* for *symptoms* or *is not necessary* for *is not required*. As a result of this step, we have concepts C and relations R with a set of alternative referring expressions for each of them. In the second step, we then build all possible triples (c_1, r, c_2) out of C and R , sample a triple of expressions (m_{c_1}, o_r, m_{c_2}) from their corresponding expression sets and

Dataset	Pairs	Concept Map				Source	
		Concepts	Tokens	Relations	Tokens	Tokens	
EDUC	30	25.0 ± 0.0	3.2 ± 0.5	25.2 ± 1.3	3.2 ± 0.5	97880.0 ± 50086.2	
EDUC _{SYN}	49,950	21.2 ± 2.3	2.8 ± 2.0	20.6 ± 2.3	2.1 ± 1.5	$623.3 \pm$	83.6
DMS _{SYN}	209,525	25.0 ± 0.2	2.9 ± 2.0	20.6 ± 0.9	3.3 ± 2.8	$302.1 \pm$	35.3

Table 7.1: Comparison of EDUC with the two synthetic training datasets.

search in a recent English Wikipedia dump for sentences mentioning all three parts. Given such triples and retrieved sentences with corresponding mentions, we then build concept maps in the final step. We sample out of all triples for which sentences were found until the sampled subset forms a graph of more than 20 nodes and concatenate the corresponding sentences to form the source text. That process yields a single new pair of input text and output graph, which we repeat until we run out of triples.

With this approach, we obtained the EDUC_{SYN} dataset of roughly 50k new pairs. All of these pairs have concept maps similar to the original 15 maps from EDUC, but use different mentions, different subsets of propositions and different sentences on the input side.

Pipeline-based Corpus Creation As the second strategy, we use a corpus of plain documents and apply existing CM-MDS machinery – the pipeline introduced in Section 6.5 – to automatically create corresponding concept maps. While this obviously yields noisy data as the pipeline cannot handle CM-MDS perfectly, similar approaches of using large noisy datasets in combination with smaller high-quality datasets have previously been successfully used for other tasks to bootstrap models, e.g. Konstas et al. (2017) for AMR parsing.

We use the DailyMail part of the corpus by Hermann et al. (2015) and process the news articles with our pipeline to extract triples of concepts and relations. We do not use the summaries the corpus provides. We then randomly sample subsets of these triples and combine them to graphs of 25 concepts. These graphs are paired with the sentences the triples were extracted from. Note that this process does not make use of the importance scoring and subgraph selection steps of the pipeline, since all concepts and relations are used to create new pairs. As a result, we obtained DMS_{SYN}, a dataset of almost 210k pairs of input texts and corresponding concept maps.

Table 7.1 compares the two synthetic datasets with EDUC. Note that they are orders of magnitude larger, with sizes comparable to the aforementioned corpora used to train neural SDS models. The length of the input, on the other hand, is much smaller for the synthetic datasets, which is part of our approach to handle the large input problem. More details on that will follow. The main difference between the two synthetic datasets is that DMS_{SYN}

is noisier and has a larger textual variety, with over 100k unique words on the input and output side, while EDUCSYN is cleaner, has a 65k input vocabulary and a small 1k output vocabulary that is very similar to the training part of EDUC.

7.3 Sequence Transduction Models

Sequence transduction models, also commonly referred to as *sequence-to-sequence models* or the *encoder-decoder architecture*, are neural networks with an encoder that processes an input sequence token by token and a decoder that generates a corresponding output sequence token by token (Goldberg, 2017). Such models have been first proposed as end-to-end models for machine translation by Sutskever et al. (2014) and Cho et al. (2014). Common architectures for the encoder and decoder component are long-short term memory (LSTM) cells (Hochreiter and Schmidhuber, 1997) and gated recurrent unit (GRU) cells (Cho et al., 2014). An important extension of this framework that has been widely adopted is the idea of an attention mechanism (Bahdanau et al., 2015, Luong et al., 2015) that allows the decoder to access specific parts of the input when generating tokens of the output. As we already pointed out in Section 2.3.2, sequence transduction models using RNNs and attention are also the architecture mainly used by current neural abstractive SDS models.

Having created the two large synthetic datasets, we face two remaining challenges when trying to apply sequence transduction models to CM-MDS: While these models learn to map sequences to sequences, our outputs are labeled graphs instead of sequences, and second, our inputs are typically much longer than the sequences that can be handled by RNNs. We present two simple workarounds in the next sections that reduce CM-MDS to a regular sequence transduction problem and thus allow the application of such models.

7.3.1 Graph Linearization

Previous work already demonstrated that sequence transduction models can be used for problems where the output is not a sequence, but can be converted to one, e.g. for parse trees (Vinyals et al., 2015b), AMR graphs (Konstas et al., 2017) or logical forms (Dong and Lapata, 2016). We follow this idea and let the model learn to map the input sequences to linearized summary concept maps.

We explore two different linearization schemes. The **triples** linearization encodes the graph as a list of concept-relation-concept triples separated by special symbols for each part. The summary concept map for the example in Figure 3.1 would be encoded as:

\$S caffeine \$R reduces \$T ADHD symptoms \$S hypnotherapy \$R has little to no effect on \$T ADHD symptoms \$S herbal supplements \$R have been used to treat \$T ADHD symptoms

As an alternative, the **concepts** linearization lists all concepts first and then defines relations providing their labels and references to the source and target concept’s position in the concept list. The example is encoded as follows:

```
$C caffeine $C ADHD symptoms $C hypnotherapy $C herbal supplements $R $0 $1
reduces $R $2 $1 has little to no effect on $R $3 $1 have been used to treat
```

Both formats have their challenges: While the latter requires the model to learn to use the $\$n$ references according to the order of the concepts, the first format is redundant and requires the exact label of a concept to be mentioned for each of the triples it participates in. To make the graph linearization deterministic, we rely on graph-input alignments and encode elements in the order in which they first occur in the input sequence.

7.3.2 Pre-Summarization

To face the second challenge, we propose a two-stage approach of first pre-summarizing a multi-document input to a shorter sequence and then feeding that sequence into a sequence transduction model. While the first step is handled by a traditional non-neural MDS model, which can easily scale to large document collections, a powerful neural model for the second step can then learn to create graph-structured summaries and can still make the final decision on which parts of its input sequence to include in the summary. A similar two-stage summarization process to handle large inputs has been independently explored in recent work by Liu et al. (2018) on neural MDS.

We experimented with several sentence-based extractive summarization methods and found an embedding-based variant of LexRank (Erkan and Radev, 2004) to perform best in our setup. As in LexRank, we build a graph with a node for each sentence and edges connecting sentences that have a similarity above a threshold ρ . As similarities, we use the cosine similarity between averaged word embeddings. Running the PageRank algorithm on this graph yields a score p_i for each sentence. Since our corpora also have a topic description for each document set, we further compute its similarity q_i with each sentence. Sentences are then scored by a linear combination of both measures

$$s_i = \lambda p_i + (1 - \lambda) r_i \quad (7.1)$$

where λ controls the influence of both factors. To obtain the input to the neural model, we sort all sentences according to their score and use the first t tokens following this order as the inputs for the neural model. We explore appropriate values for the parameters ρ , λ and t in the following experiments.

Note that this setup is also the reason for the small inputs in the created EDUCSYN and DMSYN corpora: While EDUC represents the full multi-document summarization task, the synthetic data only reflects the second part of our two-stage approach. After a neural

model for the second step has been trained on the synthetic data, we can then apply pre-summarization to the document sets of EDUC to obtain sequences of a similar length — a few hundred tokens — than can be processed by the trained neural model.

7.3.3 Experiments

We present first experimental results that assess how promising our proposed reduction of CM-MDS to a sequence transduction problem is.

Experimental Setup We use a sequence transduction model with a bidirectional encoder, unidirectional decoder and multiplicative attention (Luong et al., 2015). The encoder uses two 150-dimensional GRU cells (Cho et al., 2014) for forward and backward processing while the decoder has a single 300-dimensional cell. The network is implemented in TensorFlow (Abadi et al., 2015) and trained on a Tesla K40c. All experiments use the same vocabulary of 50k unique tokens and their embeddings are initialized with pre-trained 300-dimensional GloVe vectors (Pennington et al., 2014).⁷⁰ We train with Adam (Kingma and Ba, 2015), minimizing cross-entropy per predicted token, and use dropout of 0.2 after the embedding and encoder layers (Srivastava et al., 2014). All models are trained on either DMSYN or EDUCSYN until the loss on a held-out 5% VAL subset stops improving.

After training on the synthetic data, we evaluate the obtained models on EDUC with pre-summarization. We tuned ρ and λ of the pre-summarizer optimizing ROUGE-2 recall of the resulting summary sequences on the training part of EDUC and found 0.5 and 0.6 to be best. The length t of the input for the neural model is tuned on EDUC-Train separately for each model. For predictions, we use beam search with a beam size of 10 and turn the obtained sequences back into graphs. Since the task requires a concept map to be a connected graph, we reduce the graphs to their biggest component if they are disconnected. If they are still bigger than the summary size limit, which is 25 for EDUC, we further remove nodes, starting with the lowest-degree nodes. We report results according to the CM-MDS ROUGE metric (see Section 3.5.2) for the held-out synthetic data as well as training and test sets of EDUC.

Effect of Linearization With regard to different linearization schemes, we found *triples* to be superior to *concepts*, as models using the latter largely failed to produce correct relations in their predicted graphs. However, all models learn the syntax of both linearizations very well, and even on the test set of EDUC, they made none (*triples*) or only one (*concepts*) syntax errors. A bigger problem are the structure and the size of the produced graphs: On the test set of EDUC, predicted graphs have on average only 14.5 concepts and are often disconnected, leaving them with 10.7 after removing unconnected components. The gold data instead has 25 concepts per graph.

⁷⁰Available at <https://nlp.stanford.edu/projects/glove/>.

Approach		SYN-Val			EDUC-Train			EDUC-Test		
Data	Linearization	Pr	Re	F1	Pr	Re	F1	Pr	Re	F1
DMSYN	Triples	40.4	8.6	13.7	14.8	2.0	3.2	13.5	2.2	3.6
EDUCSYN	Triples	75.9	50.9	59.2	36.5	12.4	18.3	2.6	0.8	1.2
EDUCSYN	Concepts	54.4	53.4	53.5	26.5	16.8	20.5	1.9	1.0	1.3

Table 7.2: Sequence transduction performance for different synthetic training data and linearizations. Reported are ROUGE scores on held-out synthetic data and train and test sets of EDUC.

Effect of Pre-Summarization The tuned pre-summarization model yields sequences with an average ROUGE-2 recall around 20 on the train and test sets of EDUC. This recall limits the achievable performance of the full two-stage approach, as illustrated by the drop of performance that can be seen in Table 7.2 between SYN and EDUC-Train. Note however that this does not make the approach completely infeasible, as the pipeline-based approach also has an end-to-end recall of only 12 ROUGE recall points (see Table 6.9). When tuning the length of the pre-summarized sequence, we found that the neural models tend to perform best with pre-summarized inputs of 700 to 1000 tokens. This is interesting since the synthetic datasets used for training contain on average shorter sequences.

Effect of Training Data Table 7.2 also shows how useful the two generated synthetic datasets are. As the scores on the held-out VAL subsets show, the sequence transduction models are able to learn the task as represented by EDUCSYN to a fair amount. The noisier DMSYN data is more difficult to fit. While that difference remains when applying the models to EDUC-Train, the picture is very different on the unseen evaluation data (EDUC-Test), and we discuss the reason for that in the next section.

Vocabularies and Topic Shift The experiments revealed a big challenge for neural models which we refer to as the *topic shift*. Since the document sets of EDUC cover different topics, only a third of the vocabulary of the concept maps in the training part overlaps with the maps for the test topics. This is a huge problem for a token-level sequence transduction model. If it is trained on the training part of EDUC, it will have to generate tokens at test time it has never seen during training, rendering prediction on the test set to a partial zero-shot learning problem. Consequently, the performance of models trained on EDUCSYN, which is very similar to EDUC-Train, is particularly poor on EDUC-Test. In contrast, the model trained on DMSYN performs equally well on the train and test sets of EDUC.

Given these results, we test a simple method to allow the trained model to better transfer to the test data: We use embeddings E for a large vocabulary, initialized with pre-trained word embeddings, and use the same E for encoder inputs, decoder inputs and the decoder’s

Approach		SYN-Val			EDUC-Train			EDUC-Test		
Data	Embeddings	Pr	Re	F1	Pr	Re	F1	Pr	Re	F1
EDUCSYN	Tuned	75.9	50.9	59.2	36.5	12.4	18.3	2.6	0.8	1.2
EDUCSYN	Trans	66.6	43.9	51.7	33.1	11.8	16.7	3.4	1.1	1.6
EDUCSYN	Fixed	66.9	39.1	47.9	42.0	19.3	26.1	2.7	0.8	1.2

Table 7.3: Sequence transduction performance for different embedding learning strategies. Reported are ROUGE scores on held-out synthetic data and train and test sets of EDUC.

output projection (shown by Paulus et al. (2018) to be generally beneficial). We compare variants of our model that fine-tune E during training (*Tuned*), keep E fixed (*Fixed*) or use a transformation $E \cdot T$ with E fixed and T learned (*Trans*). Note that *Tuned* can fit the training data better, but it will change the embeddings of words in the training set and leave the others unchanged, making the generalization to them at test time difficult. *Fixed* avoids that, but does not allow learning task-specific embeddings at all. The *Trans* approach tries to trade off between these two alternatives.

As the results in Table 7.3 show, the use of pre-trained embeddings is not enough to cope with the topic shift. Both the *Trans* and *Fixed* setups have only a minor effect and do not solve the problem.⁷¹ Training on the larger DMSYN data seems to make the model more robust, as it has seen a broader range of outputs. However, none of the explored setups can currently compete with the pipeline models introduced earlier, which do not face such problems since none of their components use lexical features.

Conclusion We proposed several ideas that allow us to apply sequence transduction models to CM-MDS. Our experiments show that while these models can in principle be used to generate concept maps, they are not yet competitive with non-neural approaches. A big problem for the neural models is the topic shift between the train and test parts of EDUC due to the model’s word-level generation approach and the resulting dependency on overlapping vocabularies. To make the neural approaches more competitive, it seems to be necessary to have a training dataset with high-quality text–concept map pairs that span an output space large enough to allow the model to generalize to the test data.

As a second challenge, we identified the difficulty of predicting graph-structured outputs. While the models learned our linearization syntax well, the predicted graphs were often disconnected and smaller than the gold maps. More sophisticated approaches to suc-

⁷¹We also carried out experiments with pointer-generator mechanisms for sequence transduction models as proposed by See et al. (2017). In addition to generating words from a fixed vocabulary, they have a decoder that can also copy words from the input, in particular the input words not present in the vocabulary. We observed no noticeable difference in performance compared to a plain sequence transduction model.

cessfully predict graphs with a neural model that satisfy structural constraints such as connectedness are needed for CM-MDS.

7.4 Memory-based Graph Manipulation Models

In the previous section, we approached the challenge of creating graph-structured outputs by linearizing them to sequences. While being simple and reducing the problem to well-studied sequence-to-sequence transformations, this is far from ideal. If we want a model to understand the structure of the output, for example to satisfy constraints such as connectedness, it has to fully derive all that knowledge — what a graph is, how nodes and edges are related to each other and where that information is stored in the linearization — in addition to learning the actual task. Empirically, we also saw that the trained models have trouble producing connected graphs of the desired size. A decoder architecture specifically designed to produce graphs as outputs could relieve the learned model from understanding linearization strategies and thereby improve the performance on CM-MDS.

Towards that end, we propose a novel sequence-to-graph architecture in this section. Such models can learn to map from text sequences directly to labeled, directed graphs and therefore make linearizations obsolete. To the best of our knowledge, this is the first proposed architecture for sequence-to-graph transformations.

The sequence-to-graph architecture builds upon and combines ideas from several recent directions in deep learning research, namely

- encoder-decoder architectures (Graves, 2013, Kalchbrenner and Blunsom, 2013, Cho et al., 2014, Sutskever et al., 2014, Bahdanau et al., 2015),
- matrix-based neural graph representations (Scarselli et al., 2009, Li et al., 2016b, Johnson, 2017)
- memory-augmented neural networks (Graves et al., 2014, 2016, Gulcehre et al., 2017, 2018, Weston et al., 2015b, Sukhbaatar et al., 2015)

More specifically, we propose an encoder-decoder architecture with a structured memory as the internal representation. The encoder, similar to a neural Turing machines (NTM) (Graves et al., 2014), recurrently updates memory matrices which we use to represent a graph in a similar manner as the work of Li et al. (2016b) and Scarselli et al. (2009). Once the encoder produced a final memory representation of the input, a novel decoder then creates a graph from that representation. Similar as in sequence transduction models, text is produced word by word, however, our decoder produces a separate sequence for each node and edge of the graph as well as information on how they are connected.

Most similar to our model is the work by Johnson (2017). Building upon Li et al. (2016b), who proposed a representation and a set of recursive operations to learn distributed representation of given graphs and their elements, Johnson (2017) combines that idea with

NTM-style memory updates to build the graph representation step by step based on an input sequence. While that is conceptually very similar to the encoder of our proposed architecture, we use different (and simpler) memory representations and update operations to encode graphs. Further, in their work, Johnson (2017) uses the model for question answering on bAbI (Weston et al., 2015a), producing a single-word answer from the intermediate graph representation rather than decoding the graph itself.⁷² A decoder that can create a labeled, directed graph as the model’s output is only part of our architecture.

7.4.1 Memory-based Graph Representations

Figure 7.1 illustrates the memory of our model, consisting of four different memory matrices. A first memory matrix, $\mathbf{M}^N \in \mathbb{R}^{(n+1) \times m}$, is used to store the nodes of the graph, where each row $\mathbf{M}_{[i]}^N$, called a memory cell, is an m -dimensional dense representation of a node. During decoding, the node’s label is generated from it. Similarly, the second matrix $\mathbf{M}^E \in \mathbb{R}^{(e+1) \times m}$ stores m -dimensional representations of the edges. While each cell of it is used to generate an edge’s label during decoding, additional matrices \mathbf{M}^S and $\mathbf{M}^T \in \mathbb{R}^{(e+1) \times (n+1)}$ store distributions over $n+1$ nodes indicating the source and the target of the corresponding edges.

Similar to a recurrent network, our model processes the input sequence one token at a time and can update the memory matrices at every timestep. At the end of the encoding phase, the final graph is represented by the memory and can be decoded. For the CM-MDS task, the memory would be used to gradually build the summary concept map. Whenever a mention of a concept or relation is encountered while processing the input tokens, nodes or edges can be created using empty memory cells. If a repeated mention of a concept is encountered, rather than creating a new node, the model can update the initially used cell in the memory, e.g. to refine or strengthen that node. Figure 7.1 illustrates this conceptual use of the memory and how its content corresponds to the task’s input and output.

Note that the number of cells in the memory is fixed to n nodes and e edges. Thereby, we force the model to create only graphs equal to or smaller than the size limits \mathcal{L}_C and \mathcal{L}_R defined for CM-MDS. In that way, the architecture itself introduces the summarization aspect of the task into the model and forces it to learn to keep only important elements in the limited memory while processing the text. If all cells have been used, the model has to override the cell containing the information deemed to be least important. Kintsch and van Dijk (1978)’s text understanding model characterizes human text understanding in a similar iterative and capacity-limited manner and was previously successfully used to build non-neural summarization systems (Fang et al., 2016).

⁷²Since the internal graph representation of Johnson (2017)’s model is only used to predict answers, but never explicitly decoded, it is in fact unknown whether the parts of the memory that are supposed to represent nodes and edges actually represent such things in any way.

