TU WIEN Informatics

Halbautomatisches Erstellen von Concept Maps

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering/Internet Computing

eingereicht von

Christoph Presch, BSc
Matrikelnummer 0602554

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller
Mitwirkung: Univ.Ass. Dr.techn. Manuela Waldner, MSc

Wien, 20. August 2020

Christoph Presch	Eduard Gröller

TU WIEN Informatics

Semi-Automatic Creation of Concept Maps

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering/Internet Computing

by

Christoph Presch, BSc
Registration Number 0602554

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller
Assistance: Univ.Ass. Dr.techn. Manuela Waldner, MSc

Vienna, 20th August, 2020

Christoph Presch	Eduard Gröller

Erklärung zur Verfassung der Arbeit

Christoph Presch, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 20. August 2020

Christoph Presch

Danksagung

Diese Diplomarbeit wäre ohne die Unterstützung und Hilfe von vielen großartigen Menschen, die mich durch mein Studium und darüber hinaus begleitet haben, nicht möglich gewesen. Ich möchte meinem Betreuer Prof. Eduard Gröller für das Betreuen, Lesen und wertvolle Rückmeldungen für diese Arbeit danken. Spezieller Dank geht an dieser Stelle auch an meine zweite Betreuerin, Manuela Waldner, die mir wertvolle Ideen, großartige Hilfestellungen und Interesse in der Zusammenarbeit an dieser Diplomarbeit entgegengebracht hat. Ich möchte auch den Kollegen am Institute of Visual Computing & Human-Centered Technology an der TU Wien danken, die mir bei meinen Präsentationen im Zuge der Diplomarbeit konstruktives Feedback gegeben haben.

Ich bin außerdem meinen Kollegen bei der UPPER Solutions GmbH für ihr Verständnis und die flexible Zeiteinteilung im Zuge meiner Studien dankbar. Ich möchte auch besonders meinen Freunden Andreas, Alexander, Renè, Judith und Christina, die nahezu alle im Bereich Software Engineering, User Interface oder Graphic Design arbeiten, danken. Sie haben bei der Benutzer Evaluierung mitgemacht und wertvolles Feedback und Einsichten im Umgang mit dem Prototypen dieser Diplomarbeit gegeben.

Ein weiterer, besonderer Dank gilt meinem Studiengefährten und lieben Freund Andreas, der mich durch meine Studien an der TU Wien und viele lange Lernnächte begleitet hat. Mein tiefster Dank gilt all meinen Freunden und meiner Familie. Ohne ihre Hilfe wäre es nicht möglich gewesen, meine Studien und Arbeiten in dieser Weise fortzuführen. Zu guter Letzt möchte ich mich bei Susanne bedanken, die mir über all die Jahre ihre Unterstützung und Liebe geschenkt hat.

Acknowledgements

This master thesis would not have been possible without the help and support of many great people, who accompanied me during my studies and beyond. I want to thank my advisor Prof. Eduard Gröller for overseeing, reading and providing feedback for this work. Special thanks are directed to Manuela Waldner, the assistant advisor, who provided valuable ideas, great support and interest in working with me on this thesis and proof-reading every part of it. I also want to thank the people at the Institute of Visual Computing & Human-Centered Technology at the TU Wien, where this thesis was created, and who provided constructive feedback during my talks.

I am thankful for my colleagues at UPPER Solutions GmbH, who made it possible by understanding and making time for my studies. Additionally, I want to thank my friends Andreas, Alexander, Renè, Judith and Christina, most of whom are professionals in the fields of Software Engineering, User Interface or Graphic Design. They helped by participating in the user study of this thesis and providing valuable feedback and insight.

Another thank you is directed to my fellow and dear friend Andreas, who accompanied me during my time at the TU Wien and all the long after-hours studying. Furthermore, I express my deepest gratitude to my friends and family, without whom I would not have been able to conduct all my studies and work. Finally, I would like thank Susanne for her continuous support and love through all the years and everything else.

Kurzfassung

Concept Maps sind eine Methode zum visuellen Erfassen und Darstellen von Wissen. Sie sind außerdem eine etablierte Methode in Bereichen der Pädagogik, Wissensorganisation und vielen weiteren Anwendungen. Eine Concept Map setzt sich aus Konzepten und beschrifteten Verbindungen zwischen diesen zusammen und wird visuell as Node-Link Diagramm dargestellt. Concept Map Mining ist der Prozess der versucht, Konzepte und Verbindungen aus unstrukturiertem Text zu extrahieren. Es gibt drei Methoden, um diesen Prozess durchzuführen: manuell, halbautomatisch und voll automatisch. Eine vollautomatische Extraktion kann das subjektive mentale Modell, welches ein Benutzer bei der manuellen Erstellung auf eine Concept Map übertragen würde, nicht widerspiegeln. Der manuelle Prozess der Concept Map Erstellung wird hingegen oftmals als mühsam und ineffizient empfunden, was deren weitreichende Verwendung limitiert.

Diese Diplomarbeit präsentiert einen halbautomatischen Ansatz zum Concept Map Mining, mit welchem versucht wird, die Lücke zwischen einer manuellen und vollautomatischen Erstellung zu füllen. Der Vorteil dieses Ansatzes ist, dass die Benutzer die Kontrolle über die Erstellung der Concept Map behalten, während ineffiziente manuelle Schritte reduziert werden sollen. Der präsentierte Ansatz besteht aus einer automatischen Textverarbeitung, die Konzepte und Verbindungen aus einem Textdokument extrahiert. Dies geschieht mit der Hilfe von aktuellen Techniken aus der linguistischen Datenverarbeitung und neuralen Koreferenz-Auflösung. Der zweite Teil des Ansatzes erlaubt das manuelle Erstellen von Concept Maps in einer Benutzeroberfläche, wobei die extrahierten Konzepte und Verbindungen dem Benutzer als Vorschläge zur Auswahl präsentiert werden.

In einer Benutzerstudie wurde ein Prototyp des Ansatzes gegen einen Gold-Standard von manuell benutzererstellen Concept Maps und Concept Maps, die von einem vollautomatischen Programm erstellt wurden, verglichen. Die Ergebnisse zeigen, dass die erstellten Concept Maps mit dem halbautomatischen Prototypen eine genauere Übereinstimmung mit dem manuellen Gold-Standard, als die voll automatisch erstellten, erreichen. Zusätzlich konnte in der Evaluierung eine erhebliche Steigerung der Effizienz und Benutzerzufriedenheit bei der Erstellung der Concept Maps, im Vergleich zur manuellen Erstellung der Gold-Standard Concept Maps, festgestellt werden.

Abstract

Concept maps are a method for the visualization of knowledge and an established tool in education, knowledge organization and a variety of other fields. They are composed of concepts and interlinked relations between them and are displayed as a node-link diagram. Concept map mining is the process of extracting concept maps from unstructured text. The three approaches to mine concept maps are: manual, semi-automatic or fully automatic. A fully automatic approach cannot mirror the mental knowledge model, which a user would transfer to a manually created concept map. The manual process is often perceived as tedious and inefficient, limiting a wide-range application of concept maps.

This thesis presents a semi-automatic concept map mining approach that tries to bridge the gap between all manual construction and fully automatic approaches. The advantage of this approach is that the users still have control over how their concept map is constructed, but are not impeded by manual tasks that are often repetitive and inefficient. The presented approach is composed of an automatic text processing part, which extracts concepts and relations out of an unstructured text document and is powered by state-of-the-art natural language processing and neural coreference resolution. The second manual concept map creation part allows the creation of concept maps in a user interface and presents the extracted concepts and relations as suggestions to the user.

In a user study, an implemented prototype of the proposed semi-automatic concept map mining approach was evaluated. Manual gold standard concept maps that were created by the users and concept maps created by a fully automatic tool were compared to concept maps that were created with the prototype, proving the usefulness of the process. Results show that concept maps created with the semi-automatic prototype are significantly more similar to the gold standard than the ones created by the fully automatic tool. Additionally, considerably improved efficiency in creation duration and user satisfaction could be observed in comparison to the manual creation of the gold standard maps.

Contents

CHAPTER 1

Introduction

Concept maps are an established method to visualize key concepts and aspects of a certain topic and illustrate semantic relations between its most important entities. In education, concept maps can facilitate the process of learning, understanding and memorization [NC06]. They also provide an easy tool for visualizing and helping to consolidate knowledge in areas like knowledge organization and management [HCF01] and often play a key part in a wide range of applications from organizing business information [CF12] to journalistic work [PANa] and can even be helpful for information retrieval tasks [CHC01].

Visually, concept maps are presented as a node-link diagram, where the nodes are encircled words that represent concepts [NC06]. They are connected with linking words that describe the relationships between these concepts [NC06]. Originally, concept maps were also defined with a hierarchy in mind, where the most general concept is ordered at the top of the map and the least general concept map at the bottom [NC06]. However, later works (e.g., Zubrinic et al. [ZKM12]) do not always follow the constraint of a strict hierarchy and instead describe the topology to be able to have a variety of forms. Hence, the work in this thesis does not require a concept map to follow this strict constraint.

To correctly identify key concepts and relations, a topic must first be understood by the creator of a concept map. The creator then manually constructs the map, usually with drawing programs or special graph creation tools. Such a concept map application was, for instance, created and used as part of a research paper to measure display space usage [GWLS17]. With this application, the user could manually create labeled nodes that could be connected to display semantic relations between the concepts. While users who were allowed to employ concept maps used much less other means to organize their findings, informal feedback suggests that fully manual creation of concept maps is perceived as a very tedious task.

The overall process of automatically creating a concept map from unstructured textual data is generally called "concept map mining" (CMM) [VC08]. The CMM process can be semi-automatic or fully automatic [ZKM12].

With current methods and advances in computational linguistics, this would directly suggest applying a fully automatic approach for concept map creation out of unstructured textual data. However, fully automatic approaches face four major challenges:

1. Extracting meaningful concepts,
2. Extracting meaningful relations between these concepts,
3. Identifying corresponding concepts across sentence boundaries (coreference resolution), and
4. Summarizing the extracted concepts and relations to a coherent concept map.

These challenges all partly arise due to difficulties in today's natural language understanding and natural language processing (NLP) techniques, which are heavily used in creating concept maps. For instance, automatic extraction of named entities usually does not reach sufficient accuracy for fully automatic utilization [YB18].

Another challenge is that concept maps can be highly subjective, depending on the desired use case. Fully automatic concept maps can be used to reflect subjective differences of a set of various texts about the same focus question, which were created by different users [VC08]. They fall short, however, if creating a set of diverse concept maps by different users of the same underlying text is the main objective. This task of creating a summary of a text by using a concept map is highly subjective and can be influenced by a range of factors like personal background knowledge and subjective feelings and opinions [Fal19]. Hence, a fully automatic concept map creation approach may not be suitable for the latter purpose.

Therefore, a semi-automatic approach to partially create and extend an existing concept map is designed and implemented in this thesis. It should yield better results than a fully automatic approach, but it makes user intervention necessary. The objective of this thesis is to improve the efficiency of CMM on unstructured online text by using a semi-automatic approach and provide meaningful automatic suggestions for an existing concept map, while keeping the usability and real-time performance in mind. There are two major challenges that will be examined to solve this problem:

- At first, a state-of-the-art NLP pipeline with a suitable algorithm to mine the unstructured text and extract meaningful concepts and relations, which can subsequently be used as suggestions for the concept map, must be developed.
- Secondly, it will be crucial to find a user interface design/solution of how to present the extracted suggestions for new concepts and relations in the concept map to the user, while minimizing visual clutter and gaining high recall.

Two main hypotheses can be derived from these identified challenges, which need to be supported in order to justify the use of a semi-automatic concept map mining approach:

- H1: The semi-automatic approach can lead to a higher accuracy than the fully automatic one.

The semi-automatic CMM application needs to be able to achieve a higher accuracy in terms of representing user intention in comparison to a fully automatic approach. Even when automatic methods are partly used to extract concepts and relations out of unstructured textual data in a semi-automatic tool, the users should be able to create their subjective summary of the text with the concept map. Otherwise, a fully automatic approach would be more effective, because the summarization task, which is done manually in a semi-automatic approach, could be automated as well.

- H2: A semi-automatic approach is more efficient compared to a fully manual creation of a concept map.

The procedure of creating a concept map from unstructured textual data with the semi-automatic CMM application needs to be more efficient and less cumbersome than with a manual application. It should assist the user with helpful features and offer support that enables the user to focus on the summarization task itself. In contrast, diverting the users' attention by obstructing them with repetitive tasks, such as creating and aligning the maps' concepts and relations, should be avoided. If such a semi-automatic tool cannot deliver these improvements, it would be easier to adopt a manual tool to fulfill this task. A semi-automatic prototype application that addresses these major challenges was designed and implemented in this thesis. This protoype is based on existing techniques in CMM and NLP research:

The fundamentals to understand the purpose, usage and applications of concept maps were researched in Section 2.1. Basic natural language processing methods and NLP pipeline building blocks are outlined in Section 2.2.

The current state-of-the-art of CMM in related work was analyzed and is summarized in Chapter 3. A basic framework for CMM, on which most concept map creation techniques are based on, is illustrated in Section 3.1. Recent state-of-the-art for all three concept map creation methods can be found in their respective subsections: Section 3.2 – manual concept map creation, Section 3.3 – fully automatic concept map mining, and Section 3.4 – semi-automatic concept map mining.

The main idea and considerations about the design and implementation of the semi-automatic prototype is presented in Chapter 4, along with detailed explanations about the used NLP pipeline, extraction algorithms, and user interface design.

The evaluation of the semi-automatic prototype application and of the proposed hypotheses was conducted in correspondence with a limited expert user study. Five test users were

asked to each create five concept maps, at first with a manual concept map creation tool, which served as a gold standard, and subsequently with the semi-automatic prototype. Another five concept maps were also created with an automatic tool for comparison. Afterwards, the users answered a qualitative questionnaire and conducted a heuristic usability evaluation about their experience with the manual tool and the semi-automatic prototype. Additionally, quantitative comparison measures from all created concept maps were computed. The methodology for these five user trials, their measurements, and results are presented in Chapter 5, which should offer valuable insights in the possible usefulness of a semi-automatic CMM approach.

Finally, the contributions of this thesis to scientific research in the area of CMM can be summed up as follows:

1. A web-based semi-automatic CMM tool was developed that extracts concept and relation suggestions from a single text document and allows a user to efficiently create and extend a concept map in a user interface.

2. Results from a user study, which show that semi-automatic CMM can deliver a higher accuracy than fully automatic CMM compared to a manually created gold standard, and is more efficient than fully manual concept map creation.

CHAPTER 2

Foundations

This chapter covers two topics that are essential for understanding the process of concept map creation. Section 2.1 examines the foundations of concept maps, how they are composed, how their usefulness was discovered and for which real-world scenarios they can be applied. Section 2.2 presents the fundamentals of natural language processing (NLP), the major computational technique that will be used to mine the textual data for the semi-automatic concept map creation approach.

2.1 Concept Maps

Concept Maps were originally developed by Joseph D. Novak in a research program in 1972 [NM91]. In this program, which was based on the learning psychology of David Ausubel [Aus63], he strived to understand changes in the knowledge of children about science. According to Novak and Cañas [NC06], the learning psychology by Ausubel states the following: The main idea is that a learner has an individual cognitive structure that is composed of concepts and propositions. The process of learning is then defined as incorporating "*new concepts and propositions into existing concept and propositional frameworks held by the learner*" [NC06]. In other words, new concepts and propositions are merged with the existing individual cognitive structure of a learner. The term "knowledge structure" can be used synonymously. In this regard, Ausubel made a distinction between rote learning and meaningful learning, where rote learning relies on repetition of new knowledge and meaningful learning tries to integrate new knowledge into an already existing framework of concepts and relations. Both learning techniques are said to be able to absorb new knowledge in the long-term memory, but the disadvantage of rote learning is that knowledge is more quickly forgotten if it is not repeated enough. Furthermore, the knowledge is not able to be integrated in or enhance the cognitive structure of the learner. In resemblance to the cognitive structure, Novak was led to invent concept maps:

"*Out of the necessity to find a better way to represent children's conceptual understanding emerged the idea of representing children's knowledge in the form of a concept map. Thus was born a new tool not only for use in research, but also for many other uses.*" [NC06]

Novak's theory was that concept maps can facilitate meaningful learning, because they act as a scaffold or template for organizing and structuring knowledge [NC06]. Additionally, they allow a learner to apply the learned knowledge in new contexts and retain it for a longer period of time [NC06]. Research in education on applying the concept mapping technique has since then shown that using concept maps to enable meaningful learning can indeed serve as a viable tool for this purpose [CCF+03]. Novak and Cañas [NC06] further defined concept maps as "*graphical tools for organizing and representing knowledge*". They are composed of concepts in the form of one or more words, which are connected or related with lines that have labels and are called linking words or linking phrases [NC06]. These phrases can describe any possible relationship and are not limited to predefined values, like "is-a" or "part-of" relations, as used in semantic web ontologies [CCH+05]. In this thesis, we will refer to linking words or phrases as relations. Together, the concepts and relations form propositions or meaningful statements that are also called semantic units or units of meaning [NC06]. Novak and Cañas [NC06] also observed that concept maps are best to be constructed with a certain question in mind that needs to be answered, which he called "focus question". This explanation of concept maps was consequently illustrated in its own concept map by Novak and Cañas [NC06]. An excerpt of this concept map can be seen in Figure 2.1.

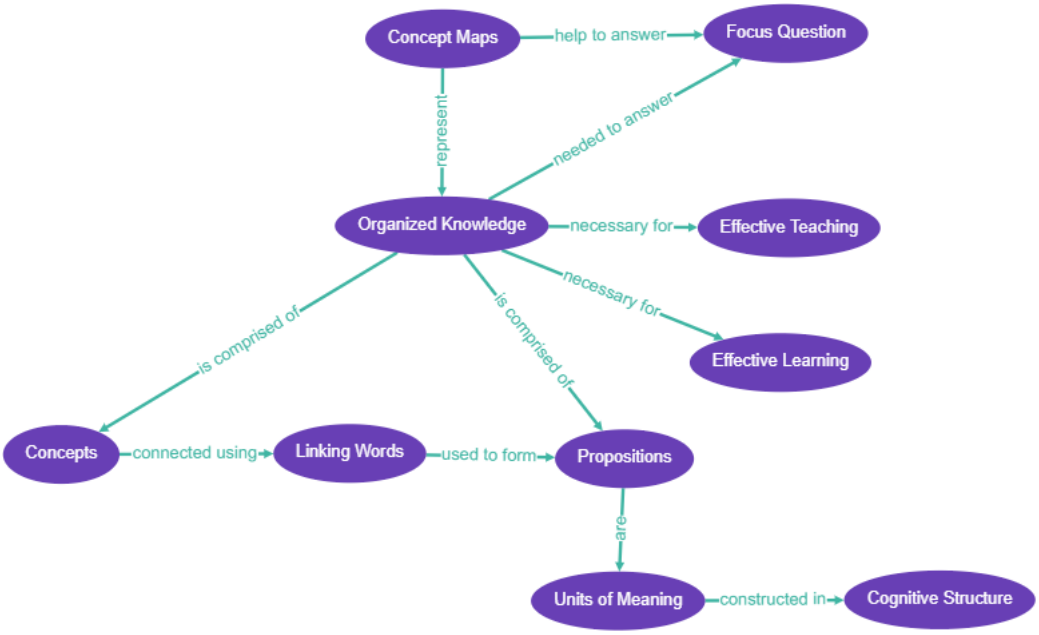

Figure 2.1: Excerpt of a concept map created by Novak and Cañas [NC06] that shows the main idea of concept maps.