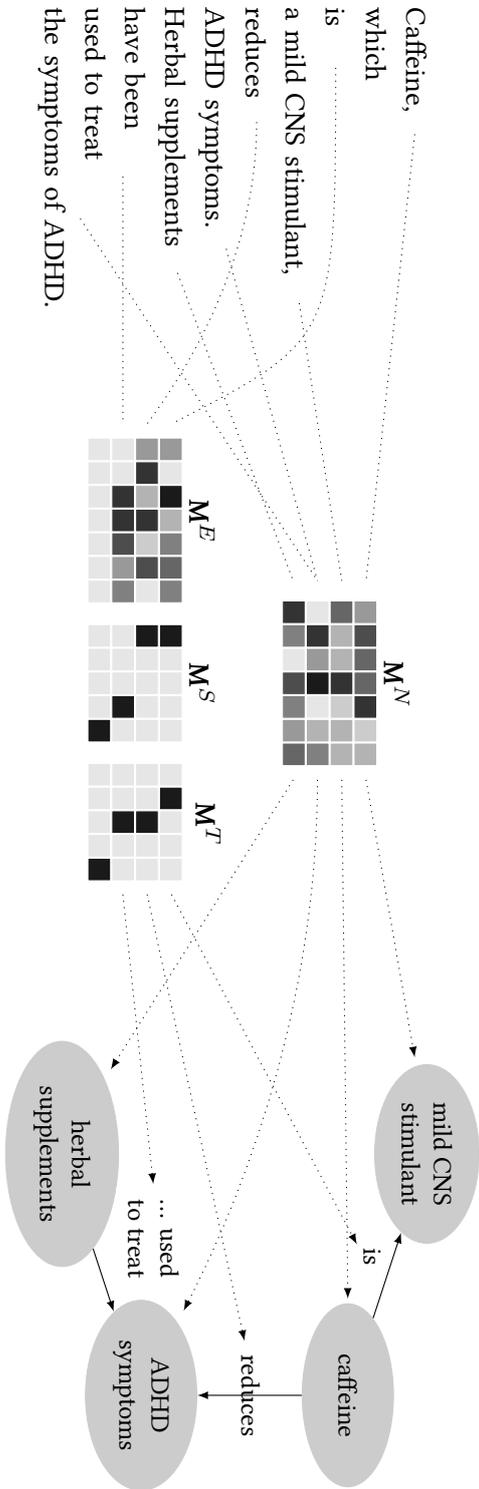


Figure 7.1: Conceptual illustration of our memory-based graph representation. The shown memory matrices M^N , M^E , M^S , M^T can store graphs with up to four nodes and four relations (NOP cells omitted). After the text on the left has been processed by the encoder, the final graph representation in the memory should be as indicated by the dotted lines. Note that the two mentions of the same concept get mapped to the same memory cell and that a cell in the edge memory remains unused. During decoding, labels for nodes and edges are generated from the cells of M^N and M^E and connected as determined by M^S and M^T . The empty edge cell immediately produces a stop symbol.

Compared to our representation, Johnson (2017)’s work uses an additional matrix for nodes and a tensor for edges to represent their types, but does not have dense representations for edges. This is due to the fact that they aim to represent typed nodes and typed, unlabeled edges, whereas we want to encode an untyped labeled graph. The preceding work of Li et al. (2016b) uses only a node matrix, as the full graph is given as input and only distributed representations for nodes are learned.

7.4.2 Sequence-to-Graph Networks

As explained earlier, the key idea of our sequence-to-graph network is that it learns to map from an input sequence to a labeled graph via the structured memory that we just introduced. More formally, let \mathbf{x} be the input sequence that should be summarized. We then formulate CM-MDS as the structured prediction problem of finding

$$G = \arg \max_{G' \in \mathcal{G}(\mathbf{x})} \text{score}(G') \quad (7.2)$$

where $\mathcal{G}(\mathbf{x})$ denotes the (infinite) set of possible valid graphs that can be constructed from \mathbf{x} and $\text{score}()$ is a function determining how well each of them summarizes the input.

To make that search tractable, we factor the scoring function into several probability distributions conditioned on the input that are learned by the sequence-to-graph network. In the following section, we describe each of its components in detail. Figure 7.2 shows an unrolled version of the model, illustrating how the encoder, memory and decoder interact.

7.4.2.1 Sequence Encoder

The task of the sequence encoder is to compose a hidden representation of the input sequence and learn to recognize mentions of concepts and relations in it. Each token x_1, \dots, x_ℓ in \mathbf{x} is represented by a word embedding. These representations are then recurrently processed by a GRU (Cho et al., 2014). The input representation at timestep t is given as

$$\mathbf{h}_t^F = GRU^F(\mathbf{h}_{t-1}^F, \mathbf{E}_{[x_t]}) \quad (7.3)$$

where \mathbf{E} is the word embedding matrix. In addition to GRU^F , we use a second cell GRU^B to process the sequence backwards and concatenate the hidden states to $\mathbf{h}_t = \mathbf{h}_t^F \parallel \mathbf{h}_t^B$.

7.4.2.2 Memory Module

Similar to the sequence encoder, the memory module is also a recurrent component. At each timestep t , it receives the current encoder state \mathbf{h}_t as input and updates the memory state. If the token x_t is part of a concept or relation mention, its task is to compare it with

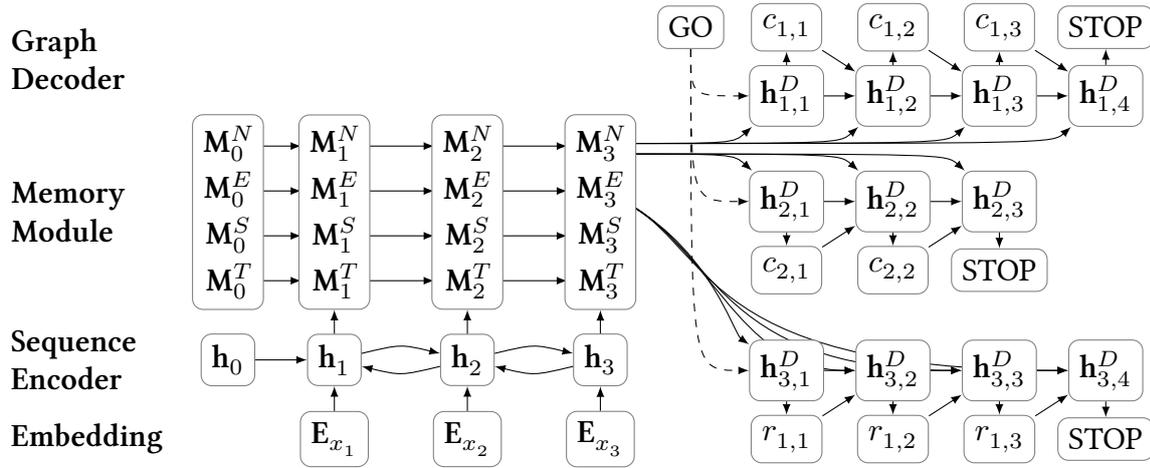


Figure 7.2: Sequence-to-graph network unrolled for a small example. An input sequence of three tokens is encoded and the memory is updated at each step. Conditioned on the final memory state, the graph decoder then generates two concept labels and one relation label token by token..

the current memory state to resolve coreferences, create a new node or edge in the graph if necessary and forget parts of the graph if the memory reaches its capacity.

Addressing Mechanism To decide which cell of the memory to access at a specific time-step, the model computes $\alpha = \text{add}(\mathbf{M}, \mathbf{h})$, an attention vector of values $\alpha_{[i]} \in [0, 1]$ per cell summing to 1, i.e. a probability distribution over the memory cells. Following previous work on NTMs (Graves et al., 2014, Gulcehre et al., 2018), we use the idea of content-based addressing for this step.⁷³ The attention vector is computed as the cosine similarity $C(\cdot; \cdot)$ between a transformed encoder state and every cell followed by a softmax.

$$\mathbf{k} = \mathbf{W}_c \mathbf{h} + \mathbf{b}_c \quad (7.4)$$

$$\mathbf{s}_{[i]} = C(\mathbf{M}_{[i]}; \mathbf{k}) \quad (7.5)$$

$$\alpha^c = \text{softmax}(\beta_c \mathbf{s}) \quad (7.6)$$

Parameters $\mathbf{W}_c \in \mathbb{R}^{m \times h}$, $\mathbf{b}_c \in \mathbb{R}^m$ and $\beta_c \in \mathbb{R}$ are learned, h is the encoder state size.

Note that initializing the memory with zeros at $t = 0$ would result in a uniform addressing distribution over all cells at $t = 1$. To make sure the model clearly chooses a cell even at this point, we initialize a memory matrix \mathbf{M} at $t = 0$ to a variable \mathbf{M}_0 which is of the same size and a trained variable. In other words, a desirable initial state of the memory is learned by the model during training.

⁷³We also experimented with additional location-based addressing, but did not observe substantial differences in the experimental setup described later in this section.

Nodes At timestep t , we use the previously described addressing mechanism to obtain the attention vector $\alpha_t^N = \text{add}^N(\mathbf{M}_{t-1}^N, \mathbf{h}_t)$ and then update the i -th memory cell following the memory update rule proposed by Graves et al. (2014):

$$\mathbf{M}_{t[i]}^N = (1 - \mathbf{e}_i \alpha_{t[i]}^N) \odot \mathbf{M}_{t-1[i]}^N + \mathbf{a}_{[i]} \alpha_{t[i]}^N \quad (7.7)$$

The two m -dimensional gate vectors are given by

$$\mathbf{e}_i = \sigma(\mathbf{W}_e [\mathbf{M}_{t-1[i]}^N \parallel \mathbf{h}_t] + \mathbf{b}_e) \quad (7.8)$$

$$\mathbf{a}_i = \tanh(\mathbf{W}_a [\mathbf{M}_{t-1[i]}^N \parallel \mathbf{h}_t] + \mathbf{b}_a) \quad (7.9)$$

with parameters $\mathbf{W}_e, \mathbf{W}_a \in \mathbb{R}^{m \times (m+h)}$ and corresponding bias vectors. While the first gate \mathbf{e}_i controls what should be erased from the memory, the gate \mathbf{a}_i defines the newly added information. This is very similar to updating the hidden state of a GRU.

Inspired by Gulcehre et al. (2018), we also add a NOP (No Operation) cell to each memory matrix. If the model does not want to update any cell at some point, it can simply attend to this additional one which is ignored during the later decoding step.

Edges Similar to nodes, the model also computes an attention $\alpha_t^E = \text{add}^E(\mathbf{M}_{t-1}^E, \mathbf{h}_t)$ and computes \mathbf{M}_t^E using the same gated update mechanism. Own sets of parameters are used for node and edge updates. For the edge's source and target nodes represented by \mathbf{M}_t^S and \mathbf{M}_t^T , the model uses the attention mechanism common in sequence transduction models (Luong et al., 2015) over the previous timesteps $t' \in [1, t-1]$ to compute

$$s_{t'} = \mathbf{h}_t \mathbf{W}_S \mathbf{h}_{t'} \quad (7.10)$$

$$\gamma = \text{softmax}(\mathbf{s}_{1:t-1}) \quad (7.11)$$

with parameters $\mathbf{W}_S \in \mathbb{R}^{h \times h}$ and then uses the attention weights to compute a weighted average of previous node memory attention vectors

$$\alpha_t^S = \sum_{t'} \alpha_{t'}^N \gamma_{t'} \quad (7.12)$$

Correspondingly, a timestep-weighted average α_t^T of node memory attentions is computed over the following timesteps $t' \in [t+1, \ell]$. The motivation for this is that whenever a relation mention occurs, the two concepts it refers to are usually mentioned right before and after it. We identify these concepts by looking at the previous and following node memory attentions. With the attention mechanism, the model can learn how far to look backwards and forward to find the relevant concepts.

The resulting distribution vectors are then stored for the currently modified edge as determined by the addressing vector α_t^E over the edge memory:

$$\mathbf{M}_t^S = (1 - \alpha_t^E)\mathbf{M}_{t-1}^S + \alpha_t^E \alpha_t^S \quad (7.13)$$

$$\mathbf{M}_t^T = (1 - \alpha_t^E)\mathbf{M}_{t-1}^T + \alpha_t^E \alpha_t^T \quad (7.14)$$

7.4.2.3 Graph Decoder

Once the complete input sequence has been encoded, we use the final memory state to decode a graph. For each node i , we decode a label c_i by conditioning a GRU on the i -th node memory cell to model the label probability as:

$$p(c_i | \mathbf{x}) = \prod_j p(c_{i,j} | c_{i,0:j-1}, \mathbf{M}_{\ell[i]}^N) \quad (7.15)$$

$$p(c_{i,j} | c_{i,0:j-1}, \mathbf{M}_{\ell[i]}^N) = \text{softmax}(\mathbf{E}^D \mathbf{h}_{i,j}^D) \quad (7.16)$$

$$\mathbf{h}_{i,j}^D = \text{GRU}^D(\mathbf{h}_{i,j-1}^D, [\mathbf{E}_{[c_{i,j-1}]}^D || \mathbf{M}_{\ell[i]}^N]) \quad (7.17)$$

Here, $c_{i,j}$ denotes the j -th token of the label c_i , $c_{i,0}$ is a special start symbol and \mathbf{E}^D a learned embedding matrix of the output vocabulary. We decode for each node until the stop symbol is produced and use the same GRU across cells.

Analogously, we also decode labels r_i from the edge memory cells, reusing the same GRU decoder as for the nodes to find the most likely sequence according to:

$$p(r_i | \mathbf{x}) = \prod_j p(r_{i,j} | r_{i,0:j-1}, \mathbf{M}_{\ell[i]}^E) \quad (7.18)$$

Note that the decoding process until this point is similar to a sequence transduction model with the only difference that instead of generating just a single sequence, we apply the RNN repeatedly to all node and edge memory cells.

To determine the source node $s(r_i)$ and target $t(r_i)$ for a decoded edge r_i , we model them as a probability distributions over all nodes as follows:

$$p(s(r_i) | \mathbf{x}) = \mathbf{M}_{\ell[i]}^S \quad (7.19)$$

$$p(t(r_i) | \mathbf{x}) = \mathbf{M}_{\ell[i]}^T \quad (7.20)$$

Note that due to their computation, the cells of \mathbf{M}^S and \mathbf{M}^T are already valid probability distributions and can be used as present in the memory.

7.4.2.4 Training

We train the model with pairs (\mathbf{x}, G^*) of input sequences and target graphs. For the CM-MDS task, that means G^* is the summary concept map while \mathbf{x} , similar as in the previous

chapter, is a sequence produced by pre-summarizing the multi-document input to a length that can be processed by the neural model.

For each type of prediction the model makes, i.e. node (7.15), edge (7.18), source (7.19) and target (7.20) predictions, we compute the cross-entropy loss against the gold data G^* . For labels, we normalize over the length of the sequence. These partial losses are averaged over nodes and edges and then combined into the overall training loss for an instance (G, G^*) with respect to the set of all parameters θ :

$$\mathcal{L}(G, G^*, \theta) = \frac{1}{3}\mathcal{L}_N + \frac{1}{3}\mathcal{L}_E + \frac{1}{6}\mathcal{L}_S + \frac{1}{6}\mathcal{L}_T \quad (7.21)$$

The loss can be minimized using stochastic gradient descent or methods such as Adam (Kingma and Ba, 2015). Note that although we only train the model to make several independent local predictions, the shared encoding network and the joint training still enable the model to learn dependencies between these parts. Similar training paradigms were shown to be effective in other structured prediction tasks (Goldberg, 2017).

An important question when defining the loss is how nodes and edges, which are sets with no particular order, should be handled. We sort them according to their first mention and assign them to ascending memory cells. This ordering requires the model to also use the cells in ascending order during creation, a consistent strategy that is easy to learn (Vinyals et al., 2015a). If a target graph has less nodes or edges than the memory capacity, we train the decoder to immediately generate the stop-symbol from the unused cells.

We train the model with minibatches of similarly long sequences and shuffle their order between epochs. The label decoder is trained with teacher forcing, i.e. we provide the previous gold token at every decoding timestep. For regularization, we add dropout after embedding the input, the sequence encoder and the decoder input.

7.4.2.5 Inference

At inference time, we use the predictions of our model to find the best graph G given \mathbf{x} as given in Equation 7.2. We define a graph’s score as the sum of the probabilities of its components:

$$score(G) = \sum_{i=1}^{|C|} p(c_i | \mathbf{x}) + \sum_{i=1}^{|R|} (p(r_i | \mathbf{x}) + p(s(r_i) | \mathbf{x}) + p(t(r_i) | \mathbf{x})) \quad (7.22)$$

Although this function decomposes nicely along the different predictions, the maximization is non-trivial as it has to be done under the connectedness constraint of CM-MDS.

We propose an ILP to find the best graph given the predictions. Let $x_{s,ij}, x_{t,ij}, i \in [1, e], j \in [1, n]$ be binary decision variables for the source and target assignment of each edge. We want to find values for these variables that maximize Equation 7.22 under several

constraints. Every edge can only have one source and one target node

$$\sum_j x_{s,ij} = 1 \quad \forall i \in [1, e] \quad (7.23)$$

$$\sum_j x_{t,ij} = 1 \quad \forall i \in [1, e] \quad (7.24)$$

and the same node cannot be selected as the source and target of one edge.

$$x_{s,ij} + x_{t,ij} = 1 \quad \forall i \in [1, e], j \in [1, n] \quad (7.25)$$

Additional constraints define that the resulting graph should be connected, which can be expressed using flow variables as shown in Section 6.4.1.

Further, we also add binary decision variables $x_{c,ij}, i \in [1, n], j \in [1, k]$ for the k -best concept labels, of which at most one can be selected per node. They serve two purposes: First, it allows the ILP to select between the possible labels for a node. In cases where the most probable labels for two nodes are the same, i.e. they would be the same concept in the resulting graph, it can choose a second-best label for one of them to obtain a graph with two distinct concepts and thus a potentially higher score. And second, since the connectedness constraint might lead to a best graph that does not contain all nodes, these variables are needed to reflect that in the scoring function. All edges, on the other hand, will always be part of an optimal solution and therefore do not have to be modeled explicitly in the ILP.

7.4.2.6 Subtasks in End-to-End Approach

Having described all aspects of the proposed neural model, we want to briefly summarize how it addresses the different subtasks of CM-MDS. Naturally, due to the approach of modeling the task end-to-end, no explicit components for different subtasks exist. Nevertheless, the design of the architecture was done with the task in mind and different features are meant to enable the model to handle specific challenges of CM-MDS.

Concept Mention Extraction Detecting concept mentions in the input sequence is mainly the task of the RNN encoder as well as the addressing and update mechanism of the memory. If a processed input token is found to be part of a concept mention, the node memory can be updated accordingly, whereas for other tokens, using the NOP cell or corresponding gate outputs can keep it unchanged.

Concept Mention Grouping Using the content-based addressing mechanism, the model compares the RNN representation of the current input token with each cell of the memory, allowing it to determine when concepts are mentioned repeatedly. It can thus learn to update the corresponding cells rather than to use additional ones, effectively resolving coreferences using the memory.

Relation Mention Extraction and Grouping The extraction and grouping of relation mentions is handled similar to concepts, using the corresponding memory matrix. By updating the source and target node matrices, a relation is associated to its concepts.

Concept and Relation Labeling Since the model is generative, labels are created during decoding and can be any sequence of tokens from the vocabulary. Thus, the model is able to generate abstractive concept maps.

Importance Estimation No explicit importance estimation is done. Instead, due to the limited capacity, the model always keeps just a summary-sized subset of the input in its memory. This can be seen as an alternative approach to summarization where the main task is to make local decisions of what to keep in the memory, add to it or what to override when reaching capacity. As we mentioned earlier, this is inspired by Kintsch and van Dijk (1978)’s cognitive model and summarization systems designed accordingly (Fang and Teufel, 2016, Fang et al., 2016). In addition, some part of selecting the summary-worthy subset of the multi-document input is also handled by the pre-summarizer in our setup.

Concept Map Construction The construction of the summary concept map is handled by the model’s decoder and the subsequent ILP which makes sure a connected graph is created. Due to the limited memory size, the selection of concepts and relations already happens during encoding and the size constraint is always satisfied.

While these considerations are the motivation behind the different parts of the architecture and provide some intuition as to why the model should be able to handle the task in its full scope, there is of course no guarantee that the model, when being trained end-to-end, is actually able to learn the task nor that the different subtasks are handled as intended.

7.4.3 Experiments

We now present a first experiment in which we train the novel sequence-to-graph network on artificially created data. However, the data represents only a very restricted version of the CM-MDS problem. The main purpose of the experiment is to demonstrate the general applicability of the architecture and to illustrate how the model can make predictions in practice. In Section 7.5, we apply the model to our actual datasets introduced in Section 7.2.

Data We created a dataset of simple input sequences and output pairs on the ADHD topic that was already introduced in earlier examples. We used a small inventory of 10 concepts and 25 relations to sample graphs with 5 concepts and 4 relations. For each graph, a corresponding input sequence was constructed by combining the four propositions. On the input side, we introduced slight variations into the concept mentions by using plural or

singular forms and different determiners. The propositions are connected with different conjunctions or punctuation. An example of input and output is shown in Figure 7.3.⁷⁴ The full dataset contains 10k pairs of which all input sequences and output graphs are unique. Input sequences have on average 26.7 tokens. The vocabulary consists of 48 distinct tokens.