Other features of concept maps are the hierarchical structure that can emerge when creating a concept map and where more general concepts are arranged at the top and less general concepts at a lower position [NC06]. Additionally, so called *cross-links* can be created that are basically relations between concepts in different segments or domains of a concept map and demonstrate creative thinking of the concept map creator [NC06].

Cañas et al. [CCH+05] later described important properties of a "well constructed" concept map (CM): Concepts should be nouns, relations verbs, and both should be as short as possible. Together, the concepts and relations should form a statement or proposition that makes sense. The map should be hierarchically structured and the root node representative for the topic of the map. However, the limitation of having generalization or hierarchical levels is not always seen as a strict constraint. For instance, Zubrinic [ZKM12] argues that the "*topology of a CM can take a variety of forms ranging from hierarchical, to non-hierarchical and data-driven forms*". The structure of a concept map also depends on the context where the map is used in and can therefore also vary between maps out of different contexts where similar concepts are chosen [CCH+05]. Therefore, Cañas et al. [CCH+05] argue that another important property of concept maps is that that a concept map represents the knowledge of its creator in a certain context. Hence, two concept maps about the same topic, created by different persons, can be equally correct and there is not always one true concept map.

2.1.1 Definition

Villalon and Calvo [VC08] defined the formal specification of a concept map CM as being a triplet:

$$CM = \{C, R, G\} \tag{2.1}$$

C is described as being a set of concepts and each concept c_i can either be a word or phrase, and the concept is unique in C [VC08]:

$$C = \{c_1, c_2, ..., c_n\} \tag{2.2}$$

$$c_i \in C;\ 1 \leq i \leq n \tag{2.3}$$

$$c_i, c_j \in C;\ c_i \neq c_j \tag{2.4}$$

R is defined as a set of relationships between two concepts c_p, c_q, where each relation r_i connects two concepts and has a label l_i [VC08]:

$$R = \{r_1, r_2, ..., r_m\} \tag{2.5}$$

$$r_i \in R = (c_p, c_q, l_i) \tag{2.6}$$

$$1 \leq p \leq n;\ 1 \leq q \leq n;\ 1 \leq i \leq m \tag{2.7}$$

$$c_p, c_q \in C;\ p \neq q \tag{2.8}$$

G is defined as a set of generalization levels [VC08] or hierarchical levels [ZKM12]. Each level g_s includes a set of concepts that belongs to the same level of generalization or hierarchy. Therefore, g_s is on a higher generalization or hierarchical level than g_k, if $k < s$ [VC08]:

$$G = \{g_1, g_2, ..., g_s\} \tag{2.9}$$

$$g_{k-1} < g_k < g_{k+1};\ 1 < k < s \tag{2.10}$$

$$g_k \subseteq G = \{c_1, c_2, ..., c_r\};\ 1 \leq r \leq n \tag{2.11}$$

As explained before, this thesis permits a less restricted form of a concept map and does not require a strictly hierarchical topology. Therefore, the generalization level will not be considered in its definition. For this thesis the definition of a concept map will be:

$$CM = \{C, R\} \tag{2.12}$$

2.1.2 Terminology

In literature, the term "concept map" is mostly used if it refers to visualizing knowledge. The term "knowledge graph", on the other hand, was used by Google in 2012 and stems from its use in semantic web search, where the graph is created from semi-structured data based on ontologies [Pau17]. These ontologies define a set of allowed entities and relations in a schema. Paulheim [Pau17] writes that, in a general sense, "*any graph-based representation of some knowledge could be considered a knowledge graph*". He, however, also defines a set of characteristics a knowledge graph needs to satisfy, such as that it requires a schema. Concept maps, on the other hand, do not require a schema. As

stated before, relations in concept maps are also not bound to a specific set of predefined values [CCH+05]. In consequence, and for this thesis, the term "concept map" will be used to refer to "visual representations of knowledge", as defined by Villalon et al. [VC08].

Many readers will also be familiar with the term "mind map" and might wonder if there is a difference to a "concept map". Falke [Fal19] lists the possibilities for structured text representations and thereby distinguishes mind maps from concept maps. According to Falke [Fal19], mind maps are simply depicted as concept maps with unlabeled relations. Out of the author's own experience, concept maps, which would be correctly called "concept maps" by Novak's [NC06] definition, would often be called "mind maps" by their creators, even if their relations are labeled. The term "concept map" does not seem to be common in the German speaking world, at least outside of the scientific community. It therefore might be that the term "mind map" is often used interchangeably for the same type of map, whether the relations have labels or not. This statement can be supported by findings in the evaluation part of this thesis (see Chapter 5), where none of the test users have ever heard of the term "concept map" before, but all have heard of and used "mind maps". At least for them, a "concept map" and "mind map" were the same.

2.1.3 Use Cases and Applications of Concept Maps

In a general sense, it can be concluded that concept maps are tools for knowledge visualization. This implies further possible use cases and applications for this tool, which go beyond its initial purpose and further than using them only in learning environments. The following listing is not to be seen as a complete enumeration of all existing applications, but should rather give a brief overview of what is possible:

Education: As explained before, concept mapping has its origin in education with its initial application as being a method to facilitate meaningful learning and represent the knowledge of a student about a certain topic. Concept maps can be an effective tool for learning if students construct them themselves, and they also proved to be useful if students use them as a source for studying, besides traditional learning methods and materials like textbooks or classroom learning [NA06]. Seen from a teacher's point of view, concept maps can serve as a tool for the evaluation of a student's knowledge by comparing maps created by students to ones that were created by their teacher [NC06].

Visualizing Expert Knowledge: Another important application of concept maps is their use in expert systems, where they are utilized to capture and share knowledge held by experts of a certain topic [CCH+05]. This summarized expert knowledge can be utilized in many areas. To mention an early example, Hoffman et al. [HCF01] used concept maps to create a knowledge model of weather forecasts in the gulf coast region, called STORM-LK. An example of a STORM-LK concept map can be seen in Figure 2.2. Concepts in the STORM-LK maps were linked to external information resources, such as satellite images, radar, texts, and so forth. This was achieved by using the concept mapping application CmapTools [NC06], which will be showcased later in Chapter 3.2.

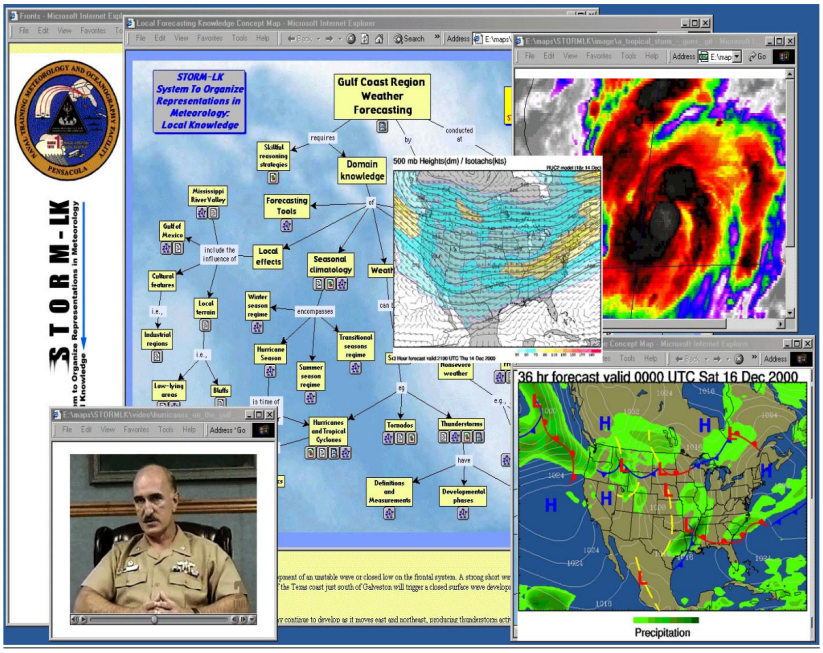

Figure 2.2: A STORM-LK concept map in use with linked external information [HCF01].

Storytelling: A wider range of applications can be summed up as "Storytelling", where the concept maps are embedded or used as visual summaries to emphasize certain circumstances or connections in a specific topic. For instance, The International Consortium of Investigative Journalists used concept maps as a visualization of the leaked documents about offshore companies and their connections to wealthy individuals and public figures [PANa]. With a graph database as a source, they illustrated these connections with the help of concept maps. Figure 2.3 shows a concept map from the offshore leaks database.

Organization and Planning: Concept maps can also be used as a tool to plan and collaborate on shared knowledge. Colosimo and Fitzgibbons [CF12] described how concept maps can be facilitated in the organization and project planning of libraries. They are used as reference points for project planning and evaluation, as well as a tool to share institutional knowledge with librarians and non-librarians and capture new ideas and resources. For instance, Colosimo and Fitzgibbons [CF12] used a concept map for a workshop to organize files and access relevant resources. The resources are attached to the corresponding concepts and can be accessed directly from the concept map. This concept map can be seen in Figure 2.4.

Semantic Web Ontologies: In semantic web, concept maps can be used as a template for creating ontologies. Zouaq et al. [ZGH11] describe a method to extract an ontology for the semantic web from initial concept maps. With this method, concepts and relations can be extracted, using metrics from graph theory to populate a semantic web ontology.

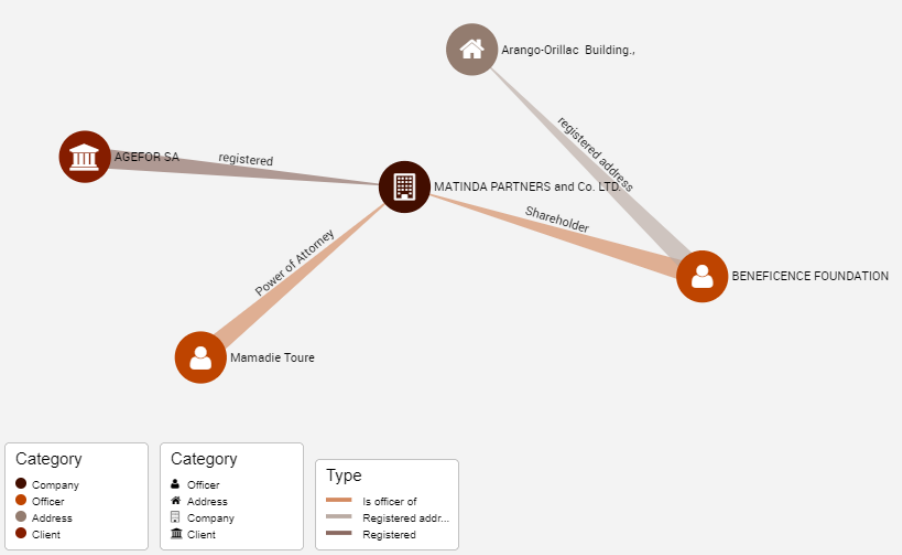

Figure 2.3: A concept map from the Panama Papers offshore leaks database [PANa].

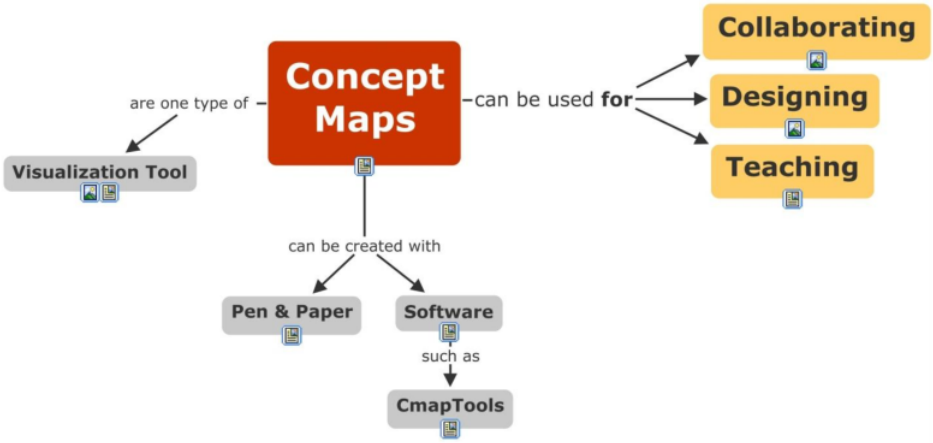

Figure 2.4: A concept map that was used to organize files and other resources for a workshop as illustrated by Colosimo and Fitzgibbons [CF12].

Information Retrieval and Summarization: Besides visual representations of textual data, concept maps have also been applied to aid in the retrieval of information and in summarizing found results. Carvalho et al. [CHC01] utilized the propositions and the hierarchical structure of a concept map to filter and rank results of search engines. Kraker et al. [KKE16] created a web application that summarizes results of a search engine for scientific papers. It groups each result to a concept, creates a label, and displays the results in the form of a map, which they call "Knowledge Map", but which technically cannot be called a traditional concept map since it lacks relations in its current state.

2.2 Natural Language Processing

In order to conduct the concept map mining and present suggestions for the concept map to the user, we need to get machine processible information out of simple text. This unstructured textual data, based upon which the concept map should be built, needs therefore to be converted and processed to structured data from which concepts and relations can be extracted. The overall technique of processing unstructured textual data stems from the research field of computational linguistics and is called **Natural Language Processing (NLP)**. The parts, we need to construct a concept map out of a text, are nouns for concepts, verbs for relations, and contextual information on where the nouns and verbs are located inside a text segment. This contextual information is necessary to locate the same entities across sentence boundaries and detect compound word phrases. For demonstration purposes, we consider the following short text segment:

The quick brown fox jumped over the lazy dog. Then it ran across the orchard.

With the help of NLP techniques we can determine the following facts about this text segment that help us to create so called *subject-verb-object* triples for a concept map:

- The segment is composed of two sentences.
- *Fox, dog,* and *orchard* are nouns.
- *Jumped* and *ran* are verbs.
- *Over* and *across* are prepositions.
- *Fox* is the subject of the first sentence.
- *Jumped* is the verb of the first sentence.
- *Dog* is the object of the first sentence.
- *It* is the subject of the second sentence.
- *Ran* is the verb of the second sentence.
- *Orchard* is the object of the second sentence.
- *It* in the second sentence refers to the *fox*.

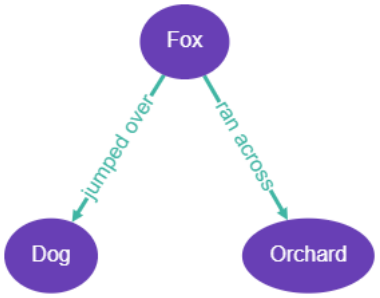

Figure 2.5: Concept map created with machine processible information about grammar and context retrieved by NLP methods.

From this knowledge we can create two *subject-verb-object* triples: "*fox - jumped over - dog*" and "*it - ran across - orchard*". Since we know that *it* in the second sentence refers to the *fox*, we can substitute *it* and form "*fox - ran across - orchard*". With these triples we can create a small concept map that can be seen in Figure 2.5. The subsequent sections explain the general NLP model, its most important steps and NLP methods to extract machine processible information. How each NLP pipeline step could be used for concept map construction, is explained in detail in Section 3.1.2.

2.2.1 The Natural Language Processing Pipeline

The linguistic information we can extract from textual data is retrieved in a sequential order of steps or layers, which is generally called a "Natural Language Processing Pipeline". The diagram in Figure 2.6 shows an example of a state-of-the-art NLP pipeline from the widely adopted open source NLP framework called "Stanford CoreNLP" [MSB+14].

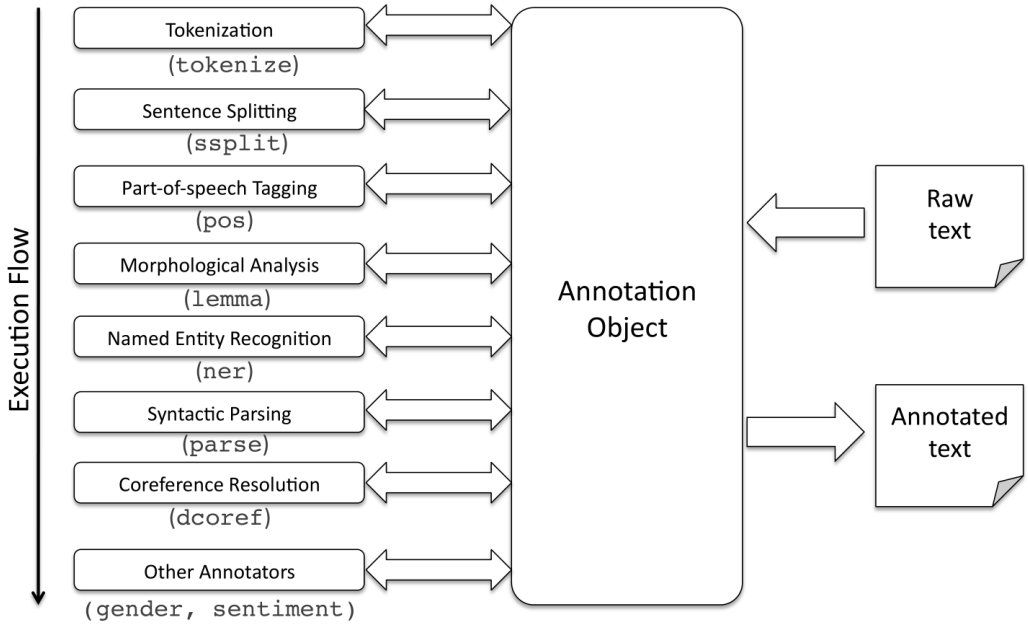

Figure 2.6: The Stanford CoreNLP pipeline [MSB+14].

In the diagram, we can see the general execution flow of an NLP pipeline, which is implemented similarly in other popular NLP frameworks like spaCy [SPA] or NLTK [BKL09]:

If a text is submitted to the NLP pipeline, an annotation object is created that holds the information that will get extracted in the execution flow and gets forwarded to the layers in the pipeline. Each layer then processes the text, adds additional information to the annotation object and returns it. The annotation object is then passed on to the next layer until every layer in the pipeline is done executing. In the end, we receive an annotated text that holds the extracted linguistic information, which can then be processed further.