Model We trained a sequence-to-graph model with 32-dimensional embeddings, encoder states, memory vectors and decoder states on 9k of the generated examples. Embeddings are learned from scratch with the other parameters. The total number of parameters is 51,432. Memory matrices have a capacity of 5 nodes and 4 edges. We train the model with Adam (Kingma and Ba, 2015), using a learning rate of 0.001, and a batch size of 50.

Results When trained on the 9k training instances, the sequence-to-graph model converges to a loss of almost 0 after 6 epochs and is then able to produce correct graphs for all inputs. The performance is only slightly lower on the unseen 1k test instances, as both parts are drawn from the same small input and output space. Nevertheless, this demonstrates that the architecture can be successfully trained and that the model is able to produce labeled graphs that correspond to the input sequences.

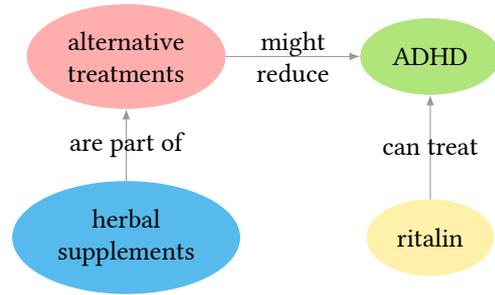
In Figure 7.3, we show how the trained model processes an input sequence. The four columns on the right illustrate the memory addressing vectors computed by the model for each of the input tokens shown in the first column. The example demonstrates that the model learned to recognize concept mentions, as the addressed node memory cell (second column) always changes when the first token of a concept mention is encountered (e.g. *alternative* or *ritalin*). When concepts are mentioned again, as in the last sentence, the corresponding memory cells are also addressed again. Similarly, the model also learned to use a new edge memory cell (third column) for each relation, which, in this simple example, corresponds to recognizing sentence boundaries.

Finally, the last two columns show that the model is also able to correctly determine source and target nodes of edges. When processing *might reduce*, for instance, the source node vector (fourth column), which is a weighted combination of the node addressing vectors computed for the preceding steps, focuses on the second cell as desired. The target node vector (fifth column) addresses the fourth cell, which is where the *adhd* concept is stored. Note that while the correct source and target nodes are not necessarily identified at all tokens of a relation mention, the resulting final memory state still leads to a correctly decoded graph in this example (and almost all other instances of this generated dataset).

While we think that training models and illustrating predictions on this simple dataset is very useful to better understand the capabilities and inner workings of the model, it is of course not the ultimate goal. In contrast to the data used here, the actual task will

⁷⁴For brevity, we show an example with only 4 concepts and 3 relations which is therefore not part of the actual dataset used in the experiment, but it has been generated in the same way.

Herbal supplements are part of alternative treatments. Ritalin can treat ADHD. Alternative treatments might reduce ADHD.



Token	Node Memory α_t^N (Eq. 7.6)	Edge Memory α_t^E (Eq. 7.6)	Source Node α_t^S (Eq. 7.12)	Target Node α_t^T (Eq. 7.12)
herbal supplements				
are				
part				
of				
alternative treatments				
.				
ritalin				
can				
treat				
adhd				
.				
alternative treatments				
might				
reduce				
adhd				
.				

Figure 7.3: Memory addressing vectors computed by our sequence-to-graph model. The model capacity is 5 nodes and 4 edges, resulting in 6- and 5-dimensional addressing vectors (last for the NOP cell). The model learned to use a new node memory cell whenever a new concept is mentioned and successfully readdresses them at coreferent mentions. The model further learned to address a new edge cell at every proposition boundary and to determine source and target nodes.

have longer input sequences and output graphs, a much larger vocabulary on both sides, a larger variety of mentions for the same concept, more tokens in the input that are neither a concept nor a relation mention and a noisier relationship between input and output. The next section carries out experiments on data that exhibits more of these properties.

7.5 End-to-End Experiments

As the final experiment of this chapter, we compare both the sequence transduction model and the sequence-to-graph model against our pipeline-based approaches in the end-to-end experimental setup that we already used in Section 6.5.

7.5.1 Experimental Setup

Following the setup for the sequence transduction model, we train the sequence-to-graph model on either EDUC_{SYN} or DMS_{SYN} and use 5% of the data for validation. We use the same pre-summarizer and tune the length of the input on EDUC-Train. The same vocabulary of around 50k tokens is used, embeddings are also initialized with GloVe (Pennington et al., 2014) and shared between encoder and decoder. The encoder state size is 150 per direction and memory cells have 300 dimensions. The memory has a capacity of 25 nodes and 27 edges, providing enough space for the gold graphs. We train with batches of size 10^{75} , a learning rate of 0.001 using Adam (Kingma and Ba, 2015) and a dropout probability of 0.2.

In Table 7.4, we report task-level performance using the ROUGE and METEOR metrics proposed in Section 3.5.2. As references for comparison, we include the corpus baseline (see Section 4.3.3) and improved pipeline (see Section 6.5) presented earlier. For the sequence transduction model, we include variations using different training data, linearizations and embedding setups. For the sequence-to-graph model, we show similar variations as well as the ILP decoding setup in which several beam hypotheses are considered for every node and edge. Since the results of both neural approaches are clearly behind the performance of the pipeline-based model, we omit additional results on WIKI. Instead, we also report performance on EDUC-Train to highlight the topic shift problem. Note that neural models are not trained on EDUC-Train, however, the pre-summarizer is tuned on that data.

7.5.2 Quantitative Results

Overall Results As the results in Table 7.4 show, both neural approaches can currently not compete with the pipeline-based models. As already discussed for the sequence transduction model in Section 7.3.3, a big challenge is the topic shift between train and test sets of EDUC and the correspondingly small overlap of vocabulary, which limits the performance

⁷⁵10 is the biggest batch size possible given the memory size of the GPU used for training.

Approach	Educ-Train						Educ-Test					
	METEOR			ROUGE			METEOR			ROUGE		
	Pr	Re	F1	Pr	Re	F1	Pr	Re	F1	Pr	Re	F1
Corpus Baseline							15.12	19.49	17.00	6.03	17.98	8.91
Improved Pipeline							15.14	17.34	16.12	9.37	11.93	10.38
Sequence Transduction												
DMSYN-Triples-Tuned	21.56	9.40	13.09	14.80	1.98	3.22	19.04	9.74	12.65	13.47	2.18	3.62
EducSYN-Triples-Tuned	31.93	19.38	24.00	36.50	12.36	18.29	9.72	7.17	8.17	2.56	0.78	1.16
EducSYN-Triples-Trans	37.14	24.68	29.23	41.99	19.34	26.14	11.62	8.46	9.74	2.66	0.82	1.21
EducSYN-Triples-Fixed	29.34	18.62	22.41	33.09	11.75	16.75	11.16	7.75	8.99	3.37	1.12	1.62
EducSYN-Concepts-Tuned	20.44	18.67	19.48	26.50	16.79	20.50	8.78	7.88	8.25	1.86	1.00	1.29
Sequence-to-Graph												
DMSYN-Tuned	14.84	10.39	11.94	12.89	3.10	4.53	16.53	11.15	12.96	11.98	2.45	3.70
EducSYN-Tuned	26.79	18.06	21.48	28.73	14.88	19.18	10.11	7.75	8.71	1.45	1.00	1.14
EducSYN-Tuned-Beam	25.27	21.18	23.00	25.31	21.53	23.11	9.49	8.17	8.72	1.31	1.35	1.30
EducSYN-Trans	28.27	17.86	21.72	39.10	12.86	19.10	13.75	8.66	10.52	4.57	1.55	2.29
EducSYN-Trans-Beam	24.83	20.73	22.49	28.70	18.99	22.66	12.00	0.92	10.37	3.04	1.88	2.31

Table 7.4: End-to-end results on the train and test sections of Educ for pipeline and neural approaches. Bold face highlights the best F1-score per group. Sequence transduction results are the same as reported in Table 7.2 and Table 7.3 and repeated for easier comparison.

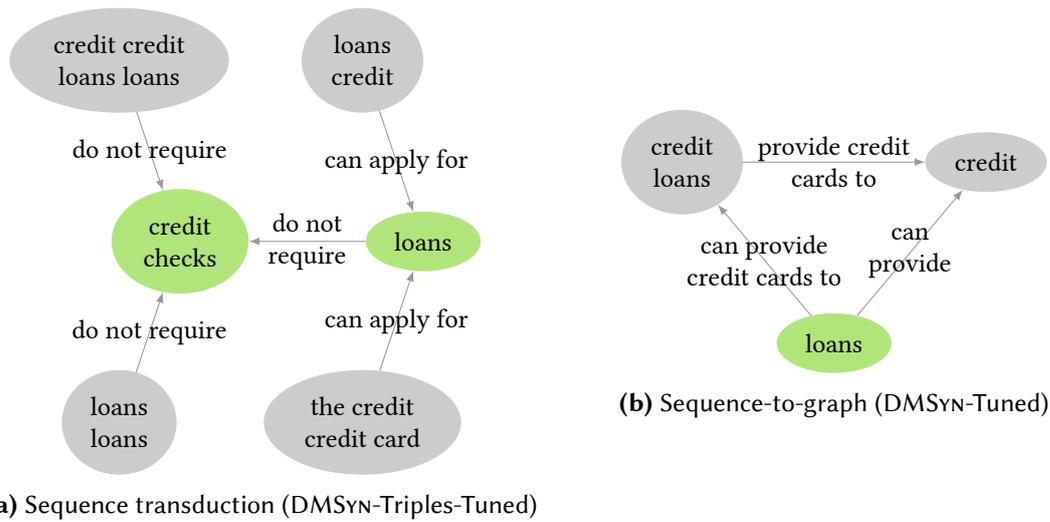


Figure 7.4: Summary concept maps predicted for the test topic “*students loans without credit history*”. Refer to the maps in Figure 6.2 (pipeline) and Figure 4.5 (reference) for comparison. Predicted maps are shown completely. Concepts in green (semantically) match a reference concept.

of the sequence-to-graph model similarly. As a consequence of that, this experimental comparison also makes it hard to draw any conclusion about the superiority of either the sequence transduction or sequence-to-graph model. For both, the version trained on the broader DMSYN transfers best to the test data, with only marginally different scores (not significant). We expect the difference between the two architectures to be more pronounced when they are trained with broader and higher-quality data.

Graph Decoding A main difference between the sequence transduction and sequence-to-graph model is how concept maps are predicted. Due to the ILP decoding, all graphs of the sequence-to-graph model are always guaranteed to be connected and thus no disconnected parts need to be discarded. As a result, the predicted graphs of the model trained on DMSYN have on average 8.6 nodes and 13.7 edges, while the corresponding sequence transduction model produces smaller graphs with only 5.1 nodes and 4.3 edges. The models trained on EDUCSYN predict graphs of around 11 nodes using both architectures. Here, using the variant of the decoding ILP that chooses among the top-10 beam hypotheses for each node label lets the sequence-to-graph model predict even bigger graphs, having more than 17 nodes on average. However, we observed that this setup often introduces several nodes with very similar labels, i.e. concepts that are redundant and not grouped correctly. Given that the scores of these bigger graphs are also not substantially better on EDUC-Test, using the most probable label seems to be preferable. In general, both architectures seem to have problems predicting graphs as large as the reference maps in the current setup.

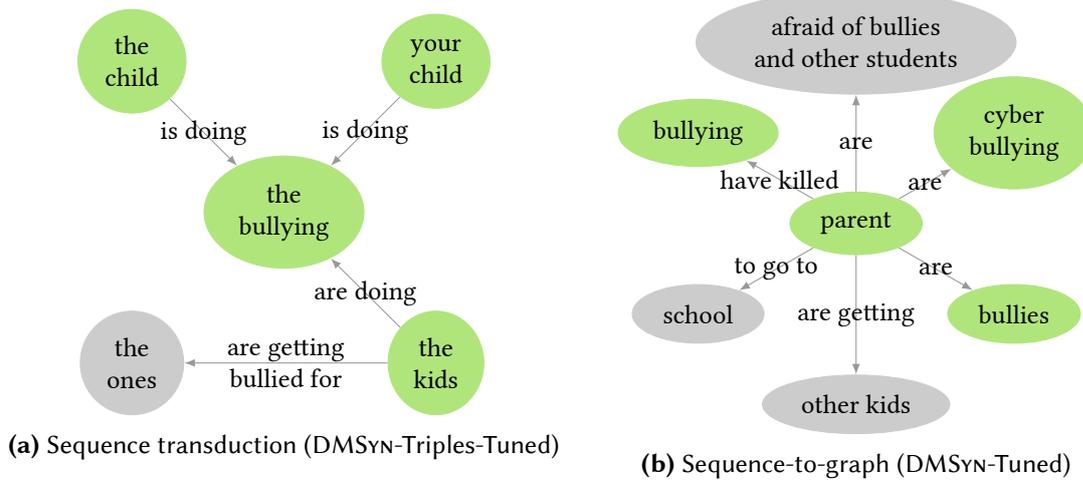


Figure 7.5: Summary concept maps predicted for the test topic “*parents dealing with their kids being cyber-bullied*”. Map (b) is only shown partially, the complete predicted concept map has 13 concepts and 12 relations. Concepts in green (semantically) match a reference concept.

7.5.3 Qualitative Analysis

To better understand the challenges that our neural models face and to determine whether there is any qualitative difference between the concept maps produced by the two alternative architectures, we also manually looked at the concept maps predicted on the test set topics. Figure 7.4 and Figure 7.5 show summary concept maps that have been predicted by models trained on the DMSYN dataset with the two architectures.

Comparison of Architectures The manual inspection confirmed the aforementioned difference that maps predicted by the sequence-to-graph model tend to be bigger. However, the sizes for both architectures vary a lot across the topics, for instance, the map in Figure 7.4 (b) has only three concepts although it is produced by the sequence-to-graph network. For other topics, the sequence transduction model predicted maps with only one concept. A problem specific to the sequence-to-graph model is that it sometimes uses a single relation label for many relations of the map. In Figure 7.5 (b), three relations (seven in the full map) use the label *are*. We observed that such a behavior occurs when the model does not clearly address specific memory cells, resulting in several cells with similar contents and correspondingly similar labels after decoding. However, apart from these observations, we did not find any consistent qualitative differences that indicate whether one of the two architectures is clearly superior.

Extraction and Grouping The analysis further revealed several issues that occur with both architectures but that are unique to neural models, explaining to some extent the lower quality of the maps compared to those created by the pipeline. Repetitions in concept

labels, such as *credit credit loans loans* (Figure 7.4 (a)), and propositions that are rather meaningless (*the kids - are getting bullied for - the ones*, Figure 7.5 (a)) or not asserted by the input text (*parent - have killed - bullying*, Figure 7.5 (b)) are part of the predicted summary maps. Since the models can freely generate concept and relation labels token by token, the chance that meaningless and factually wrong information is introduced is higher than in fully extractive approaches. Another issue are duplicate concepts as in Figure 7.5 (a), where *the child*, *your child* and *the kids* all refer to *children*, which can be attributed to the low quality of the synthetic training data. Since the examples have been created with the pipeline approach, which does not always group concept mentions correctly (see Section 6.5), it is very difficult – if not impossible – for the neural model to learn to correctly group mentions based on these noisy examples.

Summarization With regard to content selection, the predicted summary concept maps show encouraging results. As shown in the example, all included concepts are typically related to the topic, most of them are very central concepts and a large fraction indeed matches the reference map. However, as we mentioned earlier, the predicted concept maps tend to be smaller than the reference maps and it is thus unclear how well content would be selected for bigger maps. Models that are trained on EDUCSYN show a different behavior (not shown in any of the figures): The models trained on that data tend to include similar concepts in all predicted maps independent of what the input text is and the maps are therefore not very useful. This is a manifestation of the topic shift problem discussed earlier, which makes it difficult for the neural models to predict the topic-specific gold concepts of the test topics as they have not been seen at training time.

7.6 Chapter Summary

In this chapter, we explored approaches to model CM-MDS end-to-end as a single machine learning problem with neural networks. This paradigm recently led to new state-of-the-art models for many other NLP tasks. However, as we pointed out, several challenges make such a modeling approach difficult for our task: Only little training data is available, very large inputs have to be processed and no neural architecture to predict labeled graphs exist.

Towards that end, we made several contributions: First, we proposed a set of techniques that allow us to reduce CM-MDS to a sequence transduction problem and approach it with existing models for such problems. Second, we proposed sequence-to-graph networks, a novel neural network architecture that can directly predict labeled graphs. And third, we carried out a set of experiments that for the first time study the performance of neural end-to-end models on CM-MDS. While the overall performance of these neural models is not yet competitive with the pipeline-based approaches discussed earlier, the experiments are

an important first step towards developing such models and they point out the remaining challenges that have to be addressed by future work.

In order to model the task as sequence transduction, we proposed to use an extractive textual summarization system as a first step together with a neural model that can predict linearizations of graphs. In combination with generated synthetic training data, the task can then be approached with the same sequence-transduction models that have been successfully applied to tasks like machine translation. Testing that approach, our experiments revealed two important additional challenges: First, due to the topic shift between train and test sets of EDUC, neural models with generative word-level decoders have problems transferring to the test topics unless they have been trained on very large and diverse sets of examples. Our synthetic training data cannot satisfy this requirement. And second, sequence transduction models have problems predicting connected graphs of the desired size, producing substantially smaller graphs in our experiments.

As an alternative architecture, we proposed a sequence-to-graph network that can directly predict labeled graphs. Using a novel decoder and memory-based graph representations with NTM-inspired updates, it does not need to work with linearizations of graphs. With an ILP-based decoding approach, the architecture can also decode graphs that are guaranteed to be connected as required for CM-MDS. However, the network faces the same topic shift challenge as the sequence transduction model, which makes it hard to draw strong conclusions from the experimental results regarding the benefits of the architecture.

Despite the limited performance of both end-to-end approaches in our current experiments, we believe that it is a direction worth studying further. Recent work on related tasks studies similar ideas as ours, such as extractive pre-summarization of large inputs for neural models (Liu et al., 2018) and graph-structured neural networks (Battaglia et al., 2018). Hierarchical encoders for MDS inputs, as recently explored by Cohan et al. (2018), Celikyilmaz et al. (2018) and Zhang et al. (2018), could also be incorporated into our sequence-to-graph model in future work to reduce the dependency on pre-summarization. The biggest challenge for future work is, as demonstrated by our experiments, the availability of large high-quality training corpora. Towards that end, using weak or incidental supervision signals (Roth, 2017) instead of full concept maps could be a promising direction. Recent successes in transfer learning based on contextualized embeddings (Peters et al., 2018) or fine-tuning language models (Howard and Ruder, 2018, Devlin et al., 2018) are a second technique that could help to deal with the lack of large training corpora.

CHAPTER 8

Exploratory Search with Concept Maps

After the previous three chapters focused on computational models for CM-MDS, this chapter briefly looks at how summary concept maps obtained with the models can be used in practice to support exploratory search. We present a prototype of an application for this purpose and report results from an initial user study.

8.1 Motivation

As we explained in Chapter 2, the research presented in this thesis is motivated by the use case of exploratory search. When a user explores a document collection with a complex information need, an automatically created summary concept map can support the user by providing a concise overview and navigation functionality to browse the collection.

In the following section, we present a prototype of an application for this purpose. However, we want to point out that the specific design of the system is just one potential way to use summary concept maps, whereas the exploration of other options in terms of interface design and usability are left for following work in the respective research communities and commercial software engineering endeavors. As part of this thesis, the main purpose of this chapter is not to design the best document exploration system but to demonstrate a potential application of the research described in the previous chapters.