The composition and order of the layers depends on each respective use case and the pipeline in Figure 2.6 shows steps that will be used for a typical NLP task [LHH19]. Some layers may need information from a previous processing step, while others can be run independently, although this also depends on the framework that is used [SPA]. For instance, if we want to compute the canonical form of a word in the lemmatization step, we depend on the splitting of a sentence into single words in the previous tokenization layer and the generation of part-of-speech (POS) tags from the POS tagging layer.

2.2.2 Tokenization

The process of dividing a document down to the smallest unit of meaning in a text, is called tokenization. The tokens can be words, punctuation marks, numbers, ASCII/Unicode emoticons, symbols, like a dollar sign, and many more [LHH19]. This is one of the first steps and often most important part of a NLP pipeline, because subsequent layers depend on it. There is hardly any NLP pipeline that does not make use of tokenization. The task can figuratively be compared with building a vocabulary for a text document [LHH19]. This vocabulary can then serve as a basis for further analysis, like determining the importance of words based on how often they occur in the text [LHH19].

Tokenization faces similar challenges like sentence segmentation. But, from identifying sentence boundaries we go down to a more granular level of identifying word boundaries, which is an even more complex task. There can be delimiters between words like commas, hyphens, or apostrophes. For instance, if we look at the word "wouldn't" there are multiple ways to tokenize it, like "would'nt", "wouldnt" or ["would", "nt"]. A token can also occur in the form of a compound word. Such compound words are called *n-grams* that can consist of n words [LHH19]. If we consider a compound word like "solar system", this could be tokenized as two separate words ["solar", "system"], which have both meaning on their own or the *2-gram* "solar system" itself. Another factor are frequently used stop words like "and", "or", "but" or "to" and so forth. They were traditionally filtered out by NLP pipelines, but could also be important to form *n-grams* and hence information would get lost [LHH19]. Tokenization is also language dependent like most parts of an NLP pipeline [LHH19].

Tokenizers traditionally worked with regular expressions and language dependent rule-based systems [DO12]. The most popular tokenizers derive their rules from the Penn Treebank (PTB) [MMS93], which is a large collection of 4.5 million words of American English that is annotated with part-of-speech (POS) information [DO12]. The PTB corpus was the defacto standard for many years and is still relevant today [DO12]. In the recent years, the focus tends to slightly move away from using it as the sole data source for NLP tasks, as Dridan and Oepen [DO12] observed:

"*As the NLP community has moved to process data other than the PTB, some of the limitations of the PTB tokenization have been recognized, and many recently released data sets are accompanied by a note on tokenization along the lines of: Tokenization is similar to that used in PTB, except . . .[sic]*"

With the recent trend of machine learning methods like deep neural networks in NLP, scientific research also engages with new methods for tokenization by applying supervised and unsupervised machine learning for this task. One example would be Kudo et al. [KR18], who developed the language independent sub word tokenizer and detokenizer SentencePiece for neural network-based NLP systems such as neural machine translation.

2.2.3 Sentence Splitting

Dividing a document into useful parts is called sentence splitting [MSB+14] or sentence segmentation [LHH19]. The segmented parts are commonly referred to as chunks of information [LHH19]. The segmentation of a document into sentences is usually one of the first steps in a NLP pipeline [LHH19]. It can happen before or after the tokenization step [MSB+14]. A sentence is the most granular part that can contain a meaningful statement in a text and contains in general enough information to create a proposition or unit of meaning in the form of a *subject-verb-object* triple [LHH19].

Sentence segmentation is not as trivial as it might seem on the first look, because sentence boundaries are not always easy to identify. Interestingly, what marks the end of a sentence also depends on the language at hand. For instance, the English language usually depends on the use of punctuation marks like a dot, exclamation, or quotation mark. On the other hand, if we consider the Chinese language, sentence boundaries are more fuzzy and less defined, since they often lack those boundary identifiers [HC11]. If we consider an example, we can see that simply splitting a sentence by dot, quotation, or exclamation mark does not work in all cases:

My coworker said with enthusiasm "I traveled to the remote Island St. Helena!" but I have never heard of that place before in my life.

If this sentence was part of a larger document, an ordinary split method for sentence segmentation would not yield satisfactory chunks, because there would be multiple options to split this string by exclamation mark or dot besides identifying the real sentence:

Chunk 1: *My coworker said with enthusiasm "I traveled to the remote Island St*
Chunk 2: *Helena!*
Chunk 3: *" but I have never heard of that place before in my life.*

There are several methods to address this problem nowadays, which can be used with the popular NLP frameworks: regular expressions that provide quick, but often incomplete solutions, machine learning algorithms in the form of neural networks, or logistic regression [LHH19]. Read et al. [RDOS12] defined three classes of methods in sentence segmentation or sentence boundary detection (SBD) that occur in literature and into which the former mentioned methods can be sorted:

1. Rule-based SBD that rely on heuristics (i.e., regular expressions or lists of abbreviations),
2. Supervised machine learning setups (i.e., text with labeled gold standard boundaries utilizing neural networks or other algorithms like decision trees), and
3. Unsupervised machine learning approaches (i.e. Punkt by Kiss and Strunk [KS06], which has the basic idea to identify collocational bonds between candidates and real boundaries).

2.2.4 Part-of-Speech Tagging

The objective of part-of-speech (POS) tagging is to annotate words in a text with their part-of-speech, which is their grammatical role in a phrase or sentence [LHH19]. In other words this means that POS tags describe how a word is syntactically used in speech or written text of a language. For example, the POS tagged text in Figure 2.7 shows that "quick" is an adjective, "fox" is a noun, and "jumped" is a verb in its past tense.

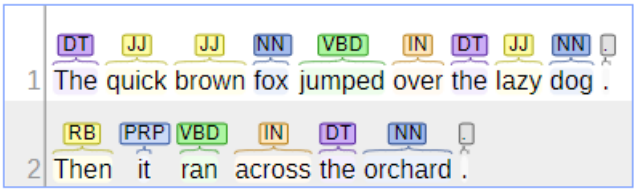

Figure 2.7: POS tagged text with Stanford CoreNLP [MSB+14].

POS taggers use tags from already annotated sets of text corpora like the Penn Treebank corpus [MMS93] to assign tags to tokenized documents. These annotated sets or collections of tags are also referred to as tagsets [BKL09]. An excerpt of POS tags from the Penn Treebank can be seen in Table 2.1.

Tag	Description
DT	Determiner
IN	Preposition or subordinating conjunction
JJ	Adjective
NN	Noun, singular or mass
NNS	Noun, plural
PRP	Personal pronoun
RB	Adverb
VB	Verb, base form
VBD	Verb, past tense

Table 2.1: Excerpt of POS tags used in the Penn Treebank [MMS93].

In practice, there are a variety of methods that a POS tagger can apply these tag information, gained from pre-annotated tagsets, on unprocessed documents. The earliest POS taggers used statistical methods like Hidden Markov Models and also rule-based systems and more naive methods like regular expressions [SD18]. State-of-the-art POS taggers use supervised machine learning models like Maximum Entropy Cyclic Dependency Network or Average Perceptron to reach a token accuracy of up to 98% [POS].

Averaged Perceptron [SHRS09] gives a good impression how algorithms for supervised POS taggers work: In the model, probabilities or weights for a tag and a specific word are trained by a set of features that include contextual information of a word, like tags of previous words or word forms. The model is then updated in iterations by comparing

classified probabilities with the gold standard, which is supervised training data in the form of an annotated corpus. New words are then tagged based on predictions by the trained weights in the model [SD18].

2.2.5 Morphological Analysis

Another important step in a NLP pipeline can be the normalization of tokens to their base form, which can be subsumed as morphological analysis [MSB+14], but is also referred to as normalization [LHH19]. This step helps, if we want to "*consolidate words that are intended to mean the same thing (and be spelled the same way) under a single token*" [LHH19]. This can be applied, if the same words are spelled with different casing, or if inflections or derived forms of a word are used, and if these words should be treated the same way [LHH19]. Normalization can, for example, be useful for improving the recall in search engines, where using a base form of a search term can broaden the search [LHH19]: If we search for the imaginary "Italian Cookery Schools Associations in Vienna" and apply normalization techniques we could retrieve results not only for pages that contain the precise search term, but also relevant results that contain "cooking", "school", or "assocation", like the "Viennese Italian Cooking School Association". Likewise, normalization can reduce the precision of search engines, because also less relevant results are retrieved [LHH19]. There are two ways to conduct normalization: stemming and lemmatization.

Stemming

Stemming is a way of normalization that aims to remove small variations, like pluralization suffixes or possessive endings of words, to get their common stem [LHH19]. Regular expressions and language specific rules are used in stemming algorithms for replacing derived forms of words with their stem [LHH19]. The most popular stemming algorithm for English is the Porter stemmer, which implements eight refined stemming rules [Por80]. For example, the Step 1a deals with "ss" suffixes, which can be seen in Table 2.2:

Rule	**Example**
SSES -> SS	caresses -> caress
IES -> I	ponies -> poni
	ties -> ti
SS -> SS	caress -> caress
S ->	cats -> cat

Table 2.2: Step 1a of the Porter stemmer [Por80].

Due to its relative simplicity and rule-based system, stemming falls short in various cases, where the stemming rules do not lead to the semantic word stem. Let us consider a good example given by Lane et al. [LHH19] to illustrate the shortcomings of stemming: If we consider the word "better", this would get stemmed to "bet" and the word would

get associated to gambling rather than its intended meaning of representing something "good". For these cases, lemmatization can help in finding the actual semantic root of a word, which is called a "lemma" [LHH19].

Lemmatization

With lemmatization, it is possible to find the root stem of a word, even if it is written with other characters than its inflection or derived form [JM19]. Therefore, lemmatization also considers the semantic meaning of a word, considering the context where the word is used, and provides generally better results than a stemmer [LHH19]. The technique has its origin in the field of morphological analysis, which engages with the question of how words are built up from smaller units called "morphemes". The most advanced lemmatizers rely on morphological parsing of the words [JM19]. In Figure 2.8 we can also see that, with lemmatization, it is possible to determine the base form of words that are used in a different grammatical tense, like the word "ran", which is used in past tense and has the base form "run".

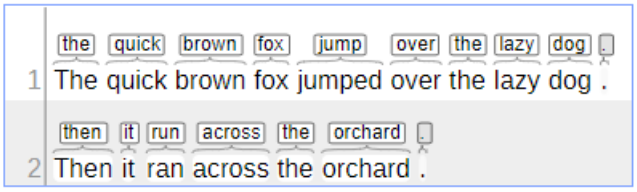

Figure 2.8: Annotated text with lemmas by Stanford CoreNLP [MSB+14].

Lemmatizers use knowledge bases, often in combination with POS tags, since context is necessary to identify the lemmas [LHH19]. One popular example of such a knowledge base is the lexical English database WordNet [Mil95], which contains semantic relations and makes lemmatization possible. WordNet lemmatizers are, for instance, used in the NLP framework NLTK [BKL09].

2.2.6 Named Entity Recognition

After tokenizing, POS tagging, and normalizing our document, additional important information can be the relation of extracted and annotated tokens to things or persons in the real world. This information can be extracted in the form of so called entities with the processing step named entity recognition (NER). A category for a named entity can be a person, location, organization, numbers, but also temporal descriptions like times, dates, and even other categories like prices or currencies [JM19]. Figure 2.9 shows a text on which NER was applied with the NLP tool spaCy [SPA] and where we can see several identified entities. Table 2.3 shows the corresponding NER categories used in spaCy [SPA].

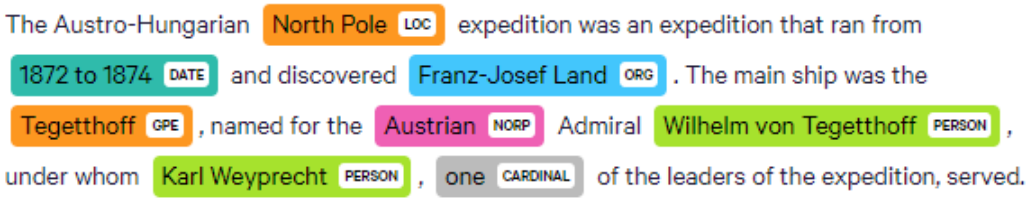

Figure 2.9: Named Entity Recognition computed with spaCy [SPA].

Category	**Description**
CARDINAL	Numerals that do not fall under another type.
DATE	Absolute or relative dates or periods.
GPE	Countries, cities, states.
LOC	Non-GPE locations, mountain ranges, bodies of water.
NORP	Nationalities or religious or political groups.
ORG	Companies, agencies, institutions, etc.
PERSON	People, including fictional.

Table 2.3: An excerpt of NER categories used in spaCy [SPA].

The difficulty in identifying named entities lies in detecting what is an entity and what is not and where boundaries for entities can be set [JM19]. Another problem is that entities can be ambiguous and therefore might possibly belong to multiple categories [JM19]. For example, if we consider the entity "Ibiza", potential entity categories could be "Location", "Geopolitical Entity" or even an "Event".

Standard NER algorithms apply word-by-word sequential labeling of named entities and "*the assigned tags capture both the boundary and the type*" [JM19]. While scientific research focuses on applying machine learning algorithms like Neural Networks, Conditional Random Fields (CRF), and Maximum Entropy Models (MEMM) for NER, commercial products often rely on rule-based systems that use a combination of lists and rules and only partly use machine learning methods [JM19].

2.2.7 Syntactic Parsing

Besides knowing which grammatical role a word plays in a sentence by applying POS tagging, it is important to know how a sentence is structured to identify the interrelations of words. For this task, we can use syntactic parsing, which is also referred to as dependency parsing, that arranges the structure of a sentence in the form of a tree with binary grammatical relations between words [JM19]. Each relation in the tree has therefore a dependant and a head, besides the root node, which has only dependents [BKL09]. The labels in the relations, also called modifiers, describe which grammatical role the dependent has for its head [JM19]. Recently, efforts have been made to standardize a multilingual set for labelling dependency relation modifiers in the Universal Dependen-

cies Project [NdMG+16]. Table 2.4 shows an excerpt of modifiers from the Universal Dependencies Project [NdMG+16].

Dependency	Description
advmod	Adverbial modifier - adverb or adverbial phrase modifying a predicate.
amod	Adjectival modifier - modifies meaning of a noun.
case	Case marking - words dependent of a noun.
det	Determiner - expressing reference of a noun in a context.
nsubj	Nominal subject - syntactic subject of a clause.
obl	Oblique nominal - used for nominals like nouns, corresponding to other adverbials like verbs.
punct	Punctuation - refers to punctuation in a sentence.

Table 2.4: An excerpt of dependencies from the Universal Dependencies Project [NdMG+16] with adapted descriptions.

For example, the relation label "amod" in the dependency parsed text in Figure 2.10 describes that "quick" and "brown" are *adjectival modifiers* of the head word "fox". In other words, this means that "fox" has the adjectives "quick" and "brown". For the head "jumped", "fox" has the relation label "nsubj" (a *nominal subject modifier*), and "dog" has the relation label "obj" (an *object modifier*). From this information, in combination with POS tags, we can infer that "fox" is the subject of the sentence, "jumped" the verb and "dog" the object, forming a *subject-verb-object* triple. Other modifiers in Figure 2.10 can be looked up in Table 2.4.

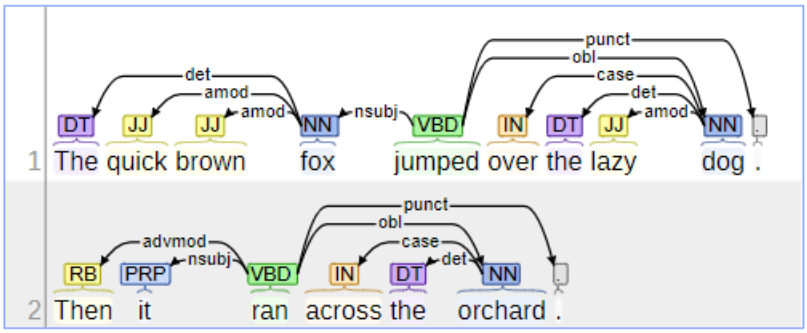

Figure 2.10: Word dependencies parsed with Stanford CoreNLP [MSB+14].

Conducting dependency parsing heavily relies on so callled dependency treebanks that are derived from the annotated corpora used in POS tagging [JM19]. Well-proven approaches for dependency parsing are Transition-Based Dependency Parsing and Graph-Based Dependency Parsing [JM19].

2.2.8 Coreference Resolution

An entity can have multiple representations across a document. For instance, if we consider the two sentences in Figure 2.11, the "fox" is referred to as "it" in the second sentence. The task of eliminating these ambiguities is called "coreference resolution" or "anaphora resolution" [LHH19]. The expressions "fox" and "it" are called *mentions* in the context of coreference resolution and two mentions that belong to the same entity are called *coreferer* [JM19].

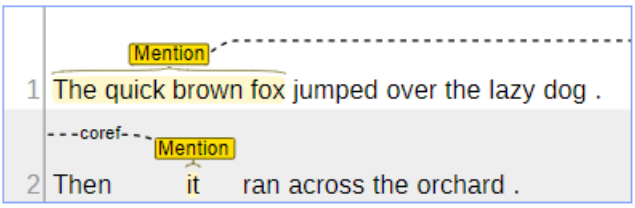

Figure 2.11: Coreference Resolution computed with Stanford CoreNLP [MSB+14].

Strongly simplified, the stages in a coreference resolution algorithm consist first of identifying possible mentions (mention detection) and then which of those mentions are possible coreferer (anaphoricity and coreference classification) [JM19]. Older coreference systems used own algorithms for each sub-task, whereas modern systems use a single end-to-end machine learning model that unifies these stages and computes scores for both tasks that are considered together in the decisions of the classifier [JM19]. The main architectures for modern coreference algorithms are [JM19]:

1. The Mention-Pair Architecture, where classifiers consider pairs of mentions and assign probabilities whether they are coreferer or not,
2. The Mention-Rank Architecture that classifies pairs of a current mention and an antecedent of that mention, and
3. Entity-Based Models, where the classifier considers not pairs but clusters of mentions and their possibilities of being corefering.

CHAPTER 3

Related Work

In this chapter, related work in the scientific research area of concept map construction and respective state-of-the-art solutions will be presented. Section 3.1 describes the general process of creating concept maps from unstructured textual data, which is called concept map mining (CMM). The term "Concept Map Mining" was first coined by Villalon and Calvo [VC08]. This process will serve as a scaffold procedure for developing the semi-automatic CMM approach for this thesis.

Kowata et al. [KCB10] tried to establish a classification system for CMM approaches in a meta study on CMM papers. They came up with several classification systems and organized them, each seen from a different perspective. The three main categories of the classification systems are [KCB10]:

1. Research Goals: What are the research goals? Who are the audiences of the resultant concept maps?

2. Data Sources: What kind of data source was used? What methods and techniques were used to handle it? Does the data source belong to a specific domain of knowledge?

3. Outputs: How does the resultant output look like? What tools were used to build the output? How do the approaches assess the resultant output?