8.2 Exploratory Search System

The prototype system for concept map-based document exploration was designed in the style of well-known search engine interfaces. As shown in Figure 8.1, the list of search results is complemented by a visualization of a summary concept map that has been extracted from the retrieved documents. This type of integration follows the paradigm of

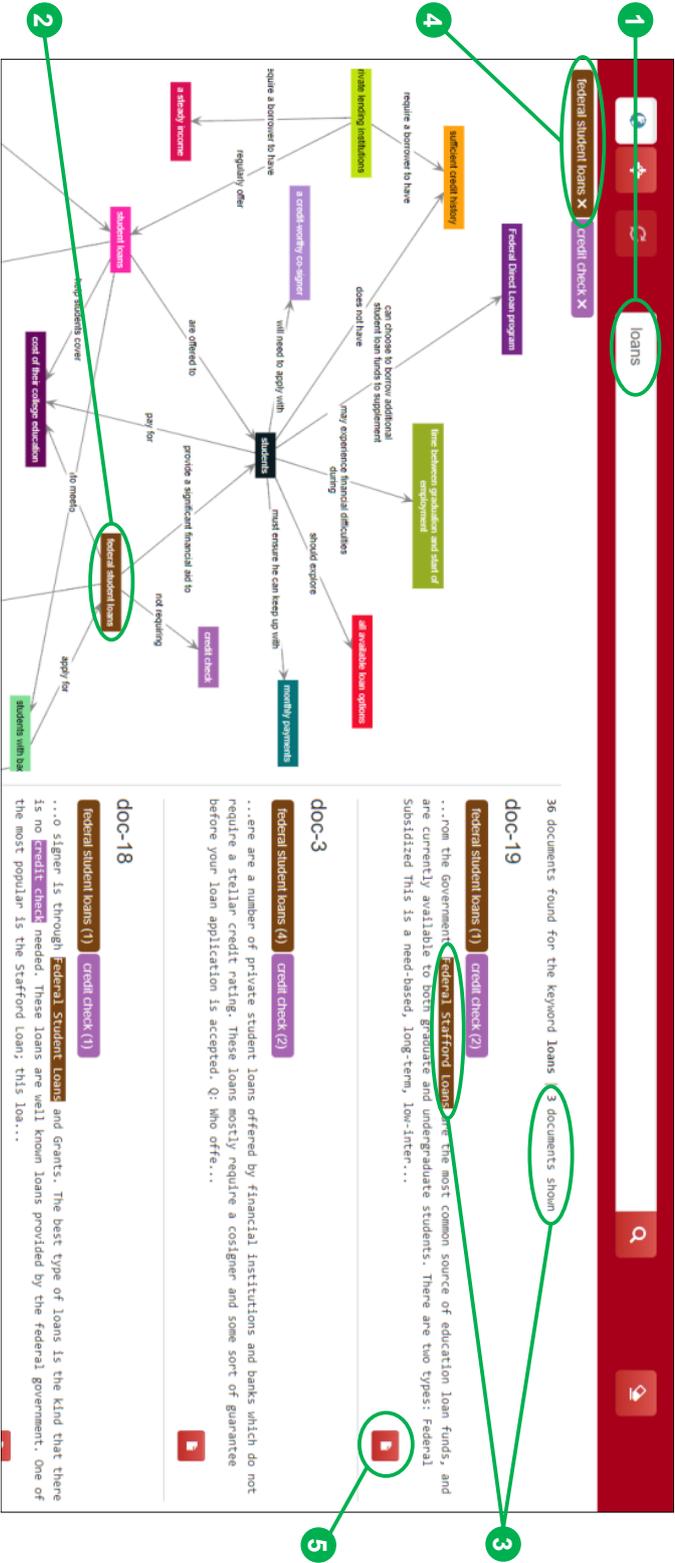


Figure 8.1: Prototype of the concept map-based document exploration system. After entering a search term (1), the system displays retrieved documents (right) and a concept map built from them (left). When clicking on a concept or relation (2), documents are filtered and highlighted (3) according to the spans associated with the selected element. Active filters are shown at the top (4). Opening a document (5) shows its full text with all highlighted spans.

faceted search, in which different taxonomies, either predefined or extracted from the results (Hearst and Stoica, 2009), are offered along with the results to filter them. Instead of the typically small, single-level taxonomies, our application takes this idea further by offering a summary concept map for filtering.

Functionality After executing a query, a user can both scroll through the list of retrieved documents or navigate through the summary concept map. Every concept and relation in the map is associated with at least one mention in the retrieved documents, as determined during the extraction of the map from the text. If a user selects an element in the map, the corresponding mentions are highlighted in the document snippets and the results are filtered to the subset of documents that contain at least one mention. Filters can be combined, which reduces the documents to those containing mentions for all selected elements. They can also be temporally deactivated and can be removed completely. In the result list, the number of corresponding mentions for each filter is displayed as a tag above the document. Colors of concepts in the map, filter tags and highlighted mentions match. In the result list, a user can switch to the full text view of a document, which also contains highlights for mentions according to the currently selected filters.

Graph Layouts The application currently provides two different concept map visualizations that we found to yield useful renderings for maps of different sizes. At any point, a user can switch between them using the buttons in the top left corner. The *full* layout is a force-directed layout showing the complete map. It allows the user to zoom in and out and pan to fully inspect the concept map. While it has the advantage that it can provide an overview of the complete map, the visualization can become complex for large maps. As an alternative, the application offers a *focused* layout. It shows only one focus concept and its direct neighbors at a time, while every neighbor concept has a number indicating how many more concepts are related to it. Selecting one of the neighbors moves that concept to the center and displays its neighbors. This allows going through the concept map step by step and it usually yields much cleaner visualizations, as the number of visible concepts is limited. The modular design of the application allows adding additional visualizations.

Interaction Example To illustrate the capabilities of the system, consider the following hypothetical interaction of a user with it: Exploring a collection of documents about student loans, the user issues the query *credit check*. The system then displays corresponding documents on the right, represented by short snippets, and a summary concept map for these documents on the left. The user zooms into the map to inspect it in detail. They then select one of the concepts, which automatically creates a filter that reduces the result list correspondingly. From the filtered result list, they open a specific document and read it. In the document, all mentions of the selected concept are highlighted. The user then continues

```

2018-10-18 10:04:11 SEARCH "credit check", results: doc-0, doc-8, doc-3, ...
2018-10-18 10:04:17 ZOOMED visible concepts: 42,65,89,57,35,24
2018-10-18 10:04:18 PANNED visible concepts: 65,89,35,24,36,47,89,92
2018-10-18 10:04:30 CONCEPT_CLICKED concept: 24
2018-10-18 10:04:30 FILTER_ADDED concept: 24
2018-10-18 10:04:32 RESULTS_SCROLLED visible: doc-3, doc-26, doc-17
2018-10-18 10:04:35 DOC_OPENED doc-17
2018-10-18 10:04:53 DOC_SCROLLED visible lines: [516-1468]
2018-10-18 10:05:33 BACK_TO_RESULTS
2018-10-18 10:05:40 LAYOUT_SWITCHED focused
...

```

Figure 8.2: Example for a user interaction log of the exploration system.

the search by closing the document and switching to the alternative graph visualization to further explore the document set.

To be able to thoroughly study how users would use a concept map for document exploration, the system is able to accurately log all interactions when used in experimental studies. Figure 8.2 illustrates how such a log looks like for the session described above. Every interaction event is captured with a time stamp, its type and relevant parameters.

8.3 User Study

To verify whether the user interface and interaction of the presented system is in line with user expectations, we conducted a first preliminary user study. 20 researchers from our lab and students from the university participated. They used the application to explore a collection of web pages on student loans (as in Figure 8.1) and answered a questionnaire asking for feedback on different parts of the system.

The results showed that the system was perceived as being very intuitive. Subjects could easily interpret the meaning of the concept map and how it can be used to filter and highlight the documents. With regard to the different layouts, 60% preferred the focused layout because it was “clearer” and “less cluttered”, while only 15% preferred the full layout, the rest being undecided. However, several subjects noted that the full layout is still useful to get the big picture, advocating to offer both options in the system. In addition, the participants provided many useful suggestions to improve the system, e.g. adding tooltips, which have been incorporated into the current version.

8.4 Chapter Summary

In this chapter, we presented a first prototype of a concept map–based exploratory search system. In line with the requirements laid out in Chapter 2, the system supports exploratory search by providing a concise, structured overview of the content of a document collection and by allowing a user to navigate to specific details in the documents.

The summary concept map, which is a core part of the system, can be automatically generated with the computational models presented in Chapters 5, 6 and 7 as well as future improvements of them. Thereby, the prototype serves as a demonstration of how those models can be used for practical purposes. The positive initial user feedback on the system showed that the application scenario is realistic and that the system is intuitive to use. That is in line with the previous, more extensive studies reviewed in Section 2.2.2 that analyzed the use of concept maps in practice and observed benefits over other types of representations to support document exploration.

Beyond its purpose as a demonstrator of a specific application scenario, the system can also be used to conduct user studies that compare how useful concept maps created with different computational models are in practice. We suggested this as the third possible type of evaluation for CM-MDS in Section 3.5.2. Using different computational models with the same exploration system can ensure a controlled experiment with comparable settings across conditions. Further, the logging capabilities of the system allow detailed analysis of such experiments. Similar user studies have been carried out by Carnot et al. (2001) and Valerio et al. (2012) to compare concept maps with alternative representations.

As the work on CM-MDS is still at its beginning and very few models covering the whole task exist, there is currently no need to perform such comparisons. In this thesis, we presented two pipeline-based models (see Section 4.3.3 and Section 6.5) and several neural network-based end-to-end models (see Chapter 7). Since both the neural models and the corpus baseline produce concept maps of rather low quality and have already been found to be substantially weaker using automatic and manual evaluations, comparing them in a user study against the improved pipeline would add only very limited insights. However, as more well-performing methods are developed in future work, such experiments will become more interesting and the system presented here can be used to carry them out.

CHAPTER 9

Conclusions

„Science never solves a problem without creating ten more.“

— George Bernard Shaw

This final chapter summarizes the findings of the thesis and outlines promising directions for future research on CM-MDS and related tasks.

9.1 Summary of Findings and Contributions

This thesis has shown that the automatic creation of multi-document summaries in the form of concept maps is an important task and that many challenges for computational models arise that have not yet been adequately addressed in previous work. We therefore proposed new models for several subtasks of CM-MDS as well as different approaches to model the task as a whole. With several newly created corpora and suggested evaluation protocols, we conducted extensive experimental evaluations. And finally, we demonstrated the practical use of concept maps for exploratory search in a demo application.

Chapter 3 introduced the central problem studied in this thesis. Based on the review of user requirements, we argued that concept maps are a very useful text representation to support users during exploratory search. Given the limited amount of existing work on extracting them from text, we proposed to study the task of concept map-based multi-document summarization (CM-MDS). We presented all of its subtasks in detail and outlined the challenges that computational models for it have to face. We further proposed two automatic metrics based on METEOR and ROUGE to evaluate automatically created concept maps against manually created references. Because these metrics, similar as in traditional textual summarization, can only evaluate some aspects of the task, we additionally proposed manual evaluation protocols that complement the metrics.

Chapter 4 addressed the lack of suitable corpora to train and evaluate computational models for CM-MDS and the fact that the annotation is particularly time-consuming and complex. We explored two different directions, the automatic extension of existing partial annotations and the use of a new scalable annotation process relying on crowdsourcing. With regard to the second strategy, we developed a new corpus creation method that effectively combines automatic preprocessing, scalable crowdsourcing and high-quality expert annotations. Its crucial component is a novel crowdsourcing scheme called low-context importance annotation. Using it, we created a new corpus of 30 document sets on educational topics, each with around 40 source documents and a summarizing concept map, that served as the main evaluation dataset for experiments in the thesis.

Chapter 5 focused on the concept and relation extraction subtasks of CM-MDS. As the first contribution, we addressed the lack of a clearly established state-of-the-art among previously proposed extraction methods by carrying out a first direct comparison of such methods. The most interesting finding of these experiments is that previously proposed methods for relation mention extraction performed particularly poor on our datasets. In addition, we proposed to extract concept and relation mentions from predicate-argument structures instead of syntactic representations. We found that this alternative approach substantially improves the relation extraction performance while performing comparable to previous work for concept mention extraction and being substantially easier to implement. As a second experiment, we performed a case study of porting a rule-based predicate-argument analysis tool from English to German. Since most previous work focused on English, it is important to know how challenging and laborious it is to also obtain such systems for other languages. We found that with roughly 10% of the effort that went into the English system, we could build a variant for German covering 89% of the rules and provided an extensive discussion of the cases that we found to be more difficult to transfer.

Chapter 6 looked at the CM-MDS subtasks of concept mention grouping, importance estimation and concept map construction. For concept mention grouping, we proposed a novel solution based on pairwise mention classification and a subsequent partitioning step. Compared to previous work on concept map mining, this approach can capture more types of coreferences and successfully improved the quality of summary concept maps on our benchmark corpus. With regard to importance estimation, we studied a broad set of features and different machine learning approaches. While we could not observe clear advantages of modeling the problem as regression, classification or ranking, we did design supervised models that clearly outperform the exclusively unsupervised methods suggested in previous work. For concept map construction, we proposed an ILP formulation that allows us to find an optimal solution to the subgraph selection problem of CM-MDS. These optimal subgraphs are superior to heuristically selected ones on our evaluation corpus. We finally

presented a pipeline covering the whole CM-MDS task that incorporates the new models we developed for the different subtasks. We performed automatic and manual evaluations on two corpora and observed that the pipeline improves upon a range of methods proposed in previous work, defining a new state-of-the-art for the task.

Chapter 7 explored approaches to model CM-MDS end-to-end as a single machine learning problem with neural networks. Several challenges make such a modeling approach difficult: Only little training data is available, very large inputs have to be processed and no neural architectures to predict labeled graphs exist. We proposed a set of techniques that allow us to reduce CM-MDS to a sequence transduction problem and approach it with existing models for such problems. Further, we proposed sequence-to-graph networks, a novel neural network architecture that can directly predict labeled graphs. And third, we carried out a set of experiments that for the first time study the performance of neural end-to-end models on CM-MDS. While the overall performance of these neural models is not yet competitive with the pipeline-based approaches of the previous chapter, the experiments are an important first step towards developing such models and pointed out the remaining challenges that have to be addressed by future work. One of them is to obtain high-quality training datasets of sufficient size. The second part of this chapter outlines specific steps that could be taken in this direction.

Chapter 8 demonstrated how the computational models developed in the previous chapters can be used for practical purposes. We presented a first prototype of an exploratory search system that — using summary concept maps — provides a concise and structured overview of a document collection and allows a user to navigate to details in order to explore the content. Besides its function as a demonstrator, the system can also facilitate user studies as an extrinsic evaluation for CM-MDS in the future.

Figure 9.1 shows a summary concept map that was automatically created based on the content of this thesis.⁷⁶ Similar to the examples that were shown in Section 6.5 and the errors discussed there, this summary also reveals some open issues of the current methods: concepts are sometimes not grouped as much as it would be desired (e.g. *concept maps* and *maps*) and some extracted relations are not very clearly labeled (e.g. *tries to keep*). However, we think it also demonstrates that even at the current level of performance, the pipeline already produces summaries that can be of use for a user. As Figure 9.1 shows, the summary does in fact contain many of the central concepts discussed in this thesis and it also presents several important relations between them. In the following and final part of the thesis, we point out how the remaining issues could be addressed in the future.

⁷⁶We used the pipeline presented in Section 6.5 with models trained on EDUC and a size limit of 10 concepts. The input is a collection with one document per chapter, excluding this paragraph describing the result.

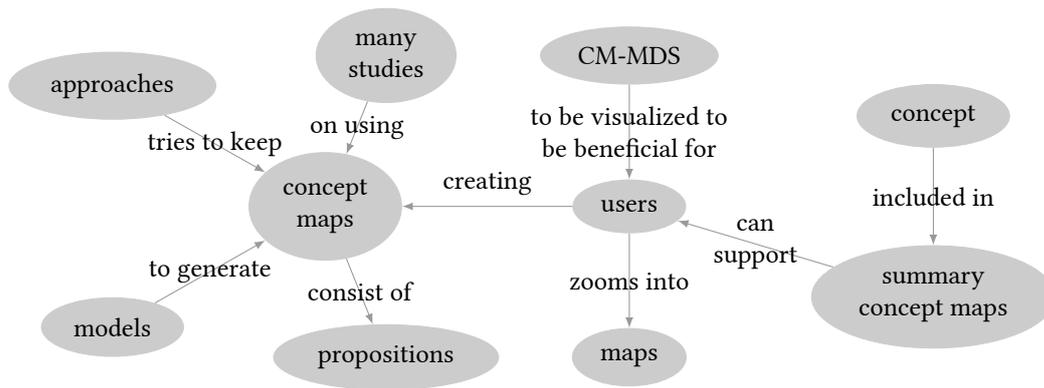


Figure 9.1: Summary concept map automatically created for the content of this thesis using the pipeline presented in Section 6.5 trained on EDUC with a size limit of $\mathcal{L}_C = 10$.

9.2 Future Research Directions

Given the research presented in this thesis, there are several directions to further improve the automatic creation of summary concept maps. As we pointed out in Chapter 3, the existing research in this area was limited when the work presented here was done and posed many open challenges. Naturally, a single thesis is not able to fill all the gaps and many open challenges are still left.

At the end of most of the preceding chapters, we already pointed out the remaining challenges for the different subtasks for CM-MDS. With regard to mention extraction, they include increasing the recall to capture more concepts and relations and also ensuring that no propositions that are not asserted by the text are being extracted. Sequence labeling techniques are an approach for this subtask that has not yet been explored for concept maps, but it requires corresponding annotated data. Finding the right concept granularity during concept mention grouping is another important open issue. In particular, handling entailment relations between mentions and having grouping approaches that adapt the granularity to the size of the summary are interesting extensions that we have not yet studied. And further, as one of the main bottlenecks of the pipeline, we identified the selection of important concepts for the summary. It is thus one of the most important issues to address more extensively. Moreover, the current approaches, in particular the best performing pipeline approach, can only scale to document collections of a limited size and can thus not be applied to large-scale document collections.

Additional subtasks and challenges of CM-MDS exist that we have not addressed at all or only handled with simple heuristics. Among them are relation mentions which could be grouped, scored and selected in a similar manner as concept mentions, including the explicit modeling of them during subgraph selection as proposed in Equation 3.1. And second, the labeling of concepts and relations could be approached with a technique that is

more sophisticated than simply choosing the most frequent mention. With the explored neural models that can freely generate labels we already made a step in this direction. Corresponding techniques could also be integrated in the pipeline approach. We have so far not focused on these additional subtasks because we consider the subtasks that we did address to be more important for the overall performance, but fully solving the CM-MDS task in the future would entail that solutions for all subtasks have to be developed.

In light of the variety of open challenges and potential for future improvements, we would like to point out a few directions that we consider to be particularly promising:

- Our experiments in Chapter 7 showed that the application of neural networks to CM-MDS is currently difficult and that this is mostly due to the lack of sufficient high-quality training data. To leverage the potential of neural models for the task, it seems worthwhile to explore options of training such models in low-resource settings. Other training paradigms, such as incidental supervision (Roth, 2017) or unsupervised learning (Dohare et al., 2018), could be solutions. Specifically, recent ideas of transfer learning based on contextualized embeddings (Peters et al., 2018) or fine-tuning language models (Howard and Ruder, 2018, Devlin et al., 2018) seem to be promising directions as they have been shown to be very powerful for other NLP tasks. Once better ways to successfully train a neural model for CM-MDS have been found, subsequent work can further study the use of graph-based neural networks for the task as proposed with our sequence-to-graph architecture. Architectures of this kind are currently explored for many different tasks (Battaglia et al., 2018) and future progress in this area can potentially also be transferred to CM-MDS.
- Instead of trying to model the whole CM-MDS task end-to-end with a single neural model, one could also approach the different subtask individually with neural models. This would yield a pipeline as described in Section 6.5 but with potentially more powerful models for each of the subtask. In addition, many of the problems faced in end-to-end modeling, such as the complex structure of the output and the large size of the input, are already handled by how the pipeline decomposes the task, leaving “simpler” subproblems for which models have to be learned. For those subproblems, the collection of suitable training data is potentially also easier and can rely on already existing datasets for related tasks such as entity recognition, keyword extraction or textual summarization. In addition, the alternative training paradigms discussed above would be equally applicable in this setup.
- Since we showed in Section 6.3.3 that a large gap exists between the upper bound and current performance in importance estimation, this subtask could be a particular focus of future work. However, most of the standard techniques for traditional summarization have already been applied. To make further improvements, additional ideas

such as incorporating external world knowledge seem to be promising. Several authors noted that the inherent importance that humans assign to certain propositions or concepts independent of a specific text plays an important role in summarization (Louis, 2014, Zopf et al., 2016a, Peyrard, 2018). In particular in the case of our benchmark corpus, for which importance annotations have been collected through crowdsourcing with minimal context, this notion of importance presumably plays a major role. Large background corpora or structured knowledge bases could be sources to obtain corresponding features that make a better importance estimation possible.