For this thesis, we use the classification system that classifies CMM approaches based on the methods and techniques used for mining the concept maps. This classification system can, according to Kowata et al. [KCB10], be sorted into the classification category Data Sources. It looks at the different CMM approaches from the perspective of a *construction manner* and defines the following three classes for concept map construction:

1. Manual representation: According to Kowata et al. [KCB10], approaches in this class only extract concepts and relations from a text. The building process of the map has do be done manually with the help of another third-party tool that is not further specified. However, this definition does not clarify whether approaches that do not involve any type of computational text processing can be included in this class. Fully manual approaches are also quite common and many web tools exist that allow to create concept maps without textual pre-processing and concept/relation extraction. For the sake of completeness, we would also need to count concept maps drawn on paper and all other forms of manual CMM to this class.

2. Automatic representation: Approaches in this class natively contain all resources and means necessary to automatically to construct and visualize concept maps.

3. Semi-automatic representation: This class includes approaches that produce a set of propositions, which are imported by a third-party tool that subsequently enables the construction of a concept map. Kowata et al. [KCB10] further clarify that all approaches that require user intervention during the CMM process are not truly automatic and can be attributed to this class.

   For instance, a system that mines text, extracts concepts and relations, but provides no further means to construct the concept map was proposed by Valerio and Leake [VL06]. This system was said to belong to the manual representation class. Alves et al. [OPC01] presented a system that uses two modules in conjuction, where the first module focuses on extracting concepts and relations from text. The second module builds the concept map and interactively asks the user questions to construct the concept map. This system belongs to the semi-automatic class, according to the classification by Kowata et al. [KCB10].

Based on the classification system by Kowata et al. [KCB10], we introduce a slightly simpler classification system for this thesis that also looks at CMM approaches from the *construction perspective*:

1. Manual Concept Map Creation: Approaches are included in this class that either do not contain a system for textual pre-processing or provide no means to natively construct a concept map out of extracted concepts and relations. Section 3.2 gives an overview of such tools.

2. Fully Automatic Concept Map Mining: All tools are included in this class that have the ability to construct concept maps fully automatically from textual or other data formats without user intervention after submitting the source data. Section 3.3 will present state-of-the-art work on this CMM class.

3. Semi-Automatic Concept Map Mining: Systems are included in this class that enable the full construction of a concept map with automatically extracted concepts

and relations based on textual pre-processing, but require user intervention. In Section 3.4, we further explore semi-automatic CMM approaches.

3.1 Concept Map Mining

Villalon and Calvo [VC08] first proposed a definition for the composition of necessary steps to conduct CMM and subsequently developed a CMM process. They defined the CMM process as "*the proper identification of a concept map CM from a document D*" [VC08] and described three steps in this process:

1. Concept extraction (CE), which deals with identifying the concepts in text and must come first in the process, because subsequent steps depend on it,
2. Relation extraction (RE), where relations between the identified concepts are determined, and
3. Topological extraction (TE) that engages with computing generalization levels for the identified concepts and can be run independently from the RE step.

Additionally, Villalon and Calvo [VC08] introduced two sub-tasks for each step: Identification, where concepts, relations, and generalization levels have first to be identified. Summarization, that subsums the task for finding representative subsets for concepts, relations, and generalization levels, and combining them to a coherent concept map.

Villalon and Calvo [VC08] also formalized this CMM process as follows: We need to find a triple CM, where CM is the final concept map as defined in Section 2.1.1. This concept map CM is mined from a text document D that represents a triple, where CD are all concepts, RD all relations and GD all generalization levels that are contained in the document. In the resulting concept map CM, C is a subset of all concepts CD, R a subset of relations RD and G a subset of generalization levels GD:

$$D = \{CD, RD, GD\} \tag{3.1}$$

Later works describe variations of this CMM process (Zubrinic et al. [ZKM12] [ZOS15], Aguiar and Cury [AC16], Falke [Fal19]), where the arrangement of steps does not necessarily contain the described sub-tasks. Instead, the sub-task of summarization acts as an own step in the CMM process or the task of topological extraction is used as a sub-task of another step. The arrangement and order of steps can be seen similarly to steps in an NLP pipeline, where a pipeline of steps turns textual data into a concept map [Fal19]. Figure 3.1 shows a general CMM process, where the generalization is done later in the CMM pipeline as a part of the summarization step. Generalization levels are called hierarchical levels and are denoted with T in this CMM pipeline.

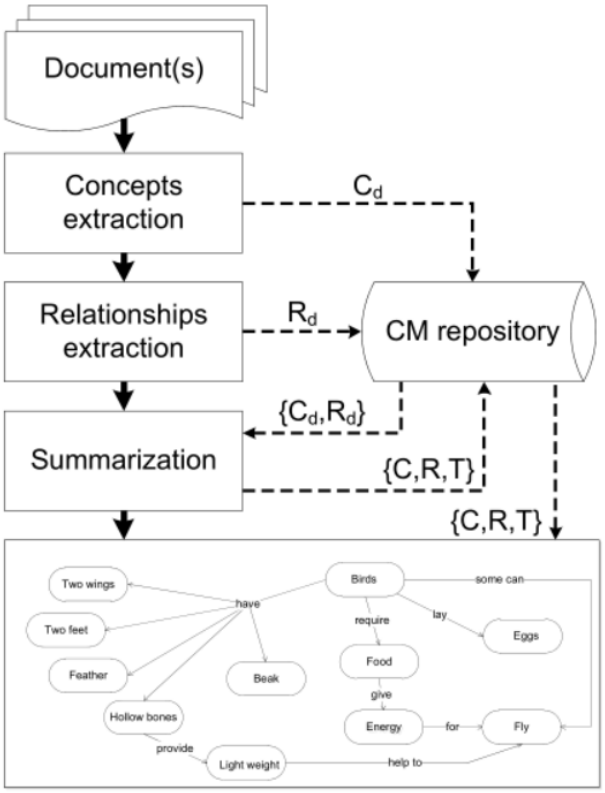

Figure 3.1: Diagram of a general CMM process as illustrated by Zubrinic et al. [ZKM12].

In the following Subsection 3.1.1 and Subsection 3.1.2 examples for CMM data sources and methods will be given. However, some of these examples (Böhm and Maicher [BM06], Paweł et al. [MDP06], Chen et al. [CKWC08], and Tseng et al. [TCRR10]) come with a caveat. They have in common that it was unclear, whether the created concept maps fully comply with Novak's definition [NC06], since the relations in the generated maps were not labeled. A clear differentiation of this kind is not always common in related works.

3.1.1 Data Sources for Concept Map Mining

Until now, we only talked about textual data sources for CMM, which is the most common data source for a CMM process [ZKM12]. Some methods use multiple source documents, while others use only single textual data sources for their CMM process [ZKM12]. Besides text, Zubrinic et al. [ZKM12] listed other data sources that are less commonly used: Structured textual data sources like domain ontologies for semantic web purposes and non-text sources like speech and video. In the following paragraphs, examples of CMM techniques based on other data sources are given. Except from these examples, the thesis will again focus only on unstructured textual data sources.

Böhm and Maicher [BM06] presented a system called SemanticTalk that creates "Topic Maps" from speech in real time. The speech was first converted to text with a commercial speech-to-text conversion system. NLP methods were then applied to extract concepts and relations, which were further processed to output a map in RDF format.

Graudina and Grundspenkins [GG08] developed a method for CMM based on OWL ontologies that are used in the semantic web to define semantic relations between entities: They utilized the similarities between concept maps and OWL ontologies for this method. OWL ontologies define a set of entities and relations that form *subject-predicate-object* triples for which instances with concrete values can be created. For concept maps, these entities and relations would be labelled concepts and relations that can be used equivalently as *concept-relation-concept* or *subject-verb-object* triples. The algorithm that was developed depends on a predefined and verified *.owl ontology file. Entities and relations were extracted from this *.owl file and transferred to an incidence matrix. Based on this matrix, a concept map was then created and displayed to the user.

3.1.2 Methods applied in Concept Map Mining

Methods that were applied in CMM over the years are manifold. The most common and often applied methods nowadays were consolidated by Zubrinic et al. [ZKM12] as follows: statistical approaches, machine learning, usages of dictionaries, and the usage of linguistic tools and techniques. An overview is given in the subsequent sections:

Statistical Approaches

According to Zubrinic et al. [ZKM12], statistical methods have the following properties: The most used applications are to analyze co-occurrences of words to determine possible relations between concepts and measure the frequency of terms to calculate the importance of concepts in the summarization step. They are language independent and efficient, but are imprecise because the semantics of words that arise from their use in a specific language are not considered. Statistical methods that were used are: Term Frequency–Inverse Document Frequency (TF-IDF), Latent Semantic Analysis (LSA), Principal Component Analysis (PCA), and Self-organizing Maps (SOMs). These methods are often used in combination with other CMM methods. The resulting concept maps are often not hierarchical.

Valerio and Leake [VL06] used term frequency analysis to calculate weights and rank all words in a document to estimate their importance for the specific topic at hand. The rank of words was then used to select the most relevant concepts to create a more clear and complete concept map.

Villalon and Calvo [VC09] described a method for concept extraction from student essays that utilized grammatic parsers and Latent Semantic Analysis (LSA). For concept extraction, the document was processed with the Stanford parser, from which a grammar tree was obtained that allowed the identification of nouns and compound nouns. However, which version of the Stanford parser they used to process the document was not reported.

The identification was done using tree regular expressions. Summarization was computed with LSA, which extracts tokens from the text, creates features, and defines a model in the form of a matrix that represents terms and text passages. After that, angles between eigenvectors of the matrix could be used to calculate distance weights. Up to 25 terms that were identified as nouns and had a high weight were used as candidates for the concept map summary.

Tseng et al. [TCRR10] extracted key terms from Chinese news stories from which concept maps were formed that served as a tool to measure scientific literacy in media. For this task, they conducted term association analysis by measuring co-occurrence of two words in the same document and calculating association weights with a modified Dice coefficient. TF-IDF was then applied on a collection of news stories to measure importance of key terms that allowed the creation of a term relation structure. The combination of key terms, association weights, and the term relation structure subsequently enabled the creation of concept maps.

Machine Learning Methods

Supervised and unsupervised machine learning methods have also been applied in the context of CMM. Techniques that were used to extract concepts and relations can be attributed to classification systems, association rules, and clustering [ZKM12].

Zouaq and Nkambou [ZN08] created a tool called TEXCOMON that was used to create domain ontologies based on concept maps that were mined before from textual data. The system used a naive Bayes classifier machine learning algorithm called Kea-3.0 that extracts representative n-grams from text documents. These n-grams were then used to further extract relevant sentences and domain terms to form concepts and relations.

Wang et al. [WCLK08] combined NLP techniques with case-based reasoning for anaphora resolution of concepts. They proposed a system that generates propositions using fuzzy set theory to mine concept maps from unstructured text. Based on interactive user feedback, the system suggests new propositions. These propositions were determined with term weights for pairs of concept phrases that have a high co-occurrence. The term weights were calculated with heuristic fuzzy rules.

In recent years many machine learning methods that utilize Neural Networks have been adopted specifically for NLP tasks and therefore benefit NLP methods that are used in CMM. These state-of-the-art solutions for NLP have already been discussed in Section 2.2.

Usage of Dictionaries

Dictionaries of predefined terms are sometimes used to narrow down the concepts and relations that can potentially be retrieved from a document collection to the current domain or focus question [ZKM12]. With these stored words from the dictionary, similar words or words that often occur with it can be retrieved from a document [ZKM12]. This

can be specifically helpful in cases if concept mapping should be applied to a domain with an unconventional vocabulary.

Paweł et al. [MDP06] used the Unified Medical Language System (UMLS) as a dictionary, which is a collection of medical sources that also contains labeled concepts. They then applied POS tagging to map phrases in the text to concepts from the UMLS. Ambiguities of concepts were resolved afterwards with an additional module that uses a semantic network with scoring functions to choose the best fitting concepts for the text.

Chen et al. [CKWC08] developed a CMM method to create concept maps about eLearning from academic articles. At first, relevant research articles were collected in a database. The keywords from these articles were then extracted, indexed and grouped with PCA, and stored as a thesaurus in a database. Based on the importance of the keywords, which was also computed in the indexing step, a relation strength was calculated. Finally, concept maps were computed from the extracted keywords and relation strengths and saved to a database. Users would subsequently be able to query the concept maps from a user interface.

NLP Techniques

Since the most frequently used data sources for CMM are text documents, NLP methods naturally become a favored tool to process the text and extract information out of it. Virtually all parts of an NLP pipeline, which were explained in Section 2.2, have been applied in the context of CMM [ZKM12]. Falke [Fal19] illustrated a different CMM process in a diagram that is composed of more granular steps than the process by Zubrinic et al. [ZKM12] presented earlier in Figure 3.1. This process provides a good perspective on how NLP methods can be used in CMM and is given in Figure 3.2.

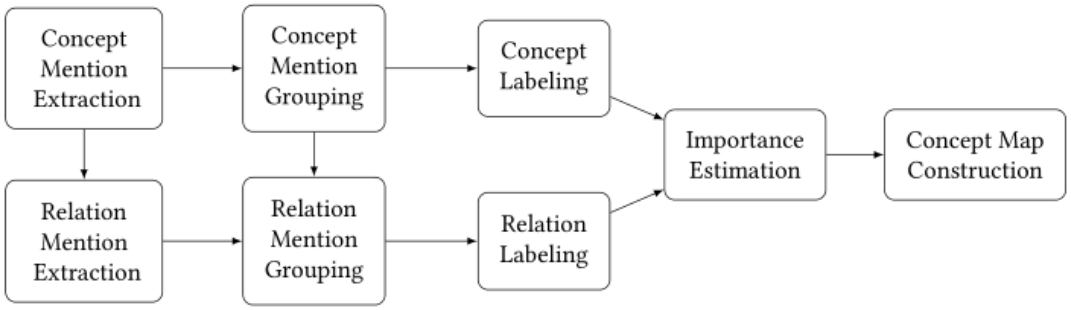

Figure 3.2: Diagram of a CMM process as illustrated by Falke [Fal19].

Tokenization and Sentence Segementation are typically used as pre-processing steps that are needed for "Concept Mention Extraction" and "Relation Mention Extraction". Villalon and Calvo [VC09] used tokenization as a pre-processing step to recognize terms for their LSA topic modeling. Zubrinic et al. [ZKM12] developed a CMM process specifically for the Croatian language and applied sentence segmentation to divide the document

into sentences. Afterwards, they used tokenization to divide the sentences into tokens and identify the basic language elements.

POS tagging can be used to identify nouns, verbs, and adjectives, which serve as concepts and relations. Therefore, POS tagging is also needed for "Concept Mention Extraction" and "Relation Mention Extraction". Villalon and Calvo [VC08] proposed POS tagging as a method to extract nouns and verbs in their CMM process. Zubrinic et al. [ZOS15] used a Croatian POS tagger. Aguiar and Cury [AC16] utilized POS tags in their Morphological Analysis for concept and relation identification.

Normalization methods like Stemming and Lemmatization are used to identify the same concepts over a document, even if different forms or inflections of these words are used. With these techniques, base forms of words can be discovered and similar morphological variants of words can be grouped to the same concepts or relations, which is necessary for the tasks of "Concept Mention Grouping" and "Relation Mention Grouping" [Fal19]. These tasks group the mentions of the same concepts and relations together, in order to create a set of concepts and relations for the concept map without duplicates [Fal19]. Normalization was conducted with stemming by Valerio and Leake [VL06], who also incorporated WordNet [Mil95] to find additional synonyms. They hereby determined word similarity to create sets of similar concepts. Zubrinic et al. [ZOS15] utilized lemmatization for their Croatian CMM process in a linguistic pre-processing step before extracting concepts. However, this resulted in some incorrect concepts, because case, number, or gender of concepts were incorrect in the final concept maps.

The steps "Concept Labeling" and "Relation Labeling" in Falke's [Fal19] CMM process refer to selecting a representative label from the groups of concepts and relations. One way would be to select the most frequent instance of a concept from a concept group [Fal19]. However, it was noted by Falke [Fal19] that in current related work, other methods to select the labels, like NLP techniques, were not reported.

Dependency Parsing reveals the structure of a sentence and the type of connection between its words. It can be used in "Concept Mention Extraction" and "Relation Mention Extraction" to detect possible concepts and relations based on their dependency modifier. Furthermore, the connections between the words enable the identification of compound noun and verb phrases. The modifiers are additionally helpful in the "Concept Map construction" step to construct *subject-verb-object* triples as propositions. Valerio and Leake [VL06] used dependency parse trees to extract linking phrases based on the pairs of concepts that are linked through a verb phrase. Afterwards, they constructed concept maps based on the found *subject-verb-object* triples. Quasim et al. [QJHL13] employed the dependency parsing of the Stanford parser to select candidate terms for possible concepts, based on extracted dependencies (*amod*, *nn*, *conj and*, *prep of*, *nsubj*).

Named Entity Recognition (NER) is useful to find concepts in the "Concept Mention Extraction" step or provide additional information for concepts that could be used in the "Concept Labeling" step. Zubrinic et al. [ZOS15] stated that they used NER for the Croatian concept maps, but did not report how the technique contributed to their CMM

process. Aguiar and Cury [AC16] did not use NER to identify concepts directly, but retained each sentence where the same entity was identified as a textual summary. They subsequently queried DBPedia, which represents the information contained in Wikipedia in a semantic web format, with the entity label and type to extract the description for the found entity. Then, a cosine similarity of the Wikipedia description and the textual summary of the entity was computed. If the similarity was high, the description was associated with the concept, else with the entity type.

Coreference or Anaphora Resolution enables to identify multiple variations of the same concept across sentence boundaries. It can therefore be applied in the steps "Concept Mention Grouping" and "Relation Mention Grouping" to find additional instances of the same concept and gather additional relations. Quasim et al. [QJHL13] conducted anaphora resolution not only for nouns but also for pronouns. For this purpose, they utilized the RAP algorithm [LL94] to extract more relations and generated more complex propositions than would be possible with only using noun phrases. Zubrinic et al. [ZOS15] used limited coreference resolution. The algorithm they employed relied on a definition of terms based on introductory sections of documents. Whenever such a term was encountered later in the document, it was associated with the defined term in the introduction. Aguiar and Cury [AC16] also incorporated anaphora resolution in their CMM process to group concepts that were represented as pronouns. However, they reported problems with resolving demonstrative, possessive, and personal pronouns in respect to the first person. As an example, they mentioned the proposition *we-define-concept* that was extracted, but did not contribute to the understanding of the topic in their concept map. Hence, personal pronouns of the first person were ignored for the construction of their concept maps.

Many of the described NLP techniques for text processing are also heavily employed in this thesis' approach for semi-automatic CMM. Which pipeline steps are used and how they are applied for the semi-automatic CMM, is explained in Chapter 4.