- An alternative direction to approach the problem of limited importance estimation performance would be to create personalized summary concept maps. A limitation of most summarization methods – no matter if they have been designed for SDS, MDS or CM-MDS – is that given the input text, they will always produce the same summary. In practice, however, different users approach a collection of documents with different information needs, background knowledge and preferences. They would therefore ideally need summaries focusing on different aspects of the content. Methods to create personalized textual summaries have been studied by Zhang et al. (2003), Berkovsky et al. (2008) and Park and An (2010). While these approaches need additional inputs describing a specific user’s interests, there is also recent work by P.V.S. and Meyer (2017) which tries to derive that information through interaction with a user. That is particularly interesting if data about a user is not available a priori or in our scenario of exploratory search, where it is often difficult for users to express their information need precisely (see Section 2.1.1). As our experiments in Section 6.3.3 showed, the identification of the most important concepts in a document collection is challenging, which can be partly attributed to varying interests of different users and a corresponding lack of agreement in reference annotations. The interactive modification of a summary concept map to a specific user’s interests is therefore another interesting direction for future work.

In addition to work aiming to improve computational methods for CM-MDS and the personalized variant of it outlined above, research that focuses on the use of automatically generated concept maps in practical applications will be crucial to ensure that this technology will ultimately successfully support real-world users. Towards that end, user studies that investigate how concept maps can be best integrated into existing document exploration tools, how they have to be visualized to be beneficial for a user and how their usage can be made more popular would be interesting projects. As we summarized in Section 2.2.2, several studies in this direction have been performed in the past. Once computational models for CM-MDS become more mature, it would be important to repeat such experiments using the then state-of-the-art models and summary concept maps that have been automatically created with them.

As we pointed out in Chapter 1, information is nowadays abundantly available and information overload a serious problem that many people face. In this thesis, we approached this problem by developing automatic structured summarization techniques that can support people during the exploration of document collections. The structured summarization task offers both interesting practical applications and enough potential for future research that is not yet matched with a corresponding amount of attention from the community. We hope that the work in this thesis laid the ground for more research on CM-MDS and similarly structured summarization problems and that the suggestions in this section will provide useful inspiration for interesting future work in this area.

Index

- abstractive concept map, 41
- abstractive summarization, 30

- concept, 16
- concept labeling, 23, 26
- concept map, 2, 16, 41
- concept map construction, 23, 26
- concept map mining, 22
- concept map–based multi-document summarization, 42
- concept mapping, 18
- concept mention extraction, 23, 24
- concept mention grouping, 23, 25
- crowdsourcing, 65

- encoder-decoder architecture, 145
- exploratory search, 2, 9, 12
- exploratory search system, 11, 12, 35
- extractive concept map, 41
- extractive summarization, 28

- importance estimation, 23, 26, 28
- information extraction, 31
- information overload, 1

- knapsack problem, 29

- low-context importance annotation, 65

- METEOR, 55
- multi-document summarization, 27

- neural network, 141
- neural supervised summarization, 28

- open information extraction, 32

- permutation test, 56
- proposition, 17, 45
- propositional coherence, 18

- query-focused summarization, 27

- randomization test, 56
- relation, 17
- relation labeling, 23, 26
- relation mention extraction, 23, 25
- relation mention grouping, 23, 25
- ROUGE, 56

- sentence selection, 28
- sequence transduction models, 145
- sequence-to-sequence models, 145
- significance test, 56
- single-document summarization, 27
- structured text representation, 11, 12
- summary concept map, 42
- supervised summarization, 28

- text summarization, 27
- topic shift, 148

- unsupervised summarization, 28
- update summarization, 27

INDEX

List of Figures

1.1	An example for a concept map.	3
2.1	A concept map that describes the idea of concept maps.	17
2.2	Subtasks of concept map mining and their dependencies.	23
3.1	Subtasks of CM-MDS illustrated by examples.	44
4.1	Summary concept map from BIOLOGY on the topic “ <i>atom</i> ”.	62
4.2	Summary concept map from WIKI on the “ <i>British contribution to the Manhattan Project</i> ”.	62
4.3	Likert-scale crowdsourcing task with topic description and two example propositions.	66
4.4	The five-step process of our scalable manual corpus creation approach. . .	69
4.5	Excerpt from a summary concept map from EDUC for the topic “ <i>students loans without credit history</i> ”.	73
5.1	Concept extraction recall for inclusive matches at increasing thresholds of k	88
5.2	PropS representation for a German sentence from TIGER.	95
5.3	Extraction precision of PropsDE at increasing yield by genre.	101
6.1	Partitioning example with six mentions and coreference predictions. . . .	108
6.2	Excerpt from the summary concept map created with the improved pipeline for the topic “ <i>students loans without credit history</i> ”.	137
7.1	Conceptual illustration of our memory-based graph representation.	152
7.2	Sequence-to-graph network unrolled for a small example.	154
7.3	Memory addressing vectors computed by our sequence-to-graph model. . .	161
7.4	Summary concept maps predicted for the test topic “ <i>students loans without credit history</i> ”.	164

LIST OF FIGURES

7.5	Summary concept maps predicted for the test topic “ <i>parents dealing with their kids being cyber-bullied</i> ”.	165
8.1	Prototype of the concept map–based document exploration system.	170
8.2	Example for a user interaction log of the exploration system.	172
9.1	Summary concept map automatically created for the content of this thesis.	178

List of Tables

2.1	Common text representations compared by user requirements.	15
4.1	Datasets with reference annotations for concept map mining.	60
4.2	Corpus statistics for automatically created benchmark corpora.	64
4.3	Correlation of manual responsiveness scores with peer summary scores.	68
4.4	Source documents of EDUC in comparison to classic MDS datasets.	72
4.5	Part-of-speech distribution in concept and relation labels of EDUC.	74
4.6	Performance of the baseline on the EDUC test set.	75
4.7	Corpus statistics for all benchmark corpora used in the thesis.	77
5.1	Concept extraction performance by dataset.	87
5.2	Relation extraction performance by dataset.	90
5.3	Concept selection performance by dataset.	91
5.4	Analysis of the portability of PropS rules from English to German.	98
5.5	Tuple extraction performance of PropsDE by text genre.	100
6.1	Complexity comparison of mention partitioning algorithms.	113
6.2	Classification performance and time for pairwise classifications.	114
6.3	Size of the partitioning problem on the smallest and biggest document sets of the training part of EDUC.	115
6.4	Runtime and optimization results for mention partitioning on the smallest and biggest document sets of the training part of EDUC.	116
6.5	Pearson correlation between features and true importance scores.	124
6.6	Concept selection performance with different models.	125
6.7	Evaluation of summary concept maps obtained with the proposed ILP.	129
6.8	Comparison of ILP sizes and runtimes for subgraph selection on EDUC.	130
6.9	End-to-end results on EDUC for our pipeline and several baselines.	134
6.10	End-to-end results on WIKI for our pipeline and several baselines.	135

LIST OF TABLES

6.11	Human preference judgments between concept maps generated on EDUC.	136
6.12	Average number of concepts and recall per topic at different pipeline steps.	138
7.1	Comparison of EDUC with the two synthetic training datasets.	144
7.2	Sequence transduction performance by training data and linearizations.	148
7.3	Sequence transduction performance by embedding learning strategy.	149
7.4	End-to-end results on EDUC for pipeline and neural approaches.	163

LIST OF TABLES

Bibliography

- Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. arXiv preprint 1603.04467v2. 147
- Adler, M., Berant, J., and Dagan, I. (2012). Entailment-based Text Exploration with Application to the Health-care Domain. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 79–84, Jeju Island, Republic of Korea. 34
- Aggarwal, C. C. and Zhai, C. (2012). A Survey of Text Clustering Algorithms. In Aggarwal, C. C. and Zhai, C., editors, *Mining Text Data*, pages 77–128. Springer, Boston, MA, USA. 34
- Aguiar, C. Z., Cury, D., and Zouaq, A. (2016). Automatic Construction of Concept Maps from Texts. In *Proceedings of the 7th International Conference on Concept Mapping*, pages 20–30, Tallinn, Estonia. 22, 23, 24, 26, 40, 54, 60, 104, 105, 106
- Akbik, A. and Lösser, A. (2012). KrakeN: N-ary Facts in Open Information Extraction. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction & Web-scale Knowledge Extraction*, pages 52–56, Montreal, Canada. 32
- Al-Sabahi, K., Zuping, Z., and Nadher, M. (2018). A Hierarchical Structured Self-Attentive Model for Extractive Document Summarization (HSSAS). *IEEE Access*, 6:24205–24212. 29, 76
- Álvarez-Miranda, E., Ljubić, I., and Mutzel, P. (2013). The Maximum Weight Connected Subgraph Problem. In Jünger, M. and Reinelt, G., editors, *Facets of Combinatorial Optimization*, pages 245–270. Springer, Berlin, Heidelberg. 48

- Angeli, G., Premkumar, M. J., and Manning, C. D. (2015). Leveraging Linguistic Structure For Open Domain Information. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 344–354, Beijing, China. 33
- Athukorala, K., Głowacka, D., Jacucci, G., Oulasvirta, A., and Vreeken, J. (2016). Is Exploratory Search Different? A Comparison of Information Search Behavior for Exploratory and Lookup Tasks. *Journal of the Association for Information Science and Technology*, 67(11):2635–2651. 10
- Ausubel, D. P. (1968). *Educational Psychology: A Cognitive View*. Holt, Rinehart & Winston, New York, NY, USA. 16
- Backes, C., Rurainski, A., Klau, G. W., Müller, O., Stöckel, D., Gerasch, A., Küntzer, J., Maisel, D., Ludwig, N., Hein, M., Keller, A., Burtscher, H., Kaufmann, M., Meese, E., and Lenhof, H.-P. (2012). An Integer Linear Programming Approach for Finding Deregulated Subgraphs in Regulatory Networks. *Nucleic Acids Research*, 40(6):e43. 49
- Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, CA, USA. 145, 150
- Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., and Etzioni, O. (2007). Open Information Extraction from the Web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2670–2676, Hyderabad, India. 31, 32
- Barzilay, R. and Lapata, M. (2006). Aggregation via Set Partitioning for Natural Language Generation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 359–366, New York, NY, USA. 109
- Barzilay, R. and McKeown, K. R. (2005). Sentence Fusion for Multidocument News Summarization. *Computational Linguistics*, 31(3):297–328. 30
- Bassa, A., Kroll, M., and Kern, R. (2018). GerIE - An Open Information Extraction System for the German Language. *Journal of Universal Computer Science*, 24(1):2–24. 93, 101
- Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G., Vaswani, A., Allen, K., Nash, C., Langston, V., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., and Pascanu, R. (2018). Relational Inductive Biases, Deep Learning, and Graph Networks. arXiv preprint 1806.01261v2. 11, 167, 179

- Belz, A. and Kow, E. (2010). Comparing Rating Scales and Preference Judgements in Language Evaluation. In *Proceedings of the 6th International Natural Language Generation Conference*, pages 7–16, Trim, Ireland. 66
- Benikova, D., Fahrner, U., Gabriel, A., Kaufmann, M., , S. M., von Landesberger, T., and Biemann, C. (2014). Network of the Day: Aggregating and Visualizing Entity Networks from Online Sources. In *Workshop Proceedings of the 12th edition of the KONVENS Conference*, pages 48–52, Hildesheim, Germany. 35
- Berg-Kirkpatrick, T., Gillick, D., and Klein, D. (2011). Jointly Learning to Extract and Compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 481–490, Portland, OR, USA. 28, 30, 120
- Berkovsky, S., Baldwin, T., and Zukerman, I. (2008). Aspect-Based Personalized Text Summarization. In *Adaptive Hypermedia and Adaptive Web-Based Systems*, volume 5149 of *Lecture Notes in Computer Science*, Hannover, Germany. 180
- Bhutani, N., Jagadish, H. V., and Radev, D. (2016). Nested Propositions in Open Information Extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 55–64, Austin, TX, USA. 32, 33
- Biemann, C. (2006). Chinese Whispers - An Efficient Graph Clustering Algorithm and its Application to Natural Language Processing Problems. In *Proceedings of TextGraphs: The First Workshop on Graph Based Methods for Natural Language Processing*, pages 73–80, New York, NY, USA. 139
- Björkelund, A., Hafdell, L., and Nugues, P. (2009). Multilingual Semantic Role Labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 43–48, Boulder, CO, USA. 82, 92
- Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022. 34
- Bohnet, B., Nivre, J., Boguslavsky, I., Farkas, R., Ginter, F., and Hajič, J. (2013). Joint Morphological and Syntactic Analysis for Richly Inflected Languages. *Transactions of the Association for Computational Linguistics*, 1(0):415–428. 98
- Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2009). Freebase. In *Compilation Proceedings of the International Conference on Management Data & 27th Symposium on Principles of Database Systems*, pages 1247–1250, Vancouver, Canada. 52

- Boudin, F., Mougard, H., and Favre, B. (2015). Concept-based Summarization using Integer Linear Programming: From Concept Pruning to Multiple Optimal Solutions. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1918, Lisbon, Portugal. 28, 50
- Boyd-Graber, J., Hu, Y., and Mimno, D. (2017). Applications of Topic Models. *Foundations and Trends in Information Retrieval*, 11(2-3):143–296. 14
- Branavan, S., Deshpande, P., and Barzilay, R. (2007). Generating a Table-of-Contents. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 544–551, Prague, Czech Republic. 34
- Brants, S., Dipper, S., Eisenberg, P., Hansen-Schirra, S., König, E., Lezius, W., Rohrer, C., Smith, G., and Uszkoreit, H. (2004). TIGER: Linguistic Interpretation of a German Corpus. *Research on Language and Computation*, 2(4):597–620. 99
- Breitman, K. K., Casanova, M. A., and Truszkowski, W. (2007). *Semantic Web: Concepts, Technologies and Applications*. Springer, London, UK. 16, 52
- Briggs, G., Shamma, D. A., Cañas, A. J., Carff, R., Scargle, J., and Novak, J. D. (2004). Concept Maps Applied to Mars Exploration Public Outreach. In *Concept Maps: Theory, Methodology, Technology. Proceedings of the First International Conference on Concept Mapping*, pages 109–116, Pamplona, Spain. 20
- Brysbaert, M., Warriner, A. B., and Kuperman, V. (2014). Concreteness Ratings for 40 Thousand Generally Known English Word Lemmas. *Behavior Research Methods*, 46(3):904–911. 122
- Burchardt, A., Erk, K., Frank, A., Kowalski, A., Pado, S., and Pinkal, M. (2006). The SALSA Corpus: A German Corpus Resource for Lexical Semantics. In *Proceedings of LREC 2006*, pages 969–974, Genoa, Italy. 92
- Buzan, T. (1984). *Make the Most of Your Mind*. Simon and Schuster, New York, London, Toronto, Sydney. 15
- Buzan, T. (2002). *How to Mind Map*. Thorsons, London, UK. 15
- Cañas, A. J., Carff, R., Hill, G., Carvalho, M., Arguedas, M., Eskridge, T. C., Lott, J., and Carvajal, R. (2005). Concept Maps: Integrating Knowledge and Information Visualization. In Tergan, S.-O. and Keller, T., editors, *Knowledge and Information Visualization*, volume 3426 of *Lecture Notes in Computer Science*, pages 205–219. Springer Berlin Heidelberg. 17, 18

- Cañas, A. J., Carvalho, M., Arguedas, M., Leake, D. B., Maguitman, A., and Reichherzer, T. (2004). Mining the Web to Suggest Concepts During Concept Map Construction. In *Concept Maps: Theory, Methodology, Technology. Proceedings of the First International Conference on Concept Mapping*, pages 135–142, Pamplona, Spain. 20
- Cañas, A. J. and Leake, D. B. (2001). Combining Concept Mapping with CBR: Towards Experience-Based Support for Knowledge Modeling. In *Proceedings of the Fourteenth International Florida Artificial Intelligence Research Society Conference*, pages 286–290, Key West, FL, USA. 121
- Cao, Z., Li, W., Li, S., Wei, F., and Li, Y. (2016). AttSum: Joint Learning of Focusing and Summarization with Neural Attention. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 547–556, Osaka, Japan. 29
- Cao, Z., Wei, F., Dong, L., Li, S., and Zhou, M. (2015). Ranking with Recursive Neural Networks and Its Application to Multi-document Summarization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2153–2159, Austin, TX, USA. 29, 117
- Carbonell, J. and Goldstein, J. (1998). The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–336, Melbourne, Australia. 29
- Carnot, M. J., Dunn, B., and Cañas, A. J. (2001). Concept Map-Based versus Web Page-Based Interfaces in Search and Browsing. In *Proceedings of the 19th International Conference on Technology and Education*, Tallahassee, FL, USA. 19, 173
- Carvalho, M., Hewett, R., and Cañas, A. J. (2001). Enhancing Web Searches from Concept Map-based Knowledge Models. In *Proceedings of the 5th World Multi-Conference on Systems, Cybernetics and Informatics*, pages 69–73, Orlando, FL, USA. 20
- Celikyilmaz, A., Bosselut, A., He, X., and Choi, Y. (2018). Deep Communicating Agents for Abstractive Summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1662–1675, New Orleans, LA, USA. 31, 54, 57, 76, 167
- Cetto, M., Niklaus, C., Freitas, A., and Handschuh, S. (2018). Graphene: Semantically-Linked Propositions in Open Information Extraction. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2300–2311, Santa Fe, NM, USA. 32
- Chali, Y., Tanvee, M., and Nayeem, M. T. (2017). Towards Abstractive Multi-Document Summarization Using Submodular Function-Based Framework, Sentence Compression

- and Merging. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing*, pages 418–424, Taipei, Taiwan. 30
- Chaney, A. J.-B. and Blei, D. M. (2012). Visualizing Topic Models. In *Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media*, pages 419–422, Dublin, Ireland. 34
- Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27. 123
- Chang, K.-W., Chang, M.-W., Srikumar, V., and Rush, A. M., editors (2017). *Proceedings of the 2nd Workshop on Structured Prediction for Natural Language Processing*. 118
- Charikar, M. S. (2002). Similarity Estimation Techniques From Rounding Algorithms. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, pages 380–388, Montréal, Canada. 139
- Chen, N.-S., Kinshuk, Wei, C.-W., and Chen, H.-J. (2006). Mining e-Learning Domain Concept Map from Academic Articles. In *Sixth IEEE International Conference on Advanced Learning Technologies*, pages 694–698, Kerkrade, Netherlands. 24
- Chen, S.-M. and Bai, S.-M. (2010). Using Data Mining Techniques to Automatically Construct Concept Maps for Adaptive Learning Systems. *Expert Systems with Applications*, 37(6):4496–4503. 24
- Chen, X., Bennett, P. N., Collins-Thompson, K., and Horvitz, E. (2013). Pairwise Ranking Aggregation in a Crowdsourced Setting. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, pages 193–202, Rome, Italy. 66
- Cheng, J. and Lapata, M. (2016). Neural Summarization by Extracting Sentences and Words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 484–494, Berlin, Germany. 28, 118
- Chin, G., Kuchar, O. A., and Wolf, K. E. (2009). Exploring the Analytical Processes of Intelligence Analysts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 11–20, Boston, MA, USA. 10, 12, 13
- Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734, Doha, Qatar. 31, 141, 145, 147, 150, 153

- Chopra, S., Auli, M., and Rush, M. A. (2016). Abstractive Sentence Summarization with Attentive Recurrent Neural Networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98, San Diego, CA, USA. 31
- Chou, J.-K. and Yang, C.-K. (2011). PaperVis: Literature Review Made Easy. In *Proceedings of the 13th Eurographics / IEEE Symposium on Visualization*, pages 721–730, Llandudno, UK. 34
- Christensen, J., Soderland, S., Bansal, G., and Mausam (2014). Hierarchical Summarization: Scaling Up Multi-Document Summarization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 902–912, Baltimore, MD, USA. 50
- Chrupala, G. and van Genabith, J. (2007). Using Very Large Corpora to Detect Raising and Control Verbs. In *Proceedings of the Lexical Functional Grammar 2007 Conference*, pages 148–162, Stanford, CA, USA. 97
- Clarke, J. and Lapata, M. (2007). Modelling Compression with Discourse Constraints. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1–11, Prague, Czech Republic. 30
- Cohan, A., Deroncourt, F., Kim, D. S., Bui, T., Kim, S., Chang, W., and Goharian, N. (2018). A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 615–621, New Orleans, LA, USA. 31, 167
- Collins, M. (2002). Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, Philadelphia, PN, USA. 123
- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.*, 12:2493–2537. 141
- Coltheart, M. (1981). The MRC Psycholinguistic Database. *The Quarterly Journal of Experimental Psychology Section A*, 33(4):497–505. 122
- Conrad, J., Gomes, C. P., van Hoes, W.-J., Sabharwal, A., and Suter, J. (2007). Connections in Networks: Hardness of Feasibility Versus Optimality. In van Hentenryck, P. and Wolsey, L., editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 4510 of *Lecture Notes in Computer Science*, pages 16–28. Springer, Berlin, Heidelberg. 49

- Conroy, J. M. and O’Leary, D. P. (2001). Text Summarization via Hidden Markov Models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 406–407, New Orleans, LA, USA. 28
- Corley, C. and Mihalcea, R. (2005). Measuring the Semantic Similarity of Texts. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 13–18, Ann Arbor, MI, USA. 106, 115
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. MIT Press, Cambridge, MA, USA, 3rd edition. 110
- Crestan, E. and de Louty, C. (2004). Browsing Help for Faster Document Retrieval. In *Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, Switzerland. 33
- Cutting, D. R., Karger, D. R., Pedersen, J. O., and Tukey, J. W. (1992). Scatter/Gather: A Cluster-Based Approach to Browsing Large Document Collections. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 318–329, Copenhagen, Denmark. 16, 34
- Dang, H. T. (2005). Overview of DUC 2005. In *Proceedings of the Document Understanding Workshop 2005*, pages 1–12, Vancouver, Canada. 27, 54, 57
- Dang, H. T. and Owczarzak, K. (2008). Overview of the TAC 2008 Update Summarization Task. In *Proceedings of the First Text Analysis Conference*, pages 1–16, Gaithersburg, MD, USA. 27, 67
- Daumé III, H. (2006). *Practical Structured Learning Techniques for Natural Language Processing*. Ph.D. Thesis, University of Southern California, Los Angeles, CA, USA. 118
- de Belder, J. and Moens, M.-F. (2012). Coreference Clustering using Column Generation. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 245–254, Mumbai, India. 110, 111
- de la Chica, S., Ahmad, F., Martin, J. H., and Sumner, T. (2008). Pedagogically Useful Extractive Summaries for Science Education. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 177–184, Manchester, UK. 23
- de Marneffe, M.-C. and Manning, C. D. (2008). The Stanford Typed Dependencies Representation. In *Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8, Manchester, UK. 24, 84, 93, 94

- Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407. 107
- Del Corro, L. and Gemulla, R. (2013). ClausIE: Clause-Based Open Information Extraction. In *Proceedings of the 22nd International Conference on the World Wide Web*, pages 355–366, Rio de Janeiro, Brazil. 32, 99
- Denis, P. and Baldridge, J. (2007). Joint Determination of Anaphoricity and Coreference Resolution using Integer Programming. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 236–243, Rochester, NY, USA. 109
- Denkowski, M. and Lavie, A. (2014). Meteor Universal: Language Specific Translation Evaluation for Any Target Language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, MD, USA. 55
- Desrosiers, J. and Lübbecke, M. E. (2005). A Primer in Column Generation. In Desaulniers, G., Desrosiers, J., and Solomon, M. M., editors, *Column Generation*, pages 1–32. Springer Science+Business Media Inc, Boston, MA, USA. 110
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint 1801.04805v1. 167, 179
- Di Battista, G., Eades, P., Tamassia, R., and Tollis, I. G. (1998). *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, Upper Saddle River, NJ, USA. 43
- Doddington, G., Mitchell, A., Przybocki, M., Ramshaw, L., Strassel, S., and Weischedel, R. (2004). The Automatic Content Extraction (ACE) Program – Tasks, Data, and Evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*, pages 837–840, Lisbon, Portugal. 31
- Dohare, S., Gupta, V., and Karnick, H. (2018). Unsupervised Semantic Abstractive Summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 74–83, Melbourne, Australia. 179
- Dong, L. and Lapata, M. (2016). Language to Logical Form with Neural Attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 33–43, Berlin, Germany. 145
- Dou, W., Wang, X., Chang, R., and Ribarsky, W. (2011). ParallelTopics: A Probabilistic Approach to Exploring Document Collections. In *IEEE Conference on Visual Analytics Science and Technology*, pages 231–240, Providence, RI, USA. 35

- Dror, R., Baumer, G., Shlomov, S., and Reichart, R. (2018). The Hitchhiker’s Guide to Testing Statistical Significance in Natural Language Processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 1383–1392, Melbourne, Australia. 56
- Eckart de Castilho, R. and Gurevych, I. (2014). A Broad-Coverage Collection of Portable NLP Components for Building Shareable Analysis Pipelines. In *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT*, pages 1–11, Dublin, Ireland. 84
- Edmundson, H. P. (1969). New Methods in Automatic Extracting. *Journal of the ACM (JACM)*, 16(2):264–285. 28, 119
- Edwards, J. and Fraser, K. (1983). Concept Maps as Reflectors of Conceptual Understanding. *Research in Science Education*, 13(1):19–26. 19, 21
- Ehrlinger, L. and Wöß, W. (2016). Towards a Definition of Knowledge Graphs. In *Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems SEMANTiCS2016 and the 1st International Workshop on Semantic Change & Evolving Semantics*, Leipzig, Germany. 52
- Eisenstein, J. (2009). Hierarchical Text Segmentation from Multi-Scale Lexical Cohesion. In *Proceedings of the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 353–361, Boulder, CO, USA. 34
- El-Kebir, M. and Klau, G. W. (2014). Solving the Maximum-Weight Connected Subgraph Problem to Optimality. arXiv preprint 1409.5308v2. 48
- Erbs, N. (2015). *Approaches to Automatic Text Structuring*. Ph.D. Thesis, Technische Universität Darmstadt. 34
- Erbs, N., Gurevych, I., and Zesch, T. (2013). Hierarchy Identification for Automatically Generating Table-of-Contents. In *Proceedings of 9th Conference on Recent Advances in Natural Language Processing*, pages 252–260, Hissar, Bulgaria. 34
- Erkan, G. and Radev, D. R. (2004). LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. *Journal of Artificial Intelligence Research*, 22(1):457–479. 28, 121, 146
- Fader, A., Soderland, S., and Etzioni, O. (2011). Identifying Relations for Open Information Extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, Edinburgh, UK. 32, 99

- Falke, T. and Gurevych, I. (2017a). Bringing Structure into Summaries: Crowdsourcing a Benchmark Corpus of Concept Maps. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2951–2961, Copenhagen, Denmark. 5
- Falke, T. and Gurevych, I. (2017b). GraphDocExplore: A Framework for the Experimental Comparison of Graph-based Document Exploration Techniques. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 19–24, Copenhagen, Denmark. 5
- Falke, T. and Gurevych, I. (2017c). Utilizing Automatic Predicate-Argument Analysis for Concept Map Mining. In *Proceedings of the 12th International Conference on Computational Semantics*, Montpellier, France. 5, 64
- Falke, T., Meyer, C. M., and Gurevych, I. (2017). Concept-Map-Based Multi-Document Summarization using Concept Coreference Resolution and Global Importance Optimization. In *Proceedings of the 8th International Joint Conference on Natural Language Processing*, pages 801–811, Taipei, Taiwan. 5
- Falke, T., Stanovsky, G., Gurevych, I., and Dagan, I. (2016). Porting an Open Information Extraction System from English to German. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 892–898, Austin, TX, USA. 5
- Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9:1871–1874. 122
- Fang, Y. and Teufel, S. (2016). Improving Argument Overlap for Proposition-Based Summarisation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 479–485, Berlin, Germany. 31, 159
- Fang, Y., Zhu, H., Muszyńska, E., Kuhnle, A., and Teufel, S. (2016). A Proposition-Based Abstractive Summariser. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 567–578, Osaka, Japan. 31, 151, 159
- Filippova, K. and Strube, M. (2008). Sentence Fusion via Dependency Graph Compression. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 177–185, Honolulu, HI, USA. 30
- Fort, K., Adda, G., and Cohen, K. B. (2011). Amazon Mechanical Turk: Gold Mine or Coal Mine? *Computational Linguistics*, 37(2):413–420. 65
- Gaines, B. R. and Shaw, M. L. G. (1994). Using Knowledge Acquisition and Representation Tools to Support Scientific Communities. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 707–714, Seattle, WA, USA. 20

- Galler, B. A. and Fisher, M. J. (1964). An Improved Equivalence Algorithm. *Communications of the ACM*, 7(5):301–303. 109
- Gamallo, P. and Garcia, M. (2015). Multilingual Open Information Extraction. In *Proceedings of the 17th Portuguese Conference on Artificial Intelligence*, volume 9273 of *Lecture Notes in Computer Science*, pages 711–722, Coimbra, Portugal. 33, 93
- Gambhir, M. and Gupta, V. (2017). Recent Automatic Text Summarization Techniques: A Survey. *Artificial Intelligence Review*, 47(1):1–66. 28
- Gillick, D. and Favre, B. (2009). A Scalable Global Model for Summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 10–18, Boulder, CO, USA. 30, 50, 51
- Goldberg, Y. (2017). Neural Network Methods for Natural Language Processing. *Synthesis Lectures on Human Language Technologies*, 10(1):1–309. 29, 141, 145, 157
- Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. 141
- Görg, C., Liu, Z., Kihm, J., Choo, J., Park, H., and Stasko, J. T. (2013). Combining Computational Analyses and Interactive Visualization for Document Exploration and Sensemaking in Jigsaw. *IEEE Transactions on Visualization and Computer Graphics*, 19(10):1646–1663. 35
- Graves, A. (2013). Generating Sequences With Recurrent Neural Networks. arXiv preprint 1308.0850v5. 150
- Graves, A., Wayne, G., and Danihelka, I. (2014). Neural Turing Machines. arXiv preprint 1410.5401v2. 150, 154, 155
- Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., Colmenarejo, S. G., Grefenstette, E., Ramalho, T., Agapiou, J., Badia, A. P., Hermann, K. M., Zwols, Y., Ostrovski, G., Cain, A., King, H., Summerfield, C., Blunsom, P., Kavukcuoglu, K., and Hassabis, D. (2016). Hybrid Computing Using a Neural Network with Dynamic External Memory. *Nature*, 538(7626):471–476. 150
- Griffiths, T. L., Jordan, M. I., Tenenbaum, J. B., and Blei, D. M. (2004). Hierarchical Topic Models and the Nested Chinese Restaurant Process. In *Advances in Neural Information Processing Systems 16*, pages 17–24. 35
- Grusky, M., Naaman, M., and Artzi, Y. (2018). Newsroom: A Dataset of 1.3 Million Summaries with Diverse Extractive Strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 708–719, New Orleans, LA, USA. 143

- Gu, J., Lu, Z., Li, H., and Li, V. O. (2016). Incorporating Copying Mechanism in Sequence-to-Sequence Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1631–1640, Berlin, Germany. 31
- Gulcehre, C., Chandar, S., and Bengio, Y. (2017). Memory Augmented Neural Networks with Wormhole Connections. arXiv preprint 1701.08718v1. 150
- Gulcehre, C., Chandar, S., Cho, K., and Bengio, Y. (2018). Dynamic Neural Turing Machine with Continuous and Discrete Addressing Schemes. *Neural Computation*, 30(4):857–884. 150, 154, 155
- Gutwin, C., Paynter, G. W., Witten, I. H., Nevill-Manning, C. G., and Frank, E. (1999). Improving Browsing in Digital Libraries with Keyphrase Indexes. *Decision Support Systems*, 27(1-2):81–104. 14
- Habernal, I., Sukhareva, M., Raiber, F., Shtok, A., Kurland, O., Ronen, H., Bar-Ilan, J., and Gurevych, I. (2016a). New Collection Announcement: Focused Retrieval Over the Web. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '16*, pages 701–704, Pisa, Italy. 68
- Habernal, I., Zayed, O., and Gurevych, I. (2016b). C4Corpus: Multilingual Web-size Corpus with Free License. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, pages 914–922, Portorož, Slovenia. 99
- Haghighi, A. and Vanderwende, L. (2009). Exploring Content Models for Multi-Document Summarization. In *Proceedings of the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370, Boulder, CO, USA. 50
- Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., Straňák, P., Surdeanu, M., Xue, N., and Zhang, Y. (2009). The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 1–18, Boulder, CO, USA. 81, 82, 92
- Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1):10–18. 75, 114, 122
- Hasan, K. S. and Ng, V. (2014). Automatic Keyphrase Extraction: A Survey of the State of the Art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1262–1273, Baltimore, MD, USA. 14, 34

- Hatzivassiloglou, V., Klavans, J. L., Holcombe, M. L., Barzilay, R., Kan, M.-Y., and McKeown, K. (2001). Simfinder: A Flexible Clustering Tool for Summarization. In *Proceedings of the NAACL Workshop on Automatic Summarization*, Pittsburg, PA, USA. 29
- Hearst, M. A. (1992). Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of the 15th International Conference on Computational Linguistics*, pages 539–545, Nantes, France. 25
- Hearst, M. A. and Stoica, E. (2009). NLP Support for Faceted Navigation in Scholarly Collections. In *Proceedings of the 2009 Workshop on Text and Citation Analysis for Scholarly Digital Libraries*, pages 62–70, Singapore. 33, 171
- Herbrich, R., Minka, T., and Graepel, T. (2007). TrueSkill(TM): A Bayesian Skill Rating System. In *Advances in Neural Information Processing Systems 19*, pages 569–576. 67
- Herman, I., Melancon, G., and Marshall, M. S. (2000). Graph Visualization and Navigation in Information Visualization: A Survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):24–43. 43
- Hermann, K. M., Kočiský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching Machines to Read and Comprehend. In *Advances in Neural Information Processing Systems 28 (NIPS 2015)*, pages 1693–1701, Montréal, Canada. 143, 144
- Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780. 145
- Hoffman, R. R., Coffey, J. W., Ford, K. M., and Carnot, M. J. (2001). STORM-LK: A Human-Centered Knowledge Model for Weather Forecasting. In *Proceedings of the Human Factors and Ergonomics Society 45th Annual Meeting*, page 752, Minneapolis, MN, USA. 20
- Hoffman, R. R., Coffey, J. W., and Novak, J. D. (2005). Applications of Concept Maps to Web Design and Web Work. In Proctor, R. W. and Vu, K.-P. L., editors, *Handbook of Human Factors in Web Design*, pages 156–175. L. Erlbaum Associates, Mahwah, NJ, USA. 18
- Hong, K. and Nenkova, A. (2014). Improving the Estimation of Word Importance for News Multi-Document Summarization. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 712–721, Gothenburg, Sweden. 28, 117, 120
- Howard, J. and Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 328–339, Melbourne, Australia. 167, 179

- Isaacson, W. (2014). *The Innovators: How a Group of Hackers, Geniuses, and Geeks Created the Digital Revolution*. Simon & Schuster, New York. 1
- Jackson, A., Lin, J., Milligan, I., and Ruest, N. (2016). Desiderata for Exploratory Search Interfaces to Web Archives in Support of Scholarly Activities. In *Proceedings of the 16th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 103–106, Newark, NJ, USA. 10
- Jiang, J. J. and Conrath, D. W. (1997). Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In *Proceedings of the 10th Research on Computational Linguistics International Conference*, pages 19–33, Taipei, Taiwan. 106
- Jing, H. (2000). Sentence Reduction for Automatic Text Summarization. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 310–315, Seattle, WA, USA. 30
- Joachims, T. (2002). Optimizing Search Engines Using Clickthrough Data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142, Edmonton, Canada. 118
- Johnson, D. D. (2017). Learning Graphical State Transitions. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France. 150, 151, 153
- Jurafsky, M. and Martin, J. H. (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition. 26, 31, 105
- Kalchbrenner, N. and Blunsom, P. (2013). Recurrent Continuous Translation Models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, WA, USA. 150
- Kang, H., Plaisant, C., Lee, B., and Bederson, B. B. (2006). NetLens: Iterative Exploration of Content-Actor Network Data. In *2006 IEEE Symposium On Visual Analytics And Technology*, pages 91–98, Baltimore, MD, USA. 36
- Kang, Y.-A., Görg, C., and Stasko, J. T. (2011). How Can Visual Analytics Assist Investigative Analysis? Design Implications from an Evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 17(5):570–583. 12, 13, 35
- Kang, Y.-B., Delir Haghghi, P., and Burstein, F. (2016). TaxoFinder: A Graph-based Approach for Taxonomy Learning. *IEEE Transactions on Knowledge and Data Engineering*, 28(2):524–536. 34
- Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C., and Giannopoulou, E. (2007). Ontology Visualization Methods - A Survey. *ACM Computing Surveys (CSUR)*, 39(4):10. 43

- Keim, D. A., Andrienko, G., Fekete, J.-D., Görg, C., Kohlhammer, J., and Melançon, G. (2008). Visual Analytics: Definition, Process, and Challenges. In Kerren, A. et al., editors, *Information Visualization*, volume 4950 of *Lecture Notes in Computer Science*, pages 154–175. Springer Berlin Heidelberg. 1
- Kellerer, H., Pferschy, U., and Pisinger, D. (2010). *Knapsack Problems*. Springer, Berlin. 29
- Kim, S., Yeganova, L., and Wilbur, W. J. (2015). Summarizing Topical Contents from PubMed Documents Using a Thematic Analysis. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 805–810, Lisbon, Portugal. 34
- Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751, Doha, Qatar. 141
- King, D. E. (2009). Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research*, 10:1755–1758. 123
- Kingma, D. and Ba, J. L. (2015). Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, CA, USA. 147, 157, 160, 162
- Kintsch, W. and van Dijk, T. A. (1978). Toward a Model of Text Comprehension and Production. *Psychological Review*, 85(5):363–394. 151, 159
- Kiritchenko, S. and Mohammed, S. M. (2016). Capturing Reliable Fine-Grained Sentiment Associations by Crowdsourcing and Best-Worst Scaling. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 811–817, San Diego, CA, USA. 67
- Kirkpatrick, K. (2015). Putting the Data Science into Journalism. *Communications of the ACM*, 58(5):15–17. 10
- Klein, M. and Nelson, M. L. (2009). Correlation of Term Count and Document Frequency for Google N-Grams. In Hutchison, D. and et. al., editors, *Advances in Information Retrieval*, volume 5478 of *Lecture Notes in Computer Science*, pages 620–627. Springer Berlin Heidelberg, Berlin, Heidelberg. 119
- Kleinberg, J. M. (1999). Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM (JACM)*, 46(5):604–632. 121
- Knight, K. and Marcu, D. (2002). Summarization Beyond Sentence Extraction: A Probabilistic Approach to Sentence Compression. *Artificial Intelligence*, 139(1):91–107. 30

- Kof, L., Gacitua, R., Rouncefield, M., and Sawyer, P. (2010). Concept Mapping as a Means of Requirements Tracing. In *Third International Workshop on Managing Requirements Knowledge*, pages 22–31, Sydney, Australia. 24
- Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J., Paatero, V., and Saarela, A. (2000). Self Organization of a Massive Document Collection. *IEEE Transactions on Neural Networks*, 11(3):574–585. 34
- Konstas, I., Iyer, S., Yatskar, M., Choi, Y., and Zettlemoyer, L. (2017). Neural AMR: Sequence-to-Sequence Models for Parsing and Generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 146–157, Vancouver, Canada. 144, 145
- Koschützki, D., Lehmann, K. A., Peeters, L., Richter, S., Tenfelde-Podehl, D., and Zlotowski, O. (2005). Centrality Indices. In Brandes, U. and Erlebach, T., editors, *Network Analysis*, volume 3418 of *Lecture Notes in Computer Science*, pages 16–61. Springer, Berlin, Heidelberg. 121
- Kowata, J. H., Cury, D., and Silva Boeres, M. C. (2010). Concept Maps Core Elements Candidates Recognition from Text. In *Concept Maps: Making Learning Meaningful. Proceedings of the 4th International Conference on Concept Mapping*, pages 120–127, Vina del Mar, Chile. 22, 23, 24, 25, 40, 54, 91
- Kozareva, Z. and Hovy, E. (2010). A Semi-Supervised Method to Learn and Construct Taxonomies Using the Web. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1110–1118, Cambridge, MA, USA. 34
- Kübler, S. (2008). The PaGe 2008 Shared Task on Parsing German. In *Proceedings of the ACL-08: HLT Workshop on Parsing German (PaGe-08)*, pages 55–63, Columbus, OH, USA. 94
- Kummamuru, K., Lotlikar, R., Roy, S., Singal, K., and Krishnapuram, R. (2004). A Hierarchical Monothetic Document Clustering Algorithm for Summarization and Browsing Search Results. In *Proceedings of the 13th International Conference on World Wide Web*, pages 658–665, New York, NY, USA. 34
- Kupiec, J., Pedersen, J., and Chen, F. (1995). A Trainable Document Summarizer. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 68–73, Seattle, WA, USA. 28
- Landis, J. R. and Koch, G. G. (1977). The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174. 72