3.2 Manual Concept Map Creation

Manual Concept Map Creation was the initial way to build concept maps, before computational text pre-processing tools were applied and (semi-)automatic CMM processes were invented. The first tools to create concepts manually were simply pen and paper or post-its on a whiteboard or paper roll to freely move concepts around the drawing space [NC06]. To conduct the manual creation of a concept map, Novak and Cañas [NC06] set up a sequence of best practices for *constructing good concept maps*:

1. Determine a knowledge domain the concept map will be created for to establish a context that helps to create the hierarchical structure of the map. The context can be a text, laboratory or field activity, or a problem that the creator wants to understand. If someone is still learning how to create a concept map, a context should be used, which the creator is familiar with. Stating a specific focus question

is a good method to set the context for a concept map. This helps to prevent the deviation from the topic at hand while building a map, because it is always clear which question the concept map has to address.

2. Identify 15 to 25 concepts that may be used in the concept map. This can first be done by creating a listing of the most relevant concepts of the source text. Afterwards, this list can be ordered by ranking the most general concepts at the top and the most specific concepts at the bottom of the list. Not all of these listed concepts have to be transferred to the concept map.

3. Construct a preliminary concept map by choosing from the ranked list and create relations. Post-its or computer software for concept map construction are best suited for this step, because they allow the easy re-arrangement of concepts.

4. Cross-links between different segments of the domain should then be added, after the preliminary concept map was constructed. This is an important step, which shows that the creator of the map understands relations between different sub-domains of the topic. Thereby, multiple words or sentences as concepts should be avoided, as well as "String maps" that are laid out like chains, because it shows poor understanding of the topic.

5. The creation of up to three revisions of the preliminary concept map should be conducted as final steps, which is seen as a necessity to construct good concept maps. With these iterations, errors can be detected and the structure and topology of the map can be enhanced until a final revision of the map can be saved.

A state diagram of this best practice workflow was created and can be seen in Figure 3.3.

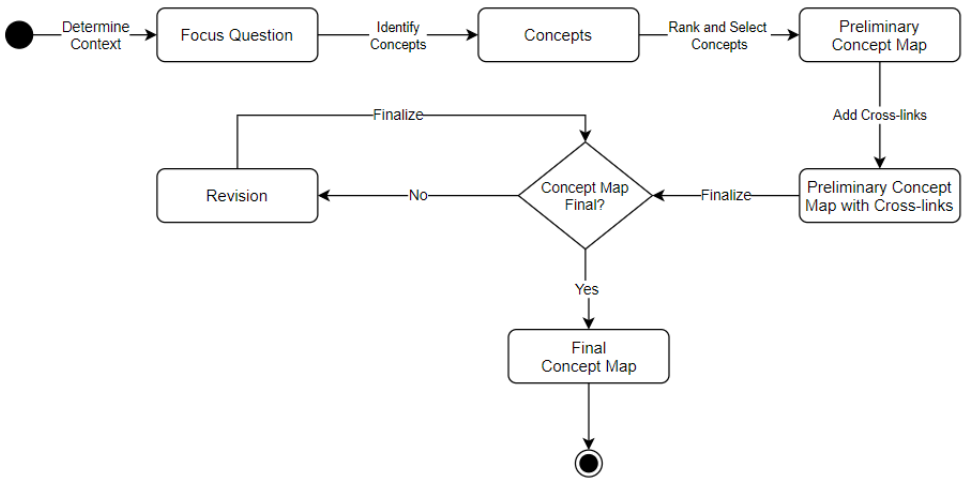

Figure 3.3: State diagram of how to construct good concept maps.

Novak and Cañas [NC06] also suggested to use CmapTools [CHC+04] [CCH+05] as a suitable software to manually construct concept maps. CmapTools is certainly the most widely known platform to create, share, and discover concept maps and foster the culture of concept mapping. The tool was first described by Cañas et al. [CHC+04] and is advanced and refined until today by researchers at the Florida Institute for Human & Machine Cognition (IHMC).

CmapTools is a software suite that consists of a client-server infrastructure. The desktop client can be installed on all major operating systems for personal computers. It allows users to create, share, and collaborate on concept maps. A search function enables to retrieve deposited concept maps from the CmapTools cloud. The concept maps themselves can be linked to other concept maps, contain sub-concept maps, and link to additional resources, like web pages, documents, images and video, or audio files. Additionally, concept maps can be extended by searching for existing concept maps based on created concept maps in the CmapTools cloud as well as resources from the web. A concept map created and shared in CmapTools by NASA about the integration of launch vehicle systems for space flight can be seen in Figure 3.4.

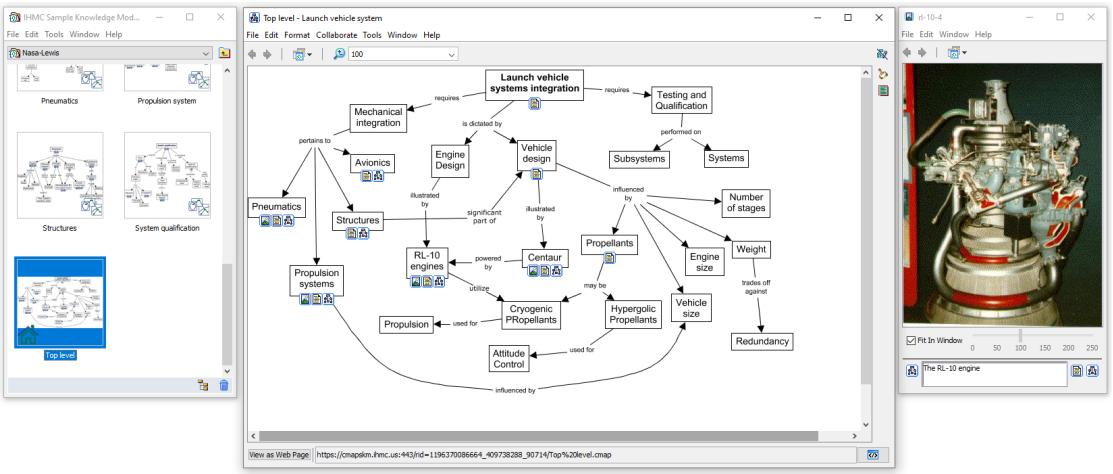

Figure 3.4: Screenshot of a concept map created by NASA about launch vehicle systems integration in the CmapTools software suite [CHC+04].

CmapTools is still extended with additional features and updates. Cañas et al. [CCL18] presented a new component for the CmapTools software suite called “eCmap” in 2018. “eCmap” is a concept map editor that can be embedded in web pages. It was developed in response to making the functionality of CmapTools more portable and available. The editor can be embedded in HTML pages and exposes a JavaScript API that enables the manipulation and control of editor functions. All concept maps created with “eCmap” and CmapTools can be retrieved and saved from the CmapTools server. The concept maps can be embedded with all editor functions activated or in read-only mode, only granting the viewing of the concept maps.

Besides CmapTools, there is a variety of online graph creation and manipulation tools that enable the quick creation of concept maps. These tools were not specifically developed for concept map creation, but provide all necessary features. MindMup [MIN], MindMeister [MEI], Mindomo [DOM] and Creately [CRE] are examples of such applications.

3.3 Fully Automatic Concept Map Mining

As explained before, fully automatic CMM techniques should not require user intervention during the CMM process. The task to fully automatically create concept maps without user intervention is certainly the most researched approach in the related work. Based on the author's research, many of the relevant papers on this approach were published in the period of 2006 to 2016 (Paweł et al. [MDP06], Villalon and Calvo [VC08], Villalon and Calvo [VC09], Zubrinic et al. [ZKM12], Quasim et al. [QJHL13], Zubrinic et al. [ZOS15] and Lee et al. [LPY15]). One of the latest papers on automatic concept map construction in the bi-annually held International Conference on Concept Mapping was published by Aguiar and Cury [AC16] in 2016. No work on this approach was included in the latest International Conference on Concept Mapping in 2018. However, novel techniques for fully automatic CMM were presented by Falke [Fal19] in his dissertation in 2019. This section is intended to give only a brief overview of selected state-of-the-art work in this area, since the focus in this thesis lies on semi-automatic approaches.

3.3.1 Using a Domain Thesaurus for Creating Concept Maps from Unstructured Texts in Croatian

Zubrinic et al. [ZOS15] proposed a fully automatic CMM approach that conducts concept mapping from multiple legal documents in the Croatian language (Cl). The CMM process that Zubrinic et al. [ZOS15] used for this approach can be seen in Figure 3.5.

The first step in their process in Figure 3.5 concerns "Preparation and linguistic preprocessing". A set of documents is extracted and cleaned from formatting characters. Afterwards, an NLP pipeline that employs special Croatian implementations of tokenization, sentence segmentation, POS tagging, lemmatization, dependency parsing, named entity recognition, and a simple form of anaphora resolution, is used to process each text as a preparation for concept extraction. For the "Concept extraction" step in Figure 3.5, Zubrinic et al. [ZOS15] used a Croatian domain thesaurus containing terms that are connected with a set of relations: equivalence, hierarchy and association. Concepts are selected with a modified TF-IDF measure, CF-IDF that calculates the weights of the concepts. Labels for concepts are selected by using words that could be extracted in the linguistic pre-processing. The words are connected with the domain thesaurus by the equivalence relation or, if not found in the thesaurus, with a Croatian version of WordNet. Non-hierarchical relations are extracted from parsed dependencies by searching for *subject-verb-object* triples. Hierarchical relations are extracted by using the hierarchical relationships. These relations, along with extracted concepts, are used to form propositions. A concept map is constructed by organizing the propositions in a

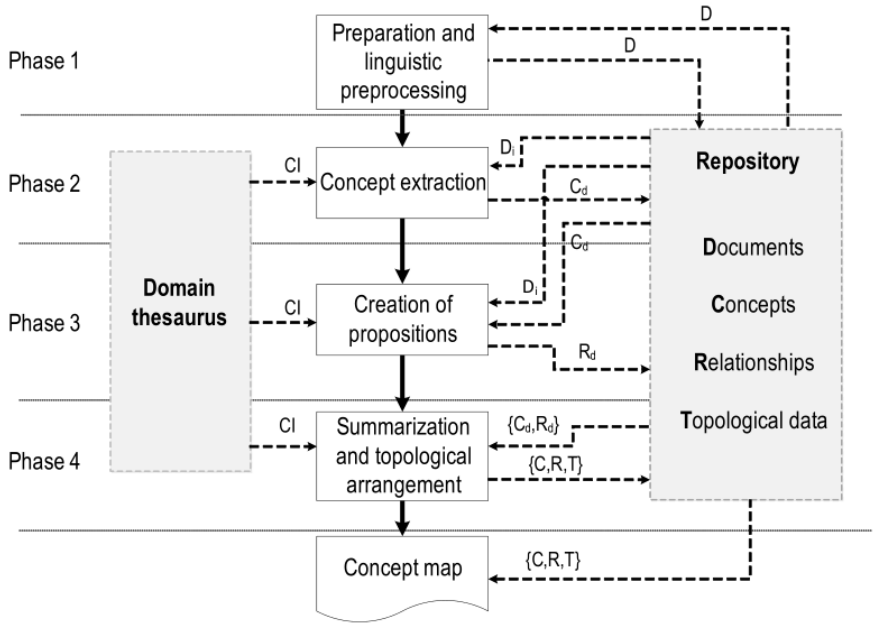

Figure 3.5: CMM process for multiple documents in the Croatian language as illustrated by Zubrinic et al. [ZOS15].

tree structure. Multiple relations are cleaned by removing relations without labels and relations with the same labels and same concepts. Concepts not connected to the map are removed as well. The topology of the map is created by using the issuer and title of each document as root concepts, using hierarchical relations and the concept weights. Finally, the concept map is trimmed by discarding concepts with the lowest weights on the outermost leaves until only up to 30 concepts remain. A file for each concept map is created for visualization purposes.

Zubrinic et al. [ZOS15] validated the technique by letting a set of 538 users grade their created concept maps in comparison to gold standard concept maps on a scale of 1 (lowest score) to 5 (highest score) that were created by experts. The features concept, relations, label of relations, hierarchy, usefulness, and summary were graded. Results show that the summary of the created concept maps had a median rating of three providing a general good summary of the documents. Extracted concepts had the best median grading of four, whereas all other features received a lower score of three.

3.3.2 State of the Art Pipeline for Automatic CMM from Texts

Aguiar and Cury [AC16] proposed an automatic CMM approach that is based only on a single data source of an unstructured text in the domain of academic articles in English. The approach utilizes the Stanford CoreNLP [MSB+14] framework together with their own algorithms. The overall process from text preparation to the summarized propositions was illustrated by Aguiar and Cury [AC16] and can be seen in Figure 3.6.

In the "Preparation" step of the process in Figure 3.6, prepositions are replaced with predefined linking phrases (e.g., "between" is replaced by "appear between"). The "Normalization" step includes the removal of special characters, anaphora resolution, and removing sentences that do not form propositions. Tokenization and POS tagging, along with sentence segmentation and dependency parsing are computed in parallel. Candidate structures for propositions are extracted from the resulting dependency parse tree with pattern matching. Concepts and relations are then formed from the candidate structures by applying an algorithm that uses morphological rules for identification. For instance, a concept is identified when the candidate structure contains a proper noun (NNP) that has an adjective (JJ). Labels for concepts and relations are determined by using a combination of identifying multi-word phrases, resolving similar concepts with a similarity measure through WordNet [Mil95], the replacement of prepositions as explained in the "Preparation" step, and additional information like entity types or labels provided by named entities through a query to DBPedia as explained in Section 3.1.2. The last steps in Figure 3.6, "Ranking" and "Summarization" are done with a form of term-frequency weights-calculation for concepts that also considers relations. This weight was developed by Leake et al. [LMR04] and is called HARD model. Finally, the generated propositions are displayed as a concept map in a user interface.

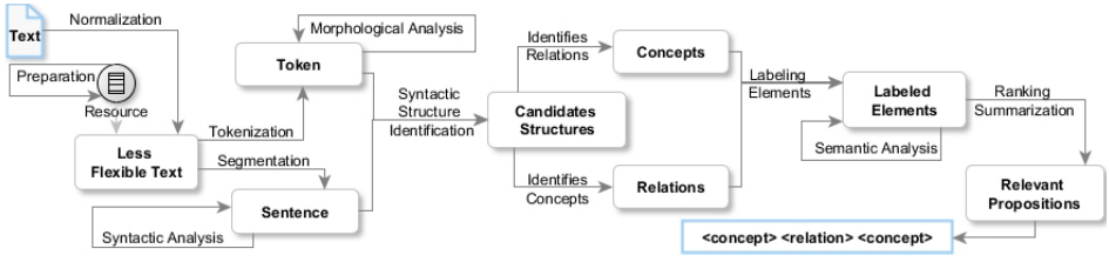

Figure 3.6: State-of-the-art CMM process as illustrated by Aguiar and Cury [AC16].

Aguiar and Cury [AC16] conducted an evaluation by running their approach on a text about concept maps and manually comparing their created map against ten maps generated by experts. They explicitly mentioned the visual quality of the created map and highlighted some of the following features: all concepts have relations, no fragments are present, no pronouns or named entities are used as concepts, concepts are small and meaningful, concepts have multi-word expressions only if necessary, relation labels are only meaningful verbs, propositions are not redundant, and the map represents the mined text.

However, comparison of their map against the expert maps yielded concept precision of 0.47 and recall of 0.67. Relation precision was 0.29 and recall 0.44. On the one hand, they attributed these scores to the fact that the experts found it difficult to create the concept map from text and used only the concepts directly present in the text. On the other hand, they experienced problems with demonstrative and possessive pronouns of the first person and relation labels that often did not correspond with relation labels

used by the experts. Aguiar and Cury [AC16] believed that the scores could benefit from the use of a domain thesaurus that will be constructed iteratively when new texts from the domain are processed.

3.3.3 Novel Approaches for CMM Sub-Tasks Using Predicate-Argument Analysis and Neural Networks

At last, there is still ongoing research in the field of automatic CMM. Falke [Fal19] proposed novel approaches for several tasks in automatic CMM that make use of current popular techniques in machine learning like neural networks.

Falke [Fal19] introduced the usage of a technique called predicate-argument analysis for concept and relation extraction, which he proved can be ported from English to German without much effort. This technique focuses on predicate-argument structures instead of more complex syntactic structures like dependency trees retrieved by dependency parsing. The syntactic structures are basically mapped to simpler predicate-argument structures with the techniques semantic role labeling and rule-based converters. Falke [Fal19] found that predicate-argument analysis improved the quality of extracted relations.

The sub-task "Concept Mention Grouping" in Figure 3.2 was enhanced by using pairwise classifications and set partitioning. Overly simplified, the classifier trains pairwise concepts on their probability of being coreferer. The classifier was trained with training sets that were composed of derived binary coreference labels from text corpora. The classified binary probabilities are subsequently used to create partitions and group the concept mentions. Pairwise classification was found to be helpful for "Concept Mention Grouping", because more types of coreferences could be extracted in comparison to previous work. However, the set partitioning approach suffered from scalability issues, especially if large corpora are used.

Finally, these improved sub-tasks were combined by Falke [Fal19] to a pipeline, which he also used to model the problem of summarizing concept maps as an end-to-end classification task employing neural networks. However, this classification task is currently not found to be competitive to other state-of-the-art techniques, which was mostly attributed to the available training data.

3.4 Semi-Automatic Concept Map Mining

To the author's knowledge, there is no standardized way, process or method to semi-automatically conduct concept mapping. Existing work on supporting users in constructing concept maps with semi-automatic approaches is scarce. Some semi-automatic approaches that require user-intervention, but were not explicitly designed to be semi-automatic, were listed by Kowata et al. [KCB10]. Since we are interested in works that were intentionally designed to be semi-automatic CMM methods, we will focus on some of these approaches.

Three approaches have been identified that have an explicit semi-automatic design: Textstorm and Clouds by Alves et al. [OPC01], a recommender system for the visualization of learning progress by Mirbagheri et al. [MHK19], and a CMM process that is supported with concept and relation suggestions by Liu et al. [LLZ13]. These approaches will be highlighted in the following subsections. The approach by Liu et al. [LLZ13] is the most similar to the system presented in this thesis.

3.4.1 Infering Concepts and Relations with Textstorm and Clouds

One of the first and most referenced work for semi-automatic CMM is a system that is composed of two modules, i.e., Textstorm and Clouds, which was created by Alves et al. [OPC01]. The purpose of the first module, Textstorm, is to process textual data and extract binary predicates. As an example, we consider the sentence "*The fox is an animal*". Textstorm would then extract the binary predicate "isa(fox, animal)". This is achieved by utilizing POS tagging with WordNet [Mil95]. The main verb of the sentence, is used as the predicate, the noun or subject before the verb serves as the first concept and the noun or object after the verb servers as the second concept in the binary predicate. Ambiguous concepts of the same instance in different sentences are resolved with a form of anaphora resolution based on history lists and case-based reasoning. At the end of Textstorm's processing, we have resulting binary predicates, which themselves could theoretically be used as a limited concept map.

However, the intention of Textstorm is to serve a base set of binary predicates for Clouds that uses these binary predicates as inputs for inductive learning algorithms. The goal of these algorithms is to infer new concepts and relations by interactively asking the user for new knowledge to construct a more refined concept map. To start the learning, Clouds has an initial set of primitive "is a" relations that form a tree. Then, Clouds reads the predicates generated by Textstorm and treats them as if they were created by a user. Inductive learning algorithms will then start the inference by asking questions to the user and utilize the resulting answers for the creation of new predicates. To emphasize how Clouds works in conjunction with Textstrom, we consider the following simple text example: "*Some animals have different living spaces than others. For instance, a fox lives in the wood. A whale lives in the sea.*" Textstorm would extract the following predicates:

have(animal, living_space); live(fox, wood); live(whale, sea)

Taking these predicates, Clouds would proceed to ask some of the following questions, which the user would have to answer with examples of predicates and boolean values:

Clouds: Define animal with the predicate "isa".
User: isa(animal, living_entity).
Clouds: Define foxes with the predicate "isa".
User: isa(fox, animal).
Clouds: Define whales with the predicate "isa".
User: isa(whale, animal).
Clouds: Define living_spaces with the predicate "isa".

User: isa(wood, living_space).
Clouds: Complete the relation live(whale, living_space).
User: live(whale, sea).
Clouds: Is it true that live(fox, sea)?
User: n.

From these answers, the inductive learning algorithms in Clouds could then infer the new predicate “not(live(fox, sea))”. Together with the user input and the extracted predicates by Textstorm, this inferred predicate would result in the concept map that can be seen in Figure 3.7.

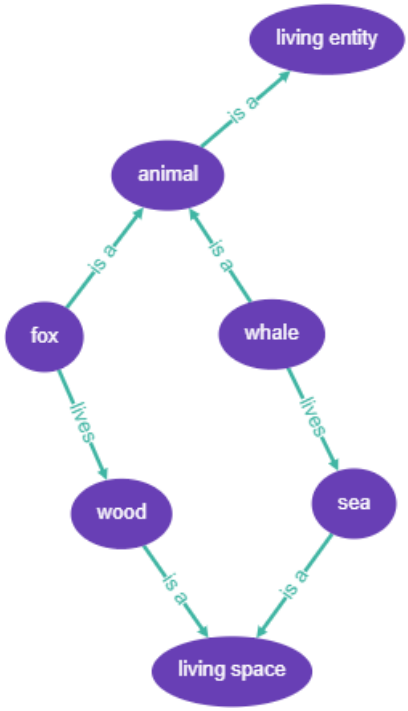

Figure 3.7: Potential concept map that could be the outcome of a semi-automatic CMM approach with Textstorm and Clouds.

Besides such simple inferences like negations in the given example, Clouds is also able to create much more powerful deductions like generalizations, specializations, or universal quantification [POC00]. However, Textstorm only reached a mean of 52% of correctly extracted predicates, which suggests that much additional effort is necessary in answering questions in Clouds [OPC01]. Furthermore, many questions have to be answered and relations need to be entered in order to construct the concept map. The number of these questions and necessary user input also increases if the number of concepts and relations grows in a larger text document.

3.4.2 Semi-Automatic Concept Maps in a Recommender System to Visualize the Learning Progress of Students

Besides the task of constructing concept maps directly, they can be used as an output visualization for depicting learning progress about a certain topic. Mirbagheri et al. [MHK19] proposed such a system, which rebuilds a predefined concept map in a semi-automatic fashion based on user feedback and displays the users' learning progress.

Initially, a concept map is built in CmapTools by experts about a specific topic or lesson, which is used as the base concept map that depicts the students' learning status. At the start of the learning process, a student can select five learning resources that are each connected to different concepts in the map. An additional five resources are recommended by the system, which contain concepts that were not in the selected resources by the student. This assures that all concepts in the map are covered by the learning resources. After reading a resource, the student can leave a score for a resource that must be inside a range predefined by experts. This score is then used to provide the five additional resources for other users of the system. When the student is finished reading all resources, a test must be taken, which comprises questions related to the learned concepts. This relation is captured in a Test Item Relationship Table (TIRT), which shows the relation of a question to a concept on a scale from 0 to 5. With TIRT and the answers to the questions, a probability $P(c_i)$ is calculated that shows the failure probability for a concept, which gets mapped to a color and subsequently the learning status of a student for a concept. The mapping of probabilities $P(c_i)$ to colors and learning states in the concept maps can be seen in Table 3.1.

Color of c_i	**$P(c_i)$**	**Learning Status**
Red	$0.75 < P(c_i) \leq 1$	It seems that you misunderstood the concept
Orange	$0.5 < P(c_i) \leq 0.75$	It seems that you partially misunderstood the concept
Yellow	$0.25 < P(c_i) \leq 0.5$	You have learned the concept well
Green	$0 \leq P(c_i) \leq 0.25$	You have learned the concept very well

Table 3.1: Mapping of learning status of concepts to colors in the concept map, adapted from Mirbagheri et al. [MHK19].

The system then recommends learning resources to the student that are related to concepts, which were not understood by the student and for which more learning is necessary. Students can retake the tests up to three times until all concepts in the map have been understood correctly, which means that all concepts have a failure probability under 25% and all concepts in the concept map are colored green. This learning-progress visualization can be seen in Figure 3.8 that shows a concept map with misunderstood concepts, which are subsequently colored green when the student has completed the learning.

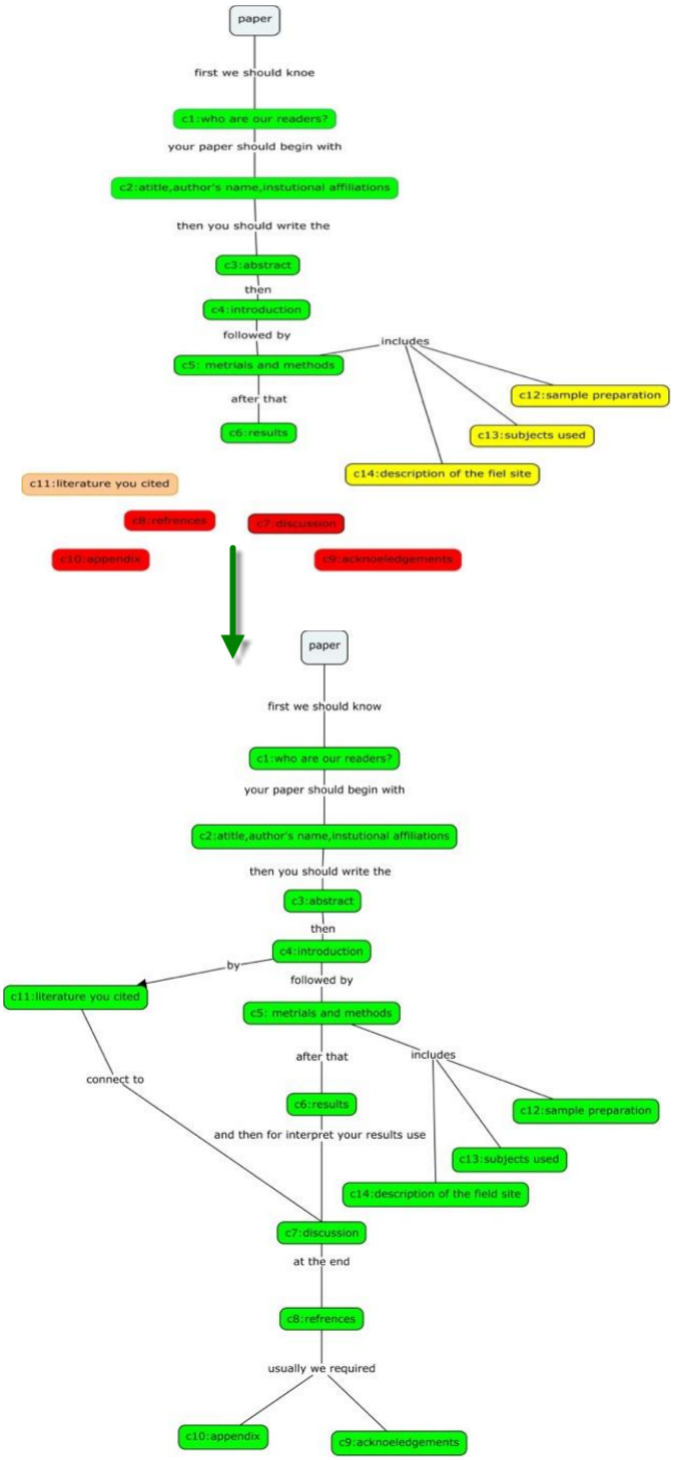

Figure 3.8: Semi-automatically created concept maps showing the learning progress of a student as depicted by Mirbagheri et al. [MHK19].

A user study was conducted by Mirbagheri et al. [MHK19] that included two groups of users, who took three tests to evaluate the proposed method. The first group did not use the recommender system, which made sure that all relevant concepts are covered in their learning material. The second group, however, used the semi-automatically constructed concept maps with the recommended and concept-linked learning material. Results in their user study show that the second group had significantly higher scores on the tests than the first group. The proposed method is an interesting way to utilize semi-automatic concept map construction and support students in learning extensive topics, by employing concept maps as a tool for knowledge visualization. Mirbagheri et al. [MHK19] also argued that this visual feedback provides more motivation for students that are learning, because seeing their learning progress and iteratively completing a concept map instead of numerical feedback is more satisfying.

3.4.3 Automatic Concept and Relation Suggestions

Another option for semi-automatic CMM is to support the user during the actual process of creating concept maps from text. Liu et al. [LLZ13] presented such an approach for Chinese texts, where the user receives suggestions for new concepts and relations directly in the drawing area of the concept map. The procedure they used to conduct CMM consists of the steps "Key Term Extraction", "Term Association Analysis", and "Automatic Recommendation". It uses multiple documents as a source. The statistic algorithm they used for "Key Term Extraction" exploits the fact that key terms are repeated often over many documents and have therefore a high word frequency. At first they apply POS tagging, filter out stop words into a stop word list, and transfer the resulting terms into an ordered word frequency list. Additionally, word frequency in the key term list is updated with terms that are associated with synonym and acronym lists, since different authors might use diverse terms for the same topic over multiple documents. To create relations between the key terms, Liu et al. [LLZ13] conducted the "Term Association Analysis" with an adapted cosine similarity weight that resembles the correlation between two terms. From the resulting key terms and association weights, a global CM over all documents is then extracted, which serves as a basis for a local CM. This local CM is subsequently used as a source of suggestions for a new text and the actual concept map that is created separately in the following "Automatic Recommendation" step.

Before the concept map can be created semi-automatically in the user interface, a copy is made of the global CM that later on becomes the local CM. At this point, a new text is entered in the user interface "Text input area", which can be seen in Figure 3.9.

A requirement for this approach to work is that the entered text covers the same underlying topic. Acronyms and synonyms that are not present in the current local CM, but occur in the text, are added to the local CM along with their present relations. Additionally, concepts and relations that never occur in the document are removed from the local CM. This local CM now serves as a basis for concept and relation suggestions to the user. In the drawing area, the user can start to construct a concept map by creating

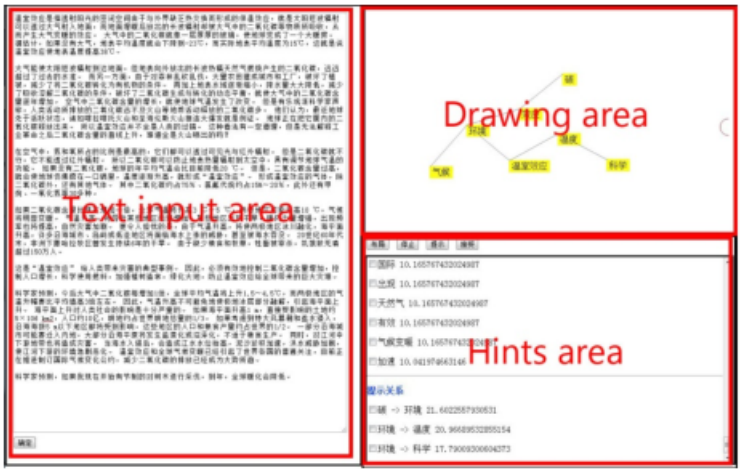

Figure 3.9: User interface of the semi-automatic CMM tool by Liu et al. [LLZ13].

concepts and relations. If the user now clicks a created concept in the drawing area and clicks the "Hints" button, concepts and relations that are associated with the clicked concept through the local CM are displayed in the "Hints area". Now the user can select suggestions from the "Hints area" by clicking a checkbox, which adds the selected concept suggestions or relations to the concept map in the "Drawing area".

Liu et al. [LLZ13] measured average acceptance rates of 50% for the suggested concepts and 40% for the relations in a user evaluation with a shorter and a longer text. They mentioned that the shorter text had a higher acceptance rate. Furthermore, Liu et al. [LLZ13] reported that the acceptance rate highly depended on the selected article at hand, because it had to match the topic of the global CM.

As already mentioned in the introduction of this subsection, the approach by Liu et al. [LLZ13] is the most similar to the one presented in this thesis. However, there are certain core aspects, the approach in this thesis does differently or tries to improve:

1. The presented CMM approach in this thesis does not require multiple documents to build a predefined global CM. Instead, a single-source text-document is processed from which suggestions for concepts and relations are extracted.
2. Suggestions for concepts and relations can be seen directly in the canvas for the concept map creation in the thesis' approach, whereas the suggestions in the work by Liu et al. [LLZ13] can be seen in the "Hints area".
3. Liu et al. [LLZ13] did not take labels for relations into account in their CMM approach. Labels for suggested relations are supported in the presented work.
4. In this thesis' approach, the concept map creation can initially be started by selecting concept suggestions. This is not possible with the approach presented by Liu et al. [LLZ13].

CHAPTER 4

A Semi-Automatic Concept Map Mining Approach

The goal of the semi-automatic CMM in this thesis is to realize a system that allows users to create a visual summarization of unstructured textual data in the form of a concept map. The system is aimed to provide a semi-automatic supporting system that enables the user to fulfill this task in an efficient way. The users should thereby be able to create their subjective version of a concept map. This approach stands in contrast to generating concept maps for a variety of texts for the same focus question by different users, as would be the application example for a fully automatic approach. Additionally, the semi-automatic approach needs to offer meaningful concept and relation suggestions to the user, which must facilitate the process of CMM and make it more efficient in contrast to a fully manual approach.

In order to address these requirements, we propose a semi-automatic CMM approach that provides the following features:

1. A state-of-the-art NLP pipeline with a suitable algorithm to mine the unstructured text and extract meaningful concepts and relations as propositions, which can subsequently be used as suggestions for the concept map.

2. An interactive user interface to present the extracted suggestions of concepts and relations to the user and provide means for the construction of the concept map, while minimizing visual clutter and gaining high recall.

3. An implementation as a web application in the browser, where a large part of the information work is conducted today.

4.1 Formal Problem Definition

The goal of the NLP pipeline is not to extract only a valid or a generalized set of concepts and relations. The user should rather be able to interactively choose a concept or relation from a variety of proposed concept and relation suggestions in the user interface. The following definition is derived from the CMM process defined by Villalon and Calvo [VC08] in Section 3.1.

Formalizing the task of the concept and relation extraction, we need to mine a text document D, where CD are all concepts and RD are all relations between these concepts in the document. From all available concepts CD and relations RD, we want to extract a set of propositions P that are composed of concept suggestions CS and relation suggestions RS. The extracted concept suggestions CS are a subset of all concepts in the document CD. The extracted relation suggestions RS are a subset of all relations in the document RD. The user is able to construct a concept map $CM = \{C, R\}$ from these extracted concept and relation suggestions in the user interface. The final CM contains concepts C and relations R, where C is a subset of the concept suggestions CS, and R a subset of relation suggestions RS.

$$D = \{CD, RD\}. \tag{4.1}$$

$$P = \{CS, RS\}. \tag{4.2}$$

$$CM = \{C, R\}. \tag{4.3}$$

$$C \subseteq CS \subseteq CD. \tag{4.4}$$

$$R \subseteq RS \subseteq RD. \tag{4.5}$$

For the task of creating a concept map manually in the user interface, we also want the user to be able to build a concept map $CM = \{C, R, G\}$ with concepts C, relations R, and generalization levels G. However, the generalization levels are optional and the emergence of a hierarchy is up to the user, since no generalization levels are predetermined by the system. Hence, we consider a concept map $CM = \{C, R\}$ as an equally valid final result of the process.

As a first step in the CMM process, the system has to provide a set of concept suggestions CS to the user when the concept map is still empty ($CM = \{C = \emptyset, R = \emptyset\}$). This set of concepts contains all the extracted concept suggestions CS that were retrieved from the NLP pipeline and its subsequent algorithms: $CS = \{c_1, c_2, ..., c_s\}$.

Relations between existing concepts in the concept map should be suggested as well. At first, we define the concepts in the concept map as the set $C = (c_1, c_2, ..., c_n)$. Let $c_i, c_j \in C; c_i \neq c_j; 1 \leq i \leq n; 1 \leq j \leq n$, be two concepts in the concept map C that have been chosen from the concept suggestions CS. Additionally, let $c_p, c_q \in CS; c_p \neq c_q; 1 \leq$

$p \leq s; 1 \leq q \leq s$, be the two concepts from the concept suggestions CS, where $c_i = c_p$ and $c_j = c_q$. If there is an extracted proposition that contains the relation between these two concepts, the system should suggest a relation $r_k \in RS = (c_p, c_q, l_k)$ from the set: $RS = \{r_1, r_2, ..., r_m\}; 1 \leq k \leq m$. The relation r_k has the label l_k.

We also want the user to be able to add concepts that are not yet part of the concept map, but which can be linked to one of the currently present concepts in the map. If a concept $c_i \in C$ is selected from the concept suggestions CS in the concept map $CM = \{C, R\}$, all extracted concept suggestions $c_q \in CS$ that are not in the concept map CM and have a possible relation $r_k \in RS = (c_p, c_q, l_k)$ with the concept c_i, where $c_i = c_p$ should be suggested as well.

4.2 The Semi-Automatic Concept Map Mining Process

The approach for conducting the semi-automatic CMM is based on state-of-the-art research in CMM. The whole process is illustrated in a diagram in Figure 4.1. There are two main steps in the process that form the semi-automatic nature of the approach:

Automatic Text Processing that extracts propositions in the form of concepts and relations from a single unstructured text. This step and its sub-steps emerged from studying the CMM processes by Zubrinic et al. [ZOS15] and Aguiar and Cury [AC16]. The naming of the sub-steps is derived from the sub-steps as proposed by Falke [Fal19]. This first task and its sub-steps is explained in more detail in Section 4.3.