- Langer, H., Lungen, H., and Bayerl, P. S. (2004). Text Type Structure and Logical Document Structure. In *Proceedings of the 2004 ACL Workshop on Discourse Annotation*, pages 49–56, Barcelona, Spain. 34
- Lawrie, D., Croft, W. B., and Rosenberg, A. (2001). Finding Topic Words for Hierarchical Summarization. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 349–357, New Orleans, LA, USA. 34
- Leacock, C. and Chodorow, M. (1998). Combining local context and WordNet sense similarity for word sense disambiguation. In Fellbaum, C., editor, *WordNet: An Electronic Lexical Database*, pages 265–284. MIT Press, London, England. 106
- Leake, D. B., Maguitman, A., and Reichherzer, T. (2003). Topic Extraction and Extension to Support Concept Mapping. In *Proceedings of the Sixteenth International Florida Artificial Intelligence Research Society Conference*, pages 325–329, St. Augustine, FL, USA. 20
- Leake, D. B., Maguitman, A., Reichherzer, T., Cañas, A. J., Carvalho, M., Arguedas, M., and Eskridge, T. C. (2004). "Googling" From a Concept Map: Towards Automatic Concept-Map-Based Query Formulation. In *Concept Maps: Theory, Methodology, Technology. Proceedings of the First International Conference on Concept Mapping*, pages 409–416, Pamplona, Spain. 20, 27
- Leban, G., Fortuna, B., Brank, J., and Grobelnik, M. (2014). Event Registry: Learning About World Events from News. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 107–110, Seoul, Republic of Korea. 35
- LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep Learning. *Nature*, 521(7553):436. 141
- Lee, B., Czerwinski, M., Robertson, G., and Bederson, B. B. (2005). Understanding Research Trends in Conferences Using PaperLens. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, pages 1969–1972, Portland, OR, USA. 10, 34
- Lee, K., He, L., Lewis, M., and Zettlemoyer, L. (2017). End-to-end Neural Coreference Resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. 105
- Lee, S., Park, Y., and Yoon, W. C. (2015). Burst Analysis for Automatic Concept Map Creation with a Single Document. *Expert Systems with Applications*, 42(22):8817–8829. 24
- Lee, Y.-J. (2004). Concept Mapping Your Web Searches: A Design Rationale and Web-enabled Application. *Journal of Computer Assisted Learning*, 20(2):103–113. 20

- Lenat, D., Prakash, M., and Shepherd, M. (1986). CYC: Using Common Sense Knowledge to Overcome Brittleness and Knowledge Acquisition Bottlenecks. *AI Magazine*, 6(4):65–85. 52
- Li, B., Liu, J., Lin, C.-Y., King, I., and Lyu, M. R. (2013). A Hierarchical Entity-Based Approach to Structuralize User Generated Content in Social Media: A Case of Yahoo! Answers. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1521–1532, Seattle, WA, USA. 34, 120
- Li, S., Ouyang, Y., Wang, W., and Sun, B. (2007). Multi-Document Summarization Using Support Vector Regression. In *Proceedings of the Document Understanding Conference 2007*, Rochester, NY, USA. 28
- Li, W. (2015). Abstractive Multi-document Summarization with Semantic Information Extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1908–1913, Lisbon, Portugal. 30, 50, 120, 121
- Li, W., He, L., and Zhuge, H. (2016a). Abstractive News Summarization based on Event Semantic Link Network. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 236–246, Osaka, Japan. 28, 30, 50, 51, 118, 120, 121, 123, 127, 128, 129, 130
- Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. (2016b). Gated Graph Sequence Neural Networks. In *Proceedings of the 4th International Conference on Learning Representations*, San Juan, Puerto Rico. 150, 153
- Lin, C.-Y. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain. 53, 56
- Lin, D. (1998). An Information-Theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 296–304, Madison, WI, USA. 26, 106
- Lin, H. and Bilmes, J. (2011). A Class of Submodular Functions for Document Summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 510–520, Portland, OR, USA. 29
- Liu, F., Flanigan, J., Thomson, S., Sadeh, N., and Smith, N. A. (2015). Toward Abstractive Summarization Using Semantic Representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086, Denver, CO, USA. 31, 51, 118, 121, 127

- Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., and Shazeer, N. (2018). Generating Wikipedia by Summarizing Long Sequences. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada. 31, 146, 167
- Liu, S., Cui, W., Wu, Y., and Liu, M. (2014). A Survey on Information Visualization: Recent Advances and Challenges. *The Visual Computer*, 30(12):1373–1393. 43
- Liu, S., Zhou, M. X., Pan, S., Song, Y., Qian, W., Cai, W., and Lian, X. (2012). TIARA: Interactive, Topic-Based Visual Text Summarization and Analysis. *ACM Transactions on Intelligent Systems and Technology*, 3(2):1–28. 35
- Liu, T.-Y. (2009). Learning to Rank for Information Retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331. 47, 118
- Lloret, E., Plaza, L., and Aker, A. (2013). Analyzing the Capabilities of Crowdsourcing Services for Text Summarization. *Language Resources and Evaluation*, 47(2):337–369. 65, 67
- Loizides, F. and Buchanan, G. (2009). An Empirical Study of User Navigation during Document Triage. In Hutchison, D. et al., editors, *Research and Advanced Technology for Digital Libraries*, volume 5714 of *Lecture Notes in Computer Science*, pages 138–149. Springer Berlin Heidelberg. 12, 13
- Louis, A. (2014). A Bayesian Method to Incorporate Background Knowledge during Automatic Text Summarization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 333–338, Baltimore, MD, USA. 51, 180
- Luhn, H. P. (1958). The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development*, 2(2):159–165. 28, 119
- Luo, D., Yang, J., Krstajic, M., Ribarsky, W., and Keim, D. A. (2012). EventRiver: Visually Exploring Text Collections with Temporal References. *IEEE Transactions on Visualization and Computer Graphics*, 18(1):93–105. 34
- Luong, M.-T., Pham, H., and Manning, C. D. (2015). Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. 145, 147, 155
- Maedche, A. (2002). *Ontology Learning for the Semantic Web*, volume 665 of *The Kluwer International Series in Engineering and Computer Science*. Springer, Boston, MA, USA. 16, 52

- Magnanti, T. L. and Wolsey, L. A. (1994). Optimal Trees. Technical report, Operations Research Center, Massachusetts Institute of Technology. Available at <http://dspace.mit.edu/bitstream/1721.1/5122>. 127
- Maña-López, M. J., de Buenaga, M., and Gómez-Hidalgo, J. M. (2004). Multidocument Summarization: An Added Value to Clustering in Interactive Retrieval. *ACM Transactions on Information Systems*, 22(2):215–241. 13, 54
- Manning, C., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge Univ. Press, Cambridge, MA, USA. 33
- Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., and McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 55–60, Baltimore, MD, USA. 123, 131
- Marchionini, G. (2006). Exploratory Search. *Communications of the ACM*, 49(4):41–46. 2, 9, 10, 13, 33
- Marcus, G. (2018). Deep Learning: A Critical Appraisal. arXiv preprint 1801.00631v1. 11
- Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330. 24
- Martins, A., Smith, N. A., and Xing, E. (2009). Concise Integer Linear Programming Formulations for Dependency Parsing. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 342–350, Singapore. 127
- Mausam, Schmitz, M., Bart, R., Soderland, S., and Etzioni, O. (2012). Open Language Learning for Information Extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534, Jeju Island, Republic of Korea. 32, 33, 99
- Mausam, M. (2016). Open information extraction systems and downstream applications. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 4074–4077, New York, NY, USA. 32, 63, 69, 84, 130
- McClure, J. R., Sonak, B., and Suen, H. K. (1999). Concept Map Assessment of Classroom Learning: Reliability, Validity, and Logistical Practicality. *Journal of Research in Science Teaching*, 36(4):475–492. 19, 54

- McDonald, R. (2007). A Study of Global Inference Algorithms in Multi-Document Summarization. In *Proceedings of the 29th European conference on IR research*, pages 557–564, Rome, Italy. 29, 30
- McKeown, K., Passonneau, R. J., Elson, D. K., Nenkova, A., and Hirschberg, J. (2005). Do Summaries Help? A Task-Based Evaluation of Multi-Document Summarization. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 210–217, Salvador, Brazil. 13, 54
- Mesquita, F., Schmidek, J., and Barbosa, D. (2013). Effectiveness and Efficiency of Open Relation Extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 447–457, Seattle, WA, USA. 32
- Mihalcea, R. and Tarau, P. (2004). TextRank: Bringing Order into Texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. 28, 121
- Mikolov, T. (2012). *Statistical Language Models based on Neural Networks*. Ph.D. Thesis, Brno University of Technology. 31, 141
- Mikolov, T., Chen, K., Corrado, G. S., and Dean, J. (2013a). Efficient Estimation of Word Representations in Vector Space. arXiv preprint 1301.3781v3. 106
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed Representations of Words and Phrases and their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, pages 3111–3119, Lake Tahoe, NV, USA. 104
- Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. J. (1990). Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography*, 3(4):235–244. 25, 55, 106
- Moriceau, V. and Tannier, X. (2014). French Resources for Extraction and Normalization of Temporal Expressions with HeidelTime. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, pages 3239–3243, Reykjavik, Iceland. 92
- Nallapati, R., Zhai, F., and Zhou, B. (2017). SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 3075–3081, San Francisco, CA, USA. 29
- Nallapati, R., Zhou, B., Santos, C. d., Gulcehre, C., and Xiang, B. (2016). Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. In *Proceedings of the*

- 20th SIGNLL Conference on Computational Natural Language Learning, pages 280–290, Berlin, Germany. 31
- Napoles, C., Gormley, M., and van Durme, B. (2012). Annotated Gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction & Web-scale Knowledge Extraction*, pages 95–100, Montréal, Canada. 143
- Nenkova, A. and McKeown, K. R. (2011). Automatic Summarization. *Foundations and Trends in Information Retrieval*, 5(2):103–233. 14, 27, 28, 30
- Nenkova, A. and Passonneau, R. (2004). Evaluating Content Selection in Summarization: The Pyramid Method. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 145–152, Boston, MA, USA. 54
- Nenkova, A., Passonneau, R., and McKeown, K. (2007). The Pyramid Method: Incorporating human content selection variation in summarization evaluation. *ACM Transactions on Speech and Language Processing*, 4(2):4. 54
- Nesbit, J. C. and Adesope, O. O. (2006). Learning With Concept and Knowledge Maps: A Meta-Analysis. *Review of Educational Research*, 76(3):413–448. 18, 19
- Nguyen, V. C., Nguyen, L. M., and Shimazu, A. (2009). A Semi-supervised Approach for Generating a Table-of-Contents. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 312–317, Borovets, Bulgaria. 34
- Niklaus, C., Cetto, M., Freitas, A., and Handschuh, S. (2018). A Survey on Open Information Extraction. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3866–3878, Santa Fe, NM, USA. 33
- Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajič, J., Manning, C. D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. (2016). Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, pages 1659–1666, Portorož, Slovenia. 33, 93
- Noreen, E. W. (1989). *Computer Intensive Methods for Testing Hypotheses: An Introduction*. Wiley, New York. 56
- Novak, J. D. and Cañas, A. J. (2006). The Origins of the Concept Mapping Tool and the Continuing Evolution of the Tool. *Information Visualization*, 5(3):175–184. 18
- Novak, J. D. and Cañas, A. J. (2007). Theoretical Origins of Concept Maps, How to Construct Them, and Uses in Education. *Reflecting Education*, 3(1):29–42. 16, 17, 21, 72

BIBLIOGRAPHY

- Novak, J. D. and Cañas, A. J. (2008). The Theory Underlying Concept Maps and How to Construct and Use Them. Technical report, Florida Institute for Human and Machine Cognition. Available at <http://cmap.ihmc.us/docs/pdf/TheoryUnderlyingConceptMaps.pdf>. 16, 21, 32
- Novak, J. D. and Gowin, D. B. (1984). *Learning How to Learn*. Cambridge University Press, Cambridge, MA, USA. 2, 15, 16
- Oliveira, A., Pereira, F. C., and Cardoso, A. (2001). Automatic Reading and Learning from Text. In *Proceedings of the International Symposium on Artificial Intelligence*, Kolhapur, India. 22, 24, 25, 26, 104, 105
- Olney, A., Cade, W., and Williams, C. (2011). Generating Concept Map Exercises from Textbooks. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 111–119, Portland, OR, USA. 24, 25, 40, 61
- Over, P., Dang, H., and Harman, D. (2007). DUC in Context. *Information Processing & Management*, 43(6):1506–1520. 27
- Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The PageRank Citation Ranking: Bringing Order to the Web. Technical Report 1999-66, Stanford InfoLab. Available at <http://ilpubs.stanford.edu:8090/422/>. 28, 121
- Palmer, M., Gildea, D., and Kingsbury, P. (2005). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106. 81
- Park, S. and An, D. U. (2010). Automatic Query-Based Personalized Summarization That Uses Pseudo Relevance Feedback With NMF. In *Proceedings of the 4th International Conference on Ubiquitous Information Management and Communication*, Suwon, Republic of Korea. 180
- Parker, J., editor (2003). *XML Topic Maps: Creating and Using Topic Maps for the Web*. Addison-Wesley, Boston, MA, USA. 15
- Patterson, E. S., Roth, E. M., and Woods, D. D. (2001). Predicting Vulnerabilities in Computer-Supported Inferential Analysis under Data Overload. *Cognition, Technology & Work*, 3(4):224–237. 1, 10
- Paulus, R., Xiong, C., and Socher, R. (2018). A Deep Reinforced Model for Abstractive Summarization. In *Proceedings of the 6th International Conference on Learning Representations*, volume 1705.04304, Vancouver, Canada. 54, 149

- Pavlick, E., Rastogi, P., Ganitkevich, J., Van Durme, B., and Callison-Burch, C. (2015). Ppdb 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 425–430, Beijing, China. 55, 143
- Pearl, J. (2018). Theoretical Impediments to Machine Learning With Seven Sparks from the Causal Revolution. arXiv preprint 1801.04016v1. 11
- Pembe, F. C. and GÜngör, T. (2015). A Tree-Based Learning Approach for Document Structure Analysis and its Application to Web Search. *Natural Language Engineering*, 21(04):569–605. 34
- Pennington, J., Socher, R., and Manning, C. D. (2014). GloVe: Global Vectors for Word Representations. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, Doha, Qatar. 104, 106, 147, 162
- Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237, New Orleans, LA, USA. 167, 179
- Petrov, S., Das, D., and McDonald, R. (2012). A Universal Part-of-Speech Tagset. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 2089–2096, Istanbul, Turkey. 74
- Peyrard, M. (2018). A Formal Definition of Importance for Summarization. arXiv preprint 1801.08991v1. 51, 180
- Peyrard, M. and Eckle-Kohler, J. (2016). Optimizing an Approximation of ROUGE - a Problem-Reduction Approach to Extractive Multi-Document Summarization. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1825–1836, Berlin, Germany. 76
- Pilehvar, M. T., Jurgens, D., and Navigli, R. (2013). Align, Disambiguate and Walk: A Unified Approach for Measuring Semantic Similarity. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1341–1351, Sofia, Bulgaria. 72, 106, 114
- Pirolli, P. and Card, S. K. (2005). The Sensemaking Process and Leverage Points for Analyst Technology as Identified Through Cognitive Task Analysis. In *Proceedings of the 2005 International Conference on Intelligence Analysis*, McLean, VA, USA. 10, 11, 13

- Pradhan, S., Moschitti, A., Xue, N., Uryupina, O., and Zhang, Y. (2012). CoNLL-2012 Shared Task: Modeling Multilingual Unrestricted Coreference in OntoNotes. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1–40, Jeju Island, Korea. 105
- Pujara, J., Rocktäschel, T., Chen, D., and Singh, S., editors (2016). *Proceedings of the 5th Workshop on Automated Knowledge Base Construction*, San Diego, CA, USA. 52
- P.V.S., A. and Meyer, C. M. (2017). Joint Optimization of User-desired Content in Multi-document Summaries by Learning from User Feedback. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1353–1363, Vancouver, Canada. 51, 180
- QasemiZadeh, B. and Schumann, A.-K. (2016). The ACL RD-TEC 2.0: A Language Resource for Evaluating Term Extraction and Entity Recognition Methods. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, pages 1862–1868, Portorož, Slovenia. 63
- Qasim, I., Jeong, J.-W., Heu, J.-U., and Lee, D.-H. (2013). Concept Map Construction From Text Documents Using Affinity Propagation. *Journal of Information Science*, 39(6):719–736. 22, 23, 24, 25, 26, 27, 40, 54, 59, 60, 85, 86, 105
- Rajaraman, K. and Tan, A.-H. (2002). Knowledge Discovery from Texts: A Concept Frame Graph Approach. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, pages 669–671, McLean, VA, USA. 22, 23, 24, 25, 26, 40, 54, 104
- Ravichandran, D., Pantel, P., and Hovy, E. (2005). Randomized Algorithms and NLP: Using Locality Sensitive Hash Function for High Speed Noun Clustering. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 622–629, Ann Arbor, MI, USA. 139
- Reichherzer, T. and Leake, D. (2006). Understanding the Role of Structure in Concept Maps. In *Proceedings of the Twenty-Eighth Annual Conference of the Cognitive Science Society*, pages 2004–2009, Vancouver, Canada. 121
- Resnik, P. (1995). Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453, Montréal, Canada. 106, 115
- Richardson, R. and Fox, E. A. (2005). Using Concept Maps as a Cross-language Resource Discovery Tool for Large Documents in Digital Libraries. In *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, page 415, Denver, CO, USA. 20

- Richardson, R. and Fox, E. A. (2007). Using Concept Maps in NDLTD as a Cross-Language Summarization Tool for Computing-Related ETDs. In *Proceedings of the 10th International Symposium on Electronic Theses and Dissertations*, Uppsala, Sweden. 20
- Riezler, S. and Maxwell, J. T. (2005). On Some Pitfalls in Automatic Evaluation and Significance Testing for MT. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 57–64, Ann Arbor, MI, USA. 56
- Roth, D. (2017). Incidental Supervision: Moving beyond Supervised Learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 4885–4890, San Francisco, CA, USA. 167, 179
- Roussinov, D. G. and Chen, H. (2001). Information Navigation on the Web by Clustering and Summarizing Query Results. *Information Processing & Management*, 37(6):789–816. 13, 54
- Ruppert, E., Klesy, J., Riedl, M., and Biemann, C. (2015). Rule-based Dependency Parse Collapsing and Propagation for German and English. In *Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology*, pages 58–66, Duisburg, Germany. 94, 98
- Rus, V. and Lintean, M. (2012). A Comparison of Greedy and Optimal Assessment of Natural Language Student Input Using Word-to-Word Similarity Metrics. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 157–162, Montréal, Canada. 106, 107
- Rus, V., Lintean, M., Banjade, R., Niraula, N., and Stefanescu, D. (2013). SEMILAR: The Semantic Similarity Toolkit. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 163–168, Sofia, Bulgaria. 114
- Rush, M. A., Chopra, S., and Weston, J. (2015). A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. 31
- Sabou, M., Bontcheva, K., Derczynski, L., and Scharl, A. (2014). Corpus Annotation through Crowdsourcing: Towards Best Practice Guidelines. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 859–866, Reykjavik, Iceland. 65, 66
- Sanderson, M. and Croft, B. (1999). Deriving Concept Hierarchies from Text. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 206–213, Berkeley, CA, USA. 34

- Sanderson, M. and Lawrie, D. (2000). Building, Testing, and Applying Concept Hierarchies. In Croft, W. B., editor, *Advances in Information Retrieval*, volume 7 of *The Kluwer International Series on Information Retrieval*, pages 235–266. Kluwer Acad. Publ, Boston, MA, USA. 14, 34
- Sandhaus, E. (2008). The New York Times Annotated Corpus Overview. Technical report, The New York Times Company, Research and Development. Available at https://catalog.ldc.upenn.edu/docs/LDC2008T19/new_york_times_annotated_corpus.pdf. 143
- Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009). The Graph Neural Network Model. *IEEE Transactions on Neural Networks*, 20(1):61–80. 150
- Schäfer, R. (2015). *Einführung in die grammatische Beschreibung des Deutschen*. Language Science Press, Berlin. 97
- Schmidhuber, J. (2015). Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61:85–117. 141
- Schneider, R., Oberhauser, T., Klatt, T., Gers, F. A., and Löser, A. (2017). Analysing Errors of Open Information Extraction Systems. In *Proceedings of the First Workshop on Building Linguistically Generalizable NLP Systems*, pages 11–18, Copenhagen, Denmark. 33
- See, A., Liu, P. J., and Manning, C. D. (2017). Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1073–1083, Vancouver, Canada. 31, 149
- Seeker, W. and Kuhn, J. (2012). Making Ellipses Explicit in Dependency Conversion for a German Treebank. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 3132–3139, Istanbul, Turkey. 94
- Shahaf, D. and Guestrin, C. (2010). Connecting the Dots Between News Articles. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 623–632, Washington, DC, USA. 34
- Shahaf, D., Guestrin, C., and Horvitz, E. (2012a). Metro Maps of Science. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1122–1130, Beijing, China. 34
- Shahaf, D., Guestrin, C., and Horvitz, E. (2012b). Trains of Thought: Generating Information Maps. In *Proceedings of the 21st International Conference on World Wide Web*, pages 899–908, Lyon, France. 34