Manual Concept Map Construction takes the propositions that were extracted in the first step and offers them as concept and relation suggestions to the user. These suggestions can be used during the manual construction of the concept map. This approach can somewhat be related to the suggestions by Liu et al. [LLZ13], with the difference that there is only one source document needed to provide these suggestions and further advances of the state-of-the-art that were explained in Section 3.4.3. The procedure for the second task, along with the user-interaction, is shown in Section 4.4.

Additionally, a prototype web-application was created that implements this proposed semi-automatic CMM approach. Implementation details, such as system architecture, used NLP framework and UI libraries, are listed in Section 4.5. This prototype was subsequently used for the user evaluation of this semi-automatic CMM approach in Chapter 5. Section 4.6 gives insight into the creation of two example concept maps with the prototype implementation of the semi-automatic CMM approach.

We will use the following example text to explain the semi-automatic CMM process:

"*The quick brown fox jumped over the lazy dog. Then it ran across the orchard. The lumberjack had known the fox for some time.*"

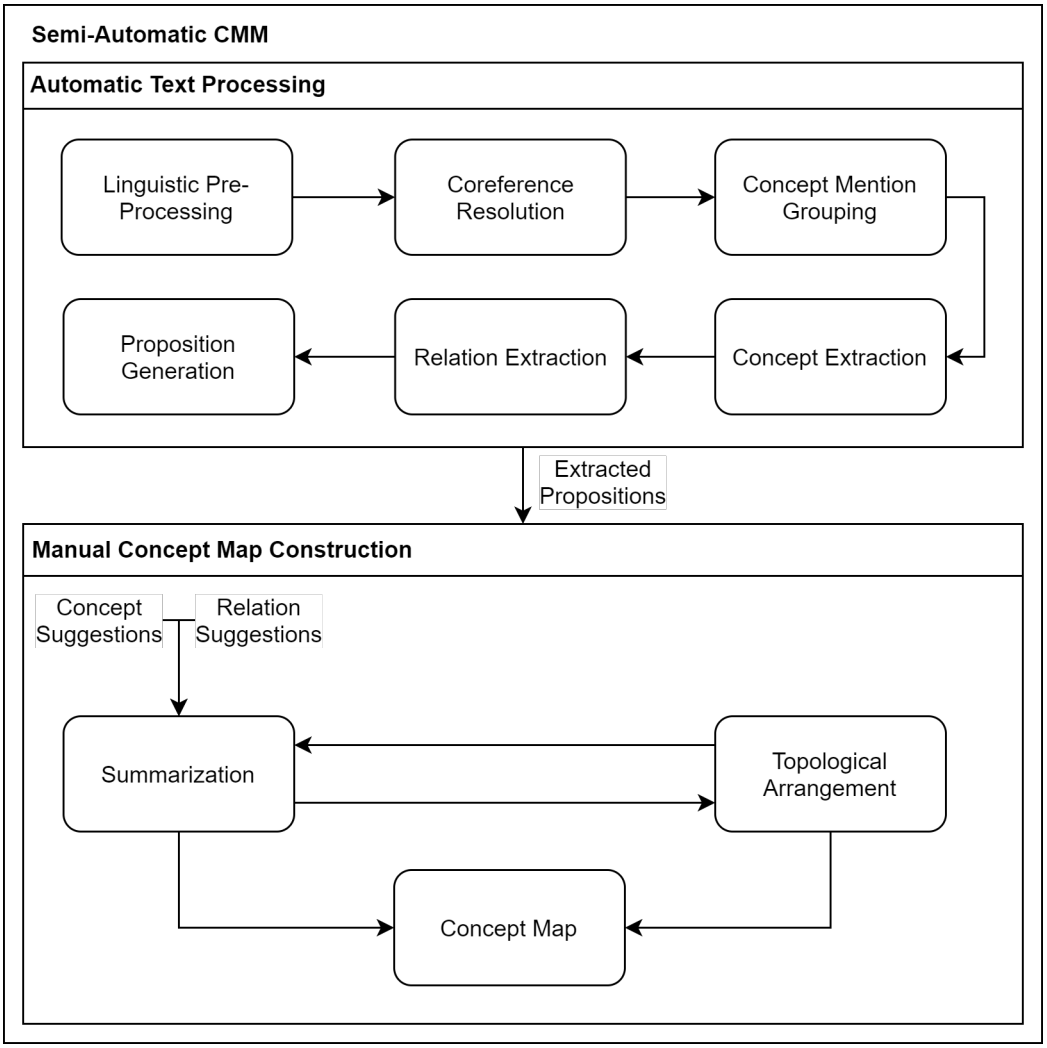

Figure 4.1: Process for the semi-automatic concept map mining approach.

4.3 Automatic Text Processing

The "Automatic Text Processing" step expects an unstructured text in a cleaned format (i.e., without any special characters such as HTML tags etc.) as an input. Subsequently, the supplied text will be processed in a pipeline of additional sub-steps. These sub-steps consist of an initial "Linguistic Pre-Processing" (Section 4.3.1) step that conducts relevant steps of a typical NLP pipeline to annotate the submitted text document. Additionally, "Coreference Resolution" (Section 4.3.2) is applied on the annotated text to resolve anaphora. Afterwards, the concept mentions are grouped with the found coreference mentions in the "Concept Mention Grouping" (Section 4.3.3) step. From the grouped mentions, "Concept Extraction" (Section 4.3.4) tries to extract noun phrases and named

entities as concepts from the annotated text. In the last sub-steps "Relation Extraction" and "Proposition Generation" (Section 4.3.5), relations are extracted and propositions are created, which is based on the identified concepts. The extracted propositions form the result of the whole "Automatic Text Processing" step (Section 4.3.6). A simple sequence of assignments for this pipeline can be seen in Algorithm 4.1. This algorithm glues the sub-steps together and defines their execution order.

Algorithm 4.1: Automatic Text Processing

```
1 ProcessText (text)
      input  : An unstructured, cleaned text string
      output : Propositions with concepts and relations
2     annotatedText = LinguisticPreProcessing(text);
3     coreferenceClusters = CoreferenceResolution(annotatedText);
4     conceptMentionGroups = GroupConceptMentions(annotatedText,
        coreferenceClusters);
5     concepts = ExtractConcepts(conceptMentionGroups, annotatedText);
6     relations = ExtractRelationsAndCreatePropositions(concepts, annotatedText);
7     return Propositions(concepts, relations);
```

4.3.1 Linguistic Pre-Processing

The "Linguistic Pre-Processing" step can be conducted with any of the available state-of-the-art NLP frameworks, such as Stanford CoreNLP [MSB+14], NLTK [BKL09] or spaCy [SPA]. Figure 4.2 shows the debug output of the sample text that was processed with the NLP framework spaCy [SPA].

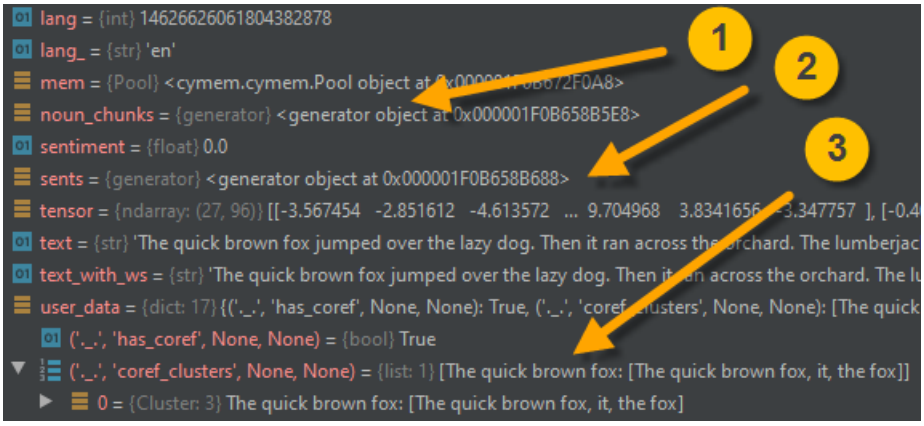

Figure 4.2: Debug output of an annotated sample text, which was processed with the NLP framework spaCy [SPA] in Python: 1) shows a generator object that enables the iteration over all noun chunks, 2) shows a generator object to iterate over all sentences, 3) shows extracted coreference clusters.

It needs to be able to process a text document, such that the most relevant NLP pipeline steps annotate the document, as explained in Section 2.2. The text needs to be tokenized and sentences segmented in order to retrieve token and sentence boundaries for concept and relation extraction. POS tags and parsed dependencies need to be available to extract noun and verb phrases. Additionally, we want to use named entities for the concept extraction, so the NLP system needs to support NER. Stemming or lemmatization was not found to be necessary for this semi-automatic CMM approach, because we want extracted concepts and relations to be as close to the original text as possible. This is important for the second manual concept map creation part of the approach.

4.3.2 Coreference Resolution

For this step, the CMM process uses a Neural Mention-Ranking Coreference Resolution model that was part of the work presented by Clark and Manning [CM16a][CM16b]. The model uses a feedforward neural network as a Mention-Pair Encoder to compute a representation for pairs of possible coreferer. In the representation $r_m(a, m)$, m is a mention of a word and a the candidate antecedent (i.e., a possible coreferer for m). The neural network receives a set of features about the candidate a and the current mention m as inputs. Some of the features that are used include word embeddings, parent dependency, POS tag, distance between mentions, and many more. A visual representation of this feedforward neural network was created by Clark and Manning [CM16b] and can be seen in Figure 4.3.

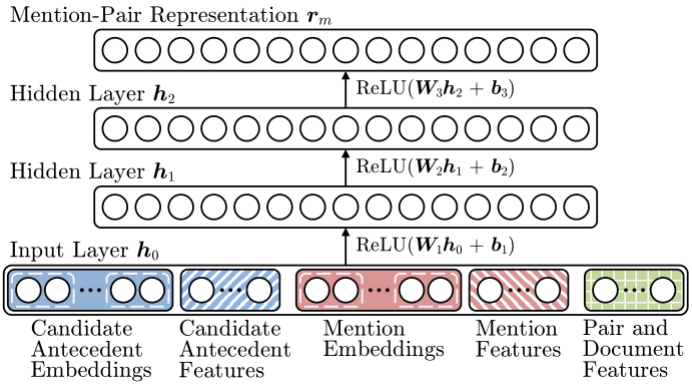

Figure 4.3: The Mention-Pair Encoder used in the Mention-Ranking model by Clark and Manning [CM16b].

The Mention-Ranking model uses this pair representation and adds a single layer to the network to compute a score $s_m(a, m) = W_m \cdot r_m(a, m) + b_m$, where W_m is a weight matrix and b_m a threshold for the layer, for each mention m and its antecedent a. This score signifies a rate for the mentions of being coreferer. The output of the Mention-Ranking model is the mention linked with its highest scoring antecedent. However, since there can be several antecedents, our used implementation of the model includes other high

ranking antecedents to become coreferer and as a result we get coreference clusters for each mention. The implementation that is used for this Mention-Ranking coreference resolution is called NeuralCoref [NEU] and is also discussed briefly in the implementation Section 4.5. The model would detect two coreferer in the sample text for the mention *it* and create the following cluster.

[*"The quick brown fox", "it", "the fox"*]

The creators of NeuralCoref [NEU] also implemented a visualization for the scores the model produces for a given text. Figure 4.4 shows a visualization with scores for the first two sentences of the example text. Thicker lines represent higher scores for coreference.

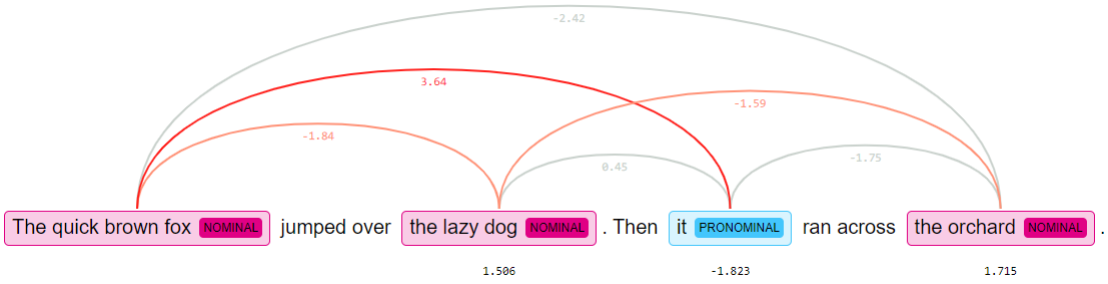

Figure 4.4: Visual representation with NeuralCoref [NEU] of the scoring for possible coreferer with the Mention-Ranking model.

4.3.3 Concept Mention Grouping

In this step, similar concepts should be grouped together and coreference clusters that have overlapping concepts will be merged. The extracted coreference clusters from the previous step already provide a basis for concept groups.

At this point, we also introduce a distinction between a *mention* of a concept and an *occurrence* of a word for the further explanation of the following "Automatic Text Processing" steps:

- A *mention* of a concept is every word in a text that refers to the semantically same concept. It can appear in a variety of forms. These are all variants of a concept that are typically contained in a coreference cluster as seen in the previous Section 4.3.2. "*The quick brown fox*", "*it*" and "*the fox*" are all *mentions* of the same concept.
- We define an *occurrence* as being the incidence of the *same* string in a text document. An *occurrence* is composed of a start and end index in the text. In the example text, the word "*The quick brown fox*" would have the occurrence: $[\{start : 0, end : 19\}]$, the word "*the fox*" the occurrence $[\{start : 103, end : 110\}]$ and "*fox*" the occurrences $[\{start : 16, end : 19\}, \{start : 107, end : 110\}]$.

Since we want to stay as close to the original text as possible, we also only want to use unaltered extracted tokens or phrases of the text for our concepts and relations. Therefore, we use noun phrases (also called noun chunks) as possible concepts. These noun phrases are composed of a noun and dependent words like adjectives that describe the noun. "*The quick brown fox*" from the previous example is such a noun phrase and each mention in a coreference cluster can be a noun phrase. To group the concepts, we initially use a coreference cluster directly as a concept mention group. Additionally, we try to merge these clusters if two coreference clusters share the same mention of a concept. Two mentions are considered as equal if the start indices of their extracted occurrences in the text match.

The example text has no coreference clusters that can be merged. However, if we imagine that the coreference resolution model would have resolved the clusters [*"The quick brown fox", "the fox"*] and [*"it", "the fox"*], they could be merged to one single cluster [*"The quick brown fox", "it", "the fox"*].

Algorithm 4.2 receives the text annotated by the NLP pipeline with the resolved coreference clusters as an input. It shows the steps to create the concept mention groups. At first, the extracted noun phrases get associated with the coreference clusters by creating a mapping of each noun phrase to a cluster and creating a unique cluster id. If a noun phrase has no associated cluster, it gets its own id and no cluster is mapped. Additionally, we map each occurrence of each noun phrase in the text to one instance of the phrase itself, if it appears more than once in the text. Pronouns like "they" can appear in many clusters and therefore we can only merge occurrences of pronouns if they belong to the same cluster. The clusters themselves are compared and if one of their mentions matches by their occurrence start index in the text, the clusters are merged to a new cluster with a new id. The algorithm returns the concepts with their occurrences, associated to a cluster with the cluster id. In the end, concepts that share the same cluster id belong to the same concept mention group. Listing 4.1 shows the result of the "Concept Mention Grouping" step with the sample text.

```
1        {"The quick brown fox", "cluster1",
2            "occurrences": [{"start": 0, "end": 19}]},
3        {"the lazy dog", "cluster2",
4            "occurrences": [{"start": 32, "end": 44}]},
5        {"it", "cluster1",
6            "occurrences": [{"start": 51, "end": 53}]},
7        {"the orchard", "cluster3",
8            "occurrences": [{"start": 65, "end": 76}]},
9        {"The lumberjack", "cluster4",
10           "occurrences": [{"start": 78, "end": 92}]},
11       {"the fox", "cluster1",
12           "occurrences": [{"start": 103, "end": 110}]},
13       {"some time", "cluster5",
14           "occurrences": [{"start": 115, "end": 124}]}
```

Listing 4.1: Result of the "Concept Mention Grouping" step.

Algorithm 4.2: Concept Mention Grouping

```
1  GroupConceptMentions (annotatedText, clusters)
       inputs : NLP annotatedText, Coreference clusters
       output : Tuples of (nounPhrase, clusterId, occurrences)
2      nounPhraseClusterMappings = new List();
3      foreach nounPhrase in annotatedText.nounPhrases do
4          foreach cluster in clusters do
5              if nounPhrase in cluster then
6                  nounPhraseClusterMappings.Add((nounPhrase, cluster.id));
7              end
8              else
9                  nounPhraseClusterMappings.Add((nounPhrase, newId()));
10             end
11         end
12     end
13     uniqueNounPhrases = new Dictionary();
14     foreach (nounPhrase, clusterId) in nounPhraseClusterMappings do
15         nounPhraseId = nounPhrase.text;
16         if IsPronoun(nounPhrase) then
17             nounPhraseId += clusterId;
18         end
19         if nounPhraseId not in uniqueNounPhrases then
20             uniqueNounPhrases[nounPhraseId] = (nounPhrase, clusterId, new
                Set(nounPhrase.occurrence));
21         end
22         else
23             occurrences = uniqueNounPhrases[nounPhraseId].occurrences;
24             occurrences.Add(nounPhrase.occurrence);
25         end
26     end
27     conceptMentionGroups = new List();
28     foreach uniqueNounPhrase in uniqueNounPhrases do
29         if HasOverlappingCluster(uniqueNounPhrase, uniqueNounPhrases) then
30             AssignNewClusterId(uniqueNounPhrase);
31         end
32         conceptMentionGroups.Add(uniqueNounPhrase);
33     end
34     return conceptMentionGroups;
```

4.3.4 Concept Extraction

In this step, we prepare the concepts that will be shown in the user interface. The base set of concepts that will be returned to the user interface are the noun phrases that were extracted and associated to a cluster id / group in the previous step. For the example text, these are the concepts that are contained in Listing 4.1.

The goal for concept extraction is *not* to choose one representative concept from each concept group as the label for the concept group. Choosing the appropriate level of detail / generality should be the user's choice in the user interface. Instead, we use these extracted noun phrases as concepts directly. A unique id is generated for each concept, in order to be able to identify it uniquely in the user interface. Furthermore, we keep the cluster ids and occurrence index locations for the text highlighting in the user interface. The information, whether a word is a pronoun or not, will be saved with a concept as well. Pronouns like "*he*", "*she*", or "*it*" should *not* be able to be chosen as concepts in the user interface. However, we also keep the pronouns and their grouping and occurrence information for the user interface's text highlighting feature.

In addition to the noun phrases, named entities that were not detected by the NLP pipeline as a noun phrase will be added as further concepts to the result for the user interface. In the example text, no named entity is present. However, we could alter the example text and add the named entity "*Monty Python*" to the last sentence as follows: "*The lumberjack Monty Phython had known the fox for some time*". Instead of the concept "*The lumberjack*", we would extract the noun phrase "*The lumberjack Monty Pyhton*" as a concept. Since we now find "*Monty Python*" as a named entity that was not marked as a noun phrase in the NLP annotated text, we can add "*Monty Python*" as an additional concept from the named entities.