- Shapira, O., Ronen, H., Adler, M., Amsterdamer, Y., Bar-Ilan, J., and Dagan, I. (2017). Interactive Abstractive Summarization for Event News Tweets. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 109–114, Copenhagen, Denmark. 53
- Shen, C. and Li, T. (2011). Learning to Rank for Query-Focused Multi-document Summarization. In *Proceedings of the IEEE 11th International Conference on Data Mining*, pages 626–634, Vancouver, Canada. 118
- Shen, R., Richardson, R., and Fox, E. A. (2003). Concept Maps as Visual Interfaces to Digital Libraries: Summarization, Collaboration, and Automatic Generation. In *European Conference on Digital Libraries*, Trondheim, Norway. 20
- Shneiderman, B. (1996). The Eyes Have It: A Task By Data Type Taxonomy for Information Visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*, pages 336–343, Boulder, CO, USA. 13
- Smith, A., Hawes, T., and Myers, M. (2014). Hi archie: Visualization for Hierarchical Topic Models. In *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*, pages 71–78, Baltimore, MD, USA. 16, 35
- Snow, R., O’Connor, B., Jurafsky, D., and Ng, A. (2008). Cheap and Fast – But is it Good? Evaluating Non-Expert Annotations for Natural Language Tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii. 65, 67
- Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, WA, USA. 141
- Sowa, J. F. (1984). *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, Reading, MA, USA. 15
- Sp rck Jones, K. (1972). A Statistical Interpretation of Term Specificity and its Application in Retrieval. *Journal of Documentation*, 28(1):11–21. 26
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958. 147
- Stanovsky, G. and Dagan, I. (2016a). Annotating and Predicting Non-Restrictive Noun Phrase Modifications. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1256–1265, Berlin, Germany. 89

- Stanovsky, G. and Dagan, I. (2016b). Creating a Large Benchmark for Open Information Extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2300–2305, Austin, TX, USA. 33, 63, 84
- Stanovsky, G., Dagan, I., and Adler, M. (2016a). Specifying and Annotating Reduced Argument Span Via QA-SRL. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 474–478, Berlin, Germany. 89
- Stanovsky, G., Ficler, J., Dagan, I., and Goldberg, Y. (2016b). Getting More Out Of Syntax with PropS. arXiv preprint 1603.01648v1. 32, 81, 83, 93
- Stanovsky, G., Michael, J., Zettlemoyer, L., and Dagan, I. (2018). Supervised Open Information Extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 885–895, New Orleans, LA, USA. 32
- Strötgen, J. and Gertz, M. (2015). A Baseline Temporal Tagger for all Languages. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 541–547, Lisbon, Portugal. 92
- Sukhbaatar, S., Szlam, A., Weston, J., and Fergus, R. (2015). End-To-End Memory Networks. In *Advances in Neural Information Processing Systems 28*, pages 2440–2448, Montréal, Canada. 150
- Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems 27*, pages 3104–3112, Montréal, Canada. 31, 141, 145, 150
- Suzuki, J. and Nagata, M. (2017). Cutting-off Redundant Repeating Generations for Neural Abstractive Summarization. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 291–297, Valencia, Spain. 31
- Tan, J., Wan, X., and Xiao, J. (2017). From Neural Sentence Summarization to Headline Generation: A Coarse-to-Fine Approach. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 4109–4115, Melbourne, Australia. 31
- Tanon, T. P., Vrandečić, D., Schaffert, S., Steiner, T., and Pintscher, L. (2016). From Freebase to Wikidata: The Great Migration. In *Proceedings of the 25th International Conference on World Wide Web*, pages 1419–1428, Montréal, Canada. 52
- Tarjan, R. E. and van Leeuwen, J. (1984). Worst-case Analysis of Set Union Algorithms. *Journal of the ACM (JACM)*, 31(2):245–281. 109

- Tauchmann, C., Arnold, T., Hanselowski, A., Meyer, C. M., and Mieskes, M. (2018). Beyond Generic Summarization: A Multi-faceted Hierarchical Summarization Corpus of Large Heterogeneous Data. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, pages 3184–3191, Miyazaki, Japan. 50
- Thadani, K. and McKeown, K. (2013). Sentence Compression with Joint Structural Inference. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 65–74, Sofia, Bulgaria. 127
- Thieroff, R. (2004). The Subjunctive Mood in German and in the Germanic Languages. In Abraham, W., editor, *Focus on Germanic Topology*, pages 315–358. Akademie Verlag, Berlin. 97
- Tixier, A., Malliaros, F., and Vazirgiannis, M. (2016). A Graph Degeneracy-based Approach to Keyword Extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1860–1870, Austin, TX, USA. 121
- Tseng, Y.-H., Chang, C.-Y., Rundgren, S.-N. C., and Rundgren, C.-J. (2010). Mining Concept Maps from News Stories for Measuring Civic Scientific Literacy in Media. *Computers & Education*, 55(1):165–177. 24
- Valerio, A. and Leake, D. B. (2006). Jump-Starting Concept Map Construction with Knowledge Extracted from Documents. In *Concept Maps: Theory, Methodology, Technology. Proceedings of the 2nd International Conference on Concept Mapping*, pages 296–303, San José, Costa Rica. 22, 24, 25, 26, 40, 60, 85, 90, 104, 133
- Valerio, A., Leake, D. B., and Cañas, A. J. (2008). Associating Documents to Concept Maps in Context. In *Concept Maps: Connecting Educators. Proceedings of the 3rd International Conference on Concept Mapping*, pages 114–121, Tallinn, Estonia and Helsinki, Finland. 24
- Valerio, A., Leake, D. B., and Cañas, A. J. (2012). Using Automatically Generated Concept Maps for Document Understanding: A Human Subjects Experiment. In *Concept Maps: Theory, Methodology, Technology. Proceedings of the 5th International Conference on Concept Mapping*, pages 438–445, Valetta, Malta. 20, 23, 54, 173
- van den Bosch, A., Bogers, T., and de Kunder, M. (2016). Estimating Search Engine Index Size Variability: A 9-Year Longitudinal Study. *Scientometrics*, 107(2):839–856. 1
- van Ham, F., Wattenberg, M., and Viegas, F. B. (2009). Mapping Text with Phrase Nets. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1169–1176. 34
- van Noortwijk, K. (2017). Integrated Legal Information Retrieval: New Developments and Educational Challenges. *European Journal of Law and Technology*, 8(1). 10

- Vanderwende, L., Banko, M., and Menezes, A. (2004). Event-Centric Summary Generation. In *Proceedings of the Document Understanding Workshop 2004*, Boston, MA, USA. 30
- Villalon, J. J. (2012). *Automated Generation of Concept Maps to Support Writing*. Ph.D. Thesis, University of Sydney. 20, 22, 23, 24, 25, 26, 54, 60, 86, 104
- Villalon, J. J. and Calvo, R. A. (2008). Concept Map Mining: A Definition and a Framework for Its Evaluation. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 357–360, Sydney, Australia. 20, 23
- Villalon, J. J. and Calvo, R. A. (2009). Concept Extraction from Student Essays, Towards Concept Map Mining. In *The 9th IEEE International Conference on Advanced Learning Technologies*, pages 221–225, Riga, Latvia. 20, 22, 40
- Villalon, J. J., Calvo, R. A., and Montenegro, R. (2010). Analysis of a Gold Standard for Concept Map Mining – How Humans Summarize Text Using Concept Maps. In *Concept Maps: Making Learning Meaningful. Proceedings of the 4th International Conference on Concept Mapping*, pages 14–22, Vina del Mar, Chile. 21, 43, 45, 74
- Vinyals, O., Bengio, S., and Kudlur, M. (2015a). Order Matters: Sequence to Sequence for Sets. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, CA, USA. 157
- Vinyals, O., Kaiser, Ł., Koo, T., Petrov, S., Sutskever, I., and Hinton, G. (2015b). Grammar as a Foreign Language. In *Advances in Neural Information Processing Systems 28*, pages 2773–2781, Montréal, Canada. 145
- Wang, L. and Ling, W. (2016). Neural Network-Based Abstract Generation for Opinions and Arguments. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 47–57, San Diego, CA, USA. 31
- Weston, J., Bordes, A., Chopra, S., Rush, A. M., van Merriënboer, B., Joulin, A., and Mikolov, T. (2015a). Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. arXiv preprint 1502.05698v10. 151
- Weston, J., Chopra, S., and Bordes, A. (2015b). Memory Networks. In *Proceedings of the 3rd International Conference on Learning Representations*. 150
- White, A. S., Reisinger, D., Sakaguchi, K., Vieira, T., Zhang, S., Rudinger, R., Rawlins, K., and van Durme, B. (2016). Universal Decompositional Semantics on Universal Dependencies. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1713–1723, Austin, TX, USA. 33, 93

- White, R. W. and Roth, R. A. (2009). Exploratory Search: Beyond the Query-Response Paradigm. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 1(1):1–98. 9
- Wities, R., Shwartz, V., Stanovsky, G., Adler, M., Shapira, O., Upadhyay, S., Roth, D., Martínez-Cámara, E., Gurevych, I., and Dagan, I. (2017). A Consolidated Open Knowledge Representation for Multiple Texts. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 12–24, Valencia, Spain. 52
- Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., and Nevill-Manning, C. G. (1999). KEA: Practical Automatic Keyphrase Extraction. In *Proceedings of the Fourth ACM Conference on Digital Libraries*, pages 254–255, Berkeley, CA, USA. 34
- Wu, F. and Weld, D. S. (2010). Open Information Extraction Using Wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 118–127, Uppsala, Sweden. 32, 99
- Wu, Z. and Palmer, M. (1994). Verb Semantics and Lexical Selection. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, Las Cruces, NM, USA. 106
- Yaari, Y. (1998). Texplora – Exploring Expository Texts via Hierarchical Representation. In *Proceedings of the Coling-ACL Workshop on Content Visualization and Intermedia Representations*, pages 25–32, Montréal, Canada. 34
- Yahya, M., Whang, S. E., Gupta, R., and Halevy, A. (2014). ReNoun: Fact Extraction for Nominal Attributes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 325–335, Doha, Qatar. 33
- Yang, H. (2012). Constructing Task-Specific Taxonomies for Document Collection Browsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1278–1289, Jeju Island, Republic of Korea. 34
- Yang, H. (2015). Browsing Hierarchy Construction by Minimum Evolution. *ACM Transactions on Information Systems*, 33(3):1–33. 34
- Yang, Y., Bao, F., and Nenkova, A. (2017). Detecting (Un)Important Content for Single-Document News Summarization. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 707–712, Valencia, Spain. 28, 118, 121

- Yao, J.-g., Wan, X., and Xiao, J. (2017). Recent Advances in Document Summarization. *Knowledge and Information Systems*, 53(2):297–336. 28
- Yeh, A. (2000). More Accurate Tests for the Statistical Significance of Result Differences. In *Proceedings of the 18th International Conference on Computational Linguistics*, Saarbrücken, Germany. 56
- Yimam, S. M., Ulrich, H., von Landesberger, T., Rosenbach, M., Regneri, M., Panchenko, A., Lehmann, F., Fahrner, U., Biemann, C., and Ballweg, K. (2016). new/s/leak – Information Extraction and Visualization for Investigative Data Journalists. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 163–168, Berlin, Germany. 10, 12, 13, 35
- Zajic, D., Dorr, B. J., Lin, J., and Schwartz, R. (2007). Multi-Candidate Reduction: Sentence Compression as a Tool for Document Summarization Tasks. *Information Processing & Management*, 43(6):1549–1570. 30
- Zhang, H., Chen, Z., Ma, W.-y., and Cai, Q. (2003). A Study for Document Summarization Based on Personal Annotation. In *Proceedings of the HLT-NAACL 03 Text Summarization Workshop*, Edmonton, Canada. 180
- Zhang, J., Tan, J., and Wan, X. (2018). Towards a Neural Network Approach to Abstractive Multi-Document Summarization. arXiv preprint 1804.09010v1. 31, 167
- Zhang, S., Rudinger, R., and van Durme, B. (2017). An Evaluation of PredPatt and Open IE via Stage 1 Semantic Role Labeling. In *Proceedings of the 12th International Conference on Computational Semantics (IWCS)*, Montpellier, France. 93
- Zhang, X., Li, G., and Feng, J. (2016). Crowdsourced Top-k Algorithms: An Experimental Evaluation. *Proceedings of the Very Large Databases Endowment*, 9(8):612–623. 66, 67
- Zheng, J., Chapman, W. W., Crowley, R. S., and Savova, G. K. (2011). Coreference Resolution: A Review of General Methodologies and Applications in the Clinical Domain. *Journal of Biomedical Informatics*, 44(6):1113–1122. 52, 105
- Zhila, A. and Gelbukh, A. (2013). Comparison of Open Information Extraction for English and Spanish. In *Proceedings of the International Conference on Computational Linguistics and Intellectual Technologies*, pages 714–722, Bekasovo, Russia. 33, 92
- Zopf, M., Botschen, T., Falke, T., Marasovic, A., Mihaylov, T., P.V.S, A., Loza Mencía, E., Fürnkranz, J., and Frank, A. (2018a). What’s Important in a Text? An Extensive Evaluation of Linguistic Annotations for Summarization. In *Proceedings of the Second International Workshop on Advances in Natural Language Processing*, page (to be published), Valencia, Spain. 5

- Zopf, M., Loza Mencía, E., and Fürnkranz, J. (2016a). Beyond Centrality and Structural Features: Learning Information Importance for Text Summarization. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 84–94. 180
- Zopf, M., Loza Mencía, E., and Fürnkranz, J. (2018b). Which Scores to Predict in Sentence Regression for Text Summarization? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1782–1791, New Orleans, LA, USA. 117
- Zopf, M., Peyrard, M., and Eckle-Kohler, J. (2016b). The Next Step for Multi-Document Summarization: A Heterogeneous Multi-Genre Corpus Built with a Novel Construction Approach. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 1535–1545, Osaka, Japan. 63, 73
- Zouaq, A., Gasevic, D., and Hatala, M. (2011). Ontologizing concept maps using graph theory. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, pages 1687–1692, TaiChung, Taiwan. 21, 52, 60
- Zouaq, A. and Nkambou, R. (2008). Building Domain Ontologies from Text for Educational Purposes. *IEEE Transactions on Learning Technologies*, 1(1):49–62. 21, 22, 24, 25, 40
- Zouaq, A. and Nkambou, R. (2009). Evaluating the Generation of Domain Ontologies in the Knowledge Puzzle Project. *IEEE Transactions on Knowledge and Data Engineering*, 21(11):1559–1572. 21, 24, 25, 40
- Zubrinic, K., Kalpic, D., and Milicevic, M. (2012). The automatic creation of concept maps from documents written using morphologically rich languages. *Expert Systems with Applications*, 39(16):12709–12718. 22, 40, 91
- Zubrinic, K., Obradovic, I., and Sjekavica, T. (2015). Implementation of method for generating concept map from unstructured text in the Croatian language. In *23rd International Conference on Software, Telecommunications and Computer Networks*, pages 220–223, Split, Croatia. 23, 25, 26, 27, 50, 54, 91, 104, 126, 129, 133

Anmerkungen zum Umgang mit Forschungsdaten

Gemäß der „Leitlinien zum Umgang mit Forschungsdaten“ der Deutschen Forschungsgemeinschaft[¶] wurden alle im Zusammenhang mit dieser Dissertation entstandenen Forschungsdaten langfristig archiviert und sofern möglich öffentlich zugänglich gemacht.

Folgende Forschungsdaten wurden frei verfügbar gemacht:

- **Korpora**

- Das in Abschnitt 4.3 beschriebene, neu erstellte Korpus steht unter der Creative Commons-BY-SA-4.0-Lizenz unter https://www.informatik.tu-darmstadt.de/ukp/research_6/data/summarization/concept_map_summaries zur Verfügung.

- **Software**

- Die für die in Unterabschnitt 4.3.3 beschriebenen Experimente notwendige Software steht unter der MIT-Lizenz unter <https://github.com/UKPLab/emnlp2017-cmapsum-corpus> zur Verfügung.
- Das in Abschnitt 5.3 beschriebene PropsDE-Tool steht unter der MIT-Lizenz unter <https://github.com/UKPLab/props-de> zur Verfügung.
- Die für die in Abschnitt 6.5 beschriebenen Experimente notwendige Software steht unter der MIT-Lizenz unter <https://github.com/UKPLab/ijcnlp2017-cmaps> zur Verfügung.
- Die in Kapitel 8 beschriebene Anwendung steht unter der Apache 2.0-Lizenz unter <https://github.com/UKPLab/emnlp2017-graphdocexplore> zur Verfügung.

- **Forschungsergebnisse**

- Alle im Zusammenhang mit dieser Dissertation stehenden Publikationen sind in der ACL Anthology (<https://aclanthology.coli.uni-saarland.de/>) verfügbar.
- Alle Forschungsergebnisse sind zudem auch in dieser Dissertation selbst dokumentiert, die von der Universitäts- und Landesbibliothek Darmstadt zur Verfügung gestellt wird.

Weitere in dieser Dissertation beschriebene Korpora können aus urheberrechtlichen Gründen nicht frei verfügbar gemacht werden. Entsprechend der DFG-Leitlinien sind diese Daten sowie damit zusammenhängende Software intern unter Nutzung der Infrastruktur der Universitäts- und Landesbibliothek Darmstadt archiviert, so dass eine Archivierung für mindestens 10 Jahre gewährleistet ist.

[¶]http://dfg.de/download/pdf/foerderung/antragstellung/forschungsdaten/richtlinien_forschungsdaten.pdf

Publikationsverzeichnis des Verfassers

Tobias Falke, Gabriel Stanovsky, Iryna Gurevych und Ido Dagan (2016): Porting an Open Information Extraction System from English to German. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, S. 892–898, Austin, Texas, USA.

Tobias Falke und Iryna Gurevych (2017): Utilizing Automatic Predicate-Argument Analysis for Concept Map Mining. In *Proceedings of the 12th International Conference on Computational Semantics*, Montpellier, Frankreich.

Tobias Falke und Iryna Gurevych (2017): Bringing Structure into Summaries: Crowdsourcing a Benchmark Corpus of Concept Maps. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, S. 2951–2961, Kopenhagen, Dänemark.

Tobias Falke und Iryna Gurevych (2017): GraphDocExplore: A Framework for the Experimental Comparison of Graph-based Document Exploration Techniques. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, S. 19–24, Kopenhagen, Dänemark.

Tobias Falke, Christian M. Meyer und Iryna Gurevych (2017): Concept-Map-Based Multi-Document Summarization using Concept Coreference Resolution and Global Importance Optimization. In *Proceedings of the 8th International Joint Conference on Natural Language Processing*, S. 801–811, Taipeh, Taiwan.

Markus Zopf, Teresa Botschen, Tobias Falke, Ana Marasovic, Todor Mihaylov, Avinesh P.V.S, Eneldo Loza Mencía, Johannes Fürnkranz und Anette Frank (2018): What’s Important in a Text? An Extensive Evaluation of Linguistic Annotations for Summarization. In *Proceedings of the Second International Workshop on Advances in Natural Language Processing*, Valencia, Spanien.

Ehrenwörtliche Erklärung[‡]

Hiermit erkläre ich, die vorgelegte Arbeit zur Erlangung des akademischen Grades “*Dr.-Ing.*” mit dem Titel “*Automatic Structured Text Summarization with Concept Maps*” selbständig und ausschließlich unter Verwendung der angegebenen Hilfsmittel erstellt zu haben. Ich habe bisher noch keinen Promotionsversuch unternommen.

Darmstadt, den 16. November 2018

Tobias Falke, M.Sc.

[‡] Gemäß § 9 Abs. 1 der Promotionsordnung der TU Darmstadt