How this process is applied in a programmatic manner, can be seen in Algorithm 4.3. The result of this step for the *unaltered* sample text is shown in Listing 4.2.

```
1        {"concept1", "The quick brown fox", "cluster1",
2            "occurrences": [{"start": 0, "end": 19}], false},
3        {"concept2", "the lazy dog", "cluster2",
4            "occurrences": [{"start": 32, "end": 44}], false},
5        {"concept3", "it", "cluster1",
6            "occurrences": [{"start": 51, "end": 53}], true},
7        {"concept4", "the orchard", "cluster3",
8            "occurrences": [{"start": 65, "end": 76}], false},
9        {"concept5", "The lumberjack", "cluster4",
10           "occurrences": [{"start": 78, "end": 92}], false},
11       {"concept6", "the fox", "cluster1",
12           "occurrences": [{"start": 103, "end": 110}], false},
13       {"concept7", "some time", "cluster5",
14           "occurrences": [{"start": 115, "end": 124}], false}
```

Listing 4.2: Result of the "Concept Extraction" step.

Algorithm 4.3: Concept Extraction

```
1  ExtractConcepts (conceptMentionGroups, annotatedText)
       inputs : Tuples of grouped nounPhrases, NLP annotatedText
       output : Tuples of (conceptId, conceptLabel, clusterId, occurrences, isPronoun)
2      concepts = new List();
3      foreach conceptMentionGroup in conceptMentionGroups do
4          conceptId = NewId();
5          conceptLabel = conceptMentionGroup.nounPhrase.text;
6          clusterId = conceptMentionGroup.clusterId;
7          occurrences = conceptMentionGroup.occurrences;
8          isPronoun = false;
9          if IsPronoun(conceptMentionGroup.nounPhrase) then
10             isPronoun = true;
11         end
12         concepts.Add((conceptId, conceptLabel, clusterId, occurrences, isPronoun));
13     end
14     foreach namedEntity in annotatedText.namedEntities do
15         if namedEntity.text not in concepts then
16             conceptId = NewId();
17             conceptLabel = namedEntity.text;
18             clusterId = NewId();
19             occurrences = new Set(namedEntity.occurrence);
20             isPronoun = false;
21             concepts.Add((conceptId, conceptLabel, clusterId, occurrences,
                isPronoun));
22         end
23     end
24     return concepts;
```

4.3.5 Relation Extraction and Proposition Generation

After extracting concepts in the previous steps, we now want to extract relations from the text document. With the concepts and relations, we can then build *subject-verb-object* triples, which form propositions about the analyzed text document. The concepts, relations, and propositions will then be presented in the user interface as suggestions to build the concept map.

Relation Extraction

The same strategy, which stated that we want to stay as close to the wording of the concepts in a text as possible, should be applied for the relations as well. Therefore, we want to extract the verb phrases that lie between the extracted concepts as relations. A verb phrase is a verb together with other accompanying words, like adverbs, that modify

the actual verb. If we consider the example "*The lumberjack had known the fox for some time*", the verb phrase that we can identify is "*had known*". Another example with an adverb is the following text snippet:

"*Hypo Vorarlberg subsequently announced that while they have complied with all laws in the past, they are planning to retreat completely from the offshore sector.*"

The verb "*announced*" is modified by the adverb "*subsequently*" to indicate that the announcement happened "*after*" some event. The verb phrases are extracted by applying regular expressions to the POS tags of the annotated text document. Some regular expressions we could use for this purpose can be seen together with extracted verb phrases for the previous example text snippet in Table 4.1:

Regular Expression	**Extracted Verb Phrases**
<VERB><ADV>?	"announced", "have", "complied", "are", "planning", "retreat completely"
<VERB>?<ADV>*<VERB>+	"subsequently announced", "have complied", "are planning", "retreat"
<VERB>*<ADV>*<PART>* <VERB>+<PART>*	"subsequently announced", "have complied", "are planning to retreat"

Table 4.1: Regular expressions for the step "Relation Extraction" together with extracted verb phrases of a sample text [STAa].

For the semi-automatic CMM process, we chose the last regular expression in Table 4.1, which is based on a stackoverflow post [STAa], after some experiments, because the most meaningful combinations of verb phrases could be detected. This last regular expression also captures verb phrases with PART (Particle) POS tags, which are "*function words that must be associated with another word or phrase to impart meaning and that do not satisfy definitions of other universal parts of speech*" [PAR]. Examples for particles are possessive markers or negation particles like the word "not" [PAR].

Proposition Generation

After verb phrases have been extracted, we build propositions by iterating over each sentence and identifying the subjects and objects of a verb in order to create *subject-verb-object* triples. For each sentence, start and end indices for each verb phrase in the sentence are determined. Then, after each verb phrase boundary in a sentence has been detected, an iteration over all verb phrases in the current sentence will be the starting point for proposition generation. The overall procedure, along with its sub-procedures can be seen in Algorithm 4.4.

For each verb phrase, the noun phrase closest to the start index of the current verb phrase will be added to the list of possible subjects m. Similarly, the noun phrase closest

Algorithm 4.4: Relation Extraction and Proposition Generation I

```
1  ExtractRelationsAndCreatePropositions (concepts, annotatedText)
       inputs : The extracted concepts, NLP annotatedText
       output : Tuples of (sourceId, targetId, label, occurrence)
2      relations = new List();
3      verbPhrasePattern = <VERB>*<ADV>*<PART>*<VERB>+<PART>*;
4      verbPhrases = RegExExtractVerbPhrases(annotatedText, verbPhrasePattern);
5      foreach sentence in annotatedText.sentences do
6          conceptsInSentence = GetConceptsInSentence(concepts, sentence);
7          verbPhrasesInSentence = GetVerbPhrasesInSentence(verbPhrases,
            sentence);
8          for i = 0; i < verbPhrasesInSentence.length; i++ do
9              previousVerbPhrase = null;
10             if i − 1 > −1 then
11                 previousVerbPhrase = verbPhrasesInSentence[i-1];
12             end
13             currentVerbPhrase = verbPhrasesInSentence[i];
14             nextVerbPhrase = null;
15             if i + 1 < verbPhrasesInSentence.length then
16                 nextVerbPhrase = verbPhrasesInSentence[i+1];
17             end
18             relations += CreateRelations(sentence, conceptsInSentence,
                currentVerbPhrase, previousVerbPhrase.start, nextVerbPhrase.start);
19         end
20     end
21     relations = ResolveClusterRelations(concepts, relations);
22     relations = RemovePronounRelations(concepts, relations);
23     return relations;
```

to the end index of the current verb phrase is added to the list of possible objects n to create *subject-verb-object* triples. To showcase this approach, we can consider a part of the altered text example: "*The lumberjack Monty Python had known the fox for some time*". With the noun phrase "*The lumberjack Monty Python*" as a subject, the verb phrase "*had known*", and the noun phrase "*the fox*" as an object, we would extract the *subject-verb-object* triple:

The lumberjack Monty Python - had known - the fox

Additionally, each named entity that is located before or after a verb phrase will be added to the list for possible subjects m or objects n respectively. For the altered text example, the named entity "*Monty Python*" would yield another possible subject and another *subject-verb-object* triple could be created:

Monty Python - had known - the fox

Subsequently, all possible subjects m and possible objects n will be combined to $m \cdot n$ *subject-verb-object* triples per verb phrase in a sentence. This sub-procedure is listed in Algorithm 4.5 and Algorithm 4.6.

In addition to the subjects and objects that can be found directly in the neighbourhood of a verb phrase, we can extract additional relations by utilizing the concept mention groups' clusters. Relations that were identified for one concept in a cluster serve as possible relations for all possible concepts in this cluster. As an example, we consider the first part of the sample text: "*The quick brown fox jumped over the lazy dog. Then it ran across the orchard*". Algorithm 4.5 and Algorithm 4.6 would generate the following *subject-verb-object* triples:

The quick brown fox - jumped - the lazy dog
it - ran - the orchard

Since both concepts "*The quick brown fox*" and "*it*" belong to the same cluster with cluster id "*cluster1*", we can generate two additional relations, resulting in the following *subject-verb-object* triples:

The quick brown fox - jumped - the lazy dog
it - ran - the orchard
The quick brown fox - ran - the orchard
it - jumped - the lazy dog

If a concept belongs to a cluster, for all relations that have a link to one of the concepts in the cluster, but no link to the currently viewed concept, a new relation to the currently viewed concept is created. This implies that a new relation r_2 will only be created, if there is no relation $r_2 = (c_1, c_3, l_1)$ from concept c_1 to c_3, but a relation $r_1 = (c_2, c_3, l_1)$, where both c_1 and c_2 belong to the same cluster. A procedure to resolve these cluster relations is defined in Algorithm 4.7 and Algorithm 4.8.

After the cluster relations are resolved, relations that have a pronoun in the relation as a subject or object can be removed. This is possible, because we just transferred relations, which were previously only connected to pronouns, to concepts with noun phrases of the same cluster. Hence, we can remove these relations, since we only want to display real noun phrases and no pronouns as concept suggestions to the user. This leaves us with the following *subject-verb-object* triples:

The quick brown fox - jumped - the lazy dog
The quick brown fox - ran - the orchard

A separate algorithm for this sub-task will not be shown, since it is trivial to remove the relations where either the source or target concept is a pronoun.

Algorithm 4.5: Relation Extraction and Proposition Generation II

```
1  CreateRelations (sentence, concepts, verbPhrase, previous, next)
       inputs : The current sentence, the concepts in the current sentence, the
                current verbPhrase, start index of the previous verbPhrase,
                startIndex of the next verbPhrase
       output : Tuples of (sourceId, targetId, label, occurrence)
2      subjects = new Set();
3      objects = new Set();
4      bestMS = null; // Best matching subject
5      bestMSO = null; // Best matching subject occurrence
6      bestMO = null; // Best matching object
7      bestMOO = null; // Best matching object occurrence
8      foreach concept in concepts do
9          foreach occ in GetOccurencesInSentence(sentence, concept.occurrences) do
10             if occ.start < verbPhrase.start then
11                 if ¬ previous ∨ (previous ∧ occ.start > previous) then
12                     if IsNamedEntity(concept) then
13                         subjects.Add(concept);
14                     end
15                     if ¬ bestMSO ∨ occ.end > bestMSO.end then
16                         bestMS = concept;
17                         bestMSO = occ;
18                     end
19                 end
20             end
21             else if occ.start > verbPhrase.start then
22                 if ¬ next ∨ (next ∧ occ.start < next) then
23                     if IsNamedEntity(concept) then
24                         objects.Add(concept);
25                     end
26                     if ¬ bestMOO ∨ occ.start < bestMOO.start then
27                         bestMO = concept;
28                         bestMOO = occ;
29                     end
30                 end
31             end
32         end
33     end
34     subjects.Add(bestMS);
35     objects.Add(bestMO);
36     return GenerateRelations(verbPhrase, subjects, objects);
```

Algorithm 4.6: Relation Extraction and Proposition Generation III

```
1  GenerateRelations (verbPhrase, subjects, objects)
       inputs : The current verbPhrase, extracted subjects and objects, which are
                both collections of concepts
       output : Tuples of (sourceId, targetId, label, occurrence)
2      relations = new List();
3      if subjects ∧ objects then
4          foreach subject in subjects do
5              foreach object in objects do
6                  source = subject.conceptId;
7                  target = object.conceptId;
8                  if (subject.clusterId ≠ object.clusterId) ∧ (source ≠ target) then
9                      relations.Add((source, target, verbPhrase.text, new
                         Occurrence(verbPhrase.start, verbPhrase.end));
10                 end
11             end
12         end
13     end
14     return relations;
```

Algorithm 4.7: Relation Extraction and Proposition Generation IV

```
1  ResolveClusterRelations (concepts, relations)
       inputs : The extracted concepts and directly found relations
       output : Tuples of (sourceId, targetId, label, occurrence)
2      clusterIdConcepts = GetClusterIdConceptsDictionary(concepts);
3      conceptIdRelations = GetConceptIdRelationsDictionary(concepts, relations);
4      foreach concept in concepts do
5          clusterConcepts = clusterIdConcepts[concept.clusterId];
6          if clusterConcepts ≠ null then
7              foreach clusterConcept in clusterConcepts do
8                  conceptRelations = conceptIdRelations[clusterConcept.id];
9                  clusterRelations = ResolveRelations(concept.id, conceptRelations,
                     clusterConcept.id, relations);
10                 conceptIdRelations[concept.Id] = conceptRelations +
                     clusterRelations;
11                 relations = relations + clusterRelations;
12             end
13         end
14     end
15     return relations;
```

Algorithm 4.8: Relation Extraction and Proposition Generation V

```
1  ResolveRelations (conceptId, conceptRelations, clusterConceptId, relations)
       inputs : The conceptId and conceptRelations of the current concept, the
                clusterConceptId of the other concept in the same cluster from which
                relations should be resolved, all found relations
       output : Tuples of (sourceId, targetId, label, occurrence)
2      resolvedRelations = new List();
3      foreach relation in relations do
4          if relation.source == clusterConceptId ∧ ¬ EdgeExists(conceptRelations,
            conceptId, relation.target, relation.label) then
5              resolvedRelations.Add((conceptId, relation.target, relation.label,
                relation.occurrence));
6          end
7          if relation.target == clusterConceptId ∧ ¬ EdgeExists(conceptRelations,
            relation.source, conceptId, relation.label) then
8              resolvedRelations.Add((relation.source, conceptId, relation.label,
                relation.occurrence));
9          end
10     end
11     return resolvedRelations;
```

4.3.6 Results

As a result for the whole "Automatic Text Processing" step, we get propositions in the form of concepts and relations. For each concept, we receive a concept id, a cluster id, a label, a flag that tells us if the concept is a pronoun and a collection of occurrences, where each occurrence is described by a start and end index in the text. Each relation has a label, a source and target property that both contain the id of a concept and a single occurrence in the text with a start and end index. The source and target properties cannot contain the same concept id, which means that a concept cannot refer to itself with a relation. The following example shows the extraction result with this approach. After processing, we receive the extracted concept and relation results that can be seen in a JSON format in Listing 4.3.

In Listing 4.3 we can also see that concepts have a collection of occurrences, whereas relations only have one occurrence. Some relations, like "*jumped*" and "*had known*", appear several times and contain subjects or objects of the same clusters. It would be also possible to consolidate these relations and return them only once in the result, with multiple occurrences, similar to the concepts. Instead of direct source and target references to concept ids, the references would need to contain the cluster ids. However, the realized data structure was chosen, because it was easier to handle the relations in the user interface with this format.

```
{
    "concepts": [
        {"id": "concept1", "clusterId": "cluster1", "label": "The quick brown fox", "isPronoun":
            false, "occurrences": [{"start": 0, "end": 19}]},
        {"id": "concept2", "clusterId": "cluster2", "label": "the lazy dog", "isPronoun": false, "
            occurrences": [{"start": 32, "end": 44}]},
        {"id": "concept3", "clusterId": "cluster1", "label": "it", "isPronoun": true, "occurrences
            ": [{"start": 51, "end": 53}]},
        {"id": "concept4", "clusterId": "cluster3", "label": "the orchard", "isPronoun": false, "
            occurrences": [{"start": 65, "end": 76}]},
        {"id": "concept5", "clusterId": "cluster4", "label": "The lumberjack", "isPronoun": false,
             "occurrences": [{"start": 78, "end": 92}]},
        {"id": "concept6", "clusterId": "cluster1", "label": "the fox", "isPronoun": false, "
            occurrences": [{"start": 103, "end": 110}]},
        {"id": "concept7", "clusterId": "cluster5", "label": "some time", "isPronoun": false, "
            occurrences": [{"start": 115, "end": 124}]}
    ],
    "relations": [
        {"label": "jumped", "source": "concept1", "target": "concept2", "occurrence": {"start": 20
            , "end": 26}},
        {"label": "had known", "source": "concept5", "target": "concept6", "occurrence": {"start":
             93, "end": 102}},
        {"label": "ran", "source": "concept1", "target": "concept4", "occurrence": {"start": 54, "
            end": 57}},
        {"label": "had known", "source": "concept5", "target": "concept1", "occurrence": {"start":
             93, "end": 102}},
        {"label": "jumped", "source": "concept6", "target": "concept2", "occurrence": {"start": 20
            , "end": 26}},
        {"label": "ran", "source": "concept6", "target": "concept4", "occurrence": {"start": 54, "
            end": 57}}
    ]
}
```

Listing 4.3: Concept and relation results from the “Automatic Text Processing” pipeline.

4.4 Manual Concept Map Construction

After the propositions have been extracted and the "Automatic Text Processing" step has finished, the concepts and relations can be transferred to the user interface. There, the user has the ability to construct the concept map. The user interface provides all tools necessary to manually build a concept map, but also offers functionality to assist the user in accordance with the extracted concepts and relations. The basic goal for the user interface is to keep the visual representation clean and clutter free and the process of concept map construction as simple and unobtrusive as possible. A good balance between using automatically extracted concept and relation suggestions and being able to manually add and arrange custom concepts and relations is pursued. An overview of the user interface is given in Section 4.4.1. Section 4.4.2 provides insight into the process of semi-automatic concept map creation.

4.4.1 User Interface Overview

The user interface for the semi-automatic CMM approach is composed of four major areas. A screenshot of the user interface is shown in Figure 4.5.

1. The "toolbar" contains buttons for supplementary functions. It consists of a button to download the created concept map in PNG format and additional buttons to fit and center the map to the canvas. With the "clear map" button, all selected concepts and relations, as well as all manually constructed concepts and relations are deleted, which essentially clears the whole map but keeps extracted concepts and relations in the background. The right-most "add text" button opens a popup, where the user can submit the text that should be processed with the "Automatic Text Processing" functionality.

2. The "canvas" displays the concept map and enables its creation and manipulation. Concepts can be added manually with a click in an empty area of the "canvas" or by clicking a concept in the "suggested concepts sidebar". After an existing concept is selected, suggestions for connected concepts and relations are displayed inside the "canvas", if the user additionally activates the "show suggestions" button. Furthermore, manual relations can be added: the user selects a concept, clicks the "add relation" button and clicks another concept to which the new relation should be connected to. With the "rename" button, created concepts and relations can be renamed and deleted with a "delete" button. Additionally, the canvas supports zoom and panning functionality with the mouse-wheel.

3. The "suggested concepts sidebar" is shown in Figure 4.6. It contains all concepts that are extracted by the "Automatic Text Processing" step. They appear after the user has submitted the text through the "add text" button in the "toolbar". Pronouns are not shown in this area, similarly to the "canvas", because we do not want pronouns as concepts. The property "isPronoun" from the returned concepts is used for their identification. If a concept is selected, it is added to the "canvas" and highlighted in the sidebar. Additionally, all concepts that belong to the same cluster are highlighted as well and cannot be selected simultaneously.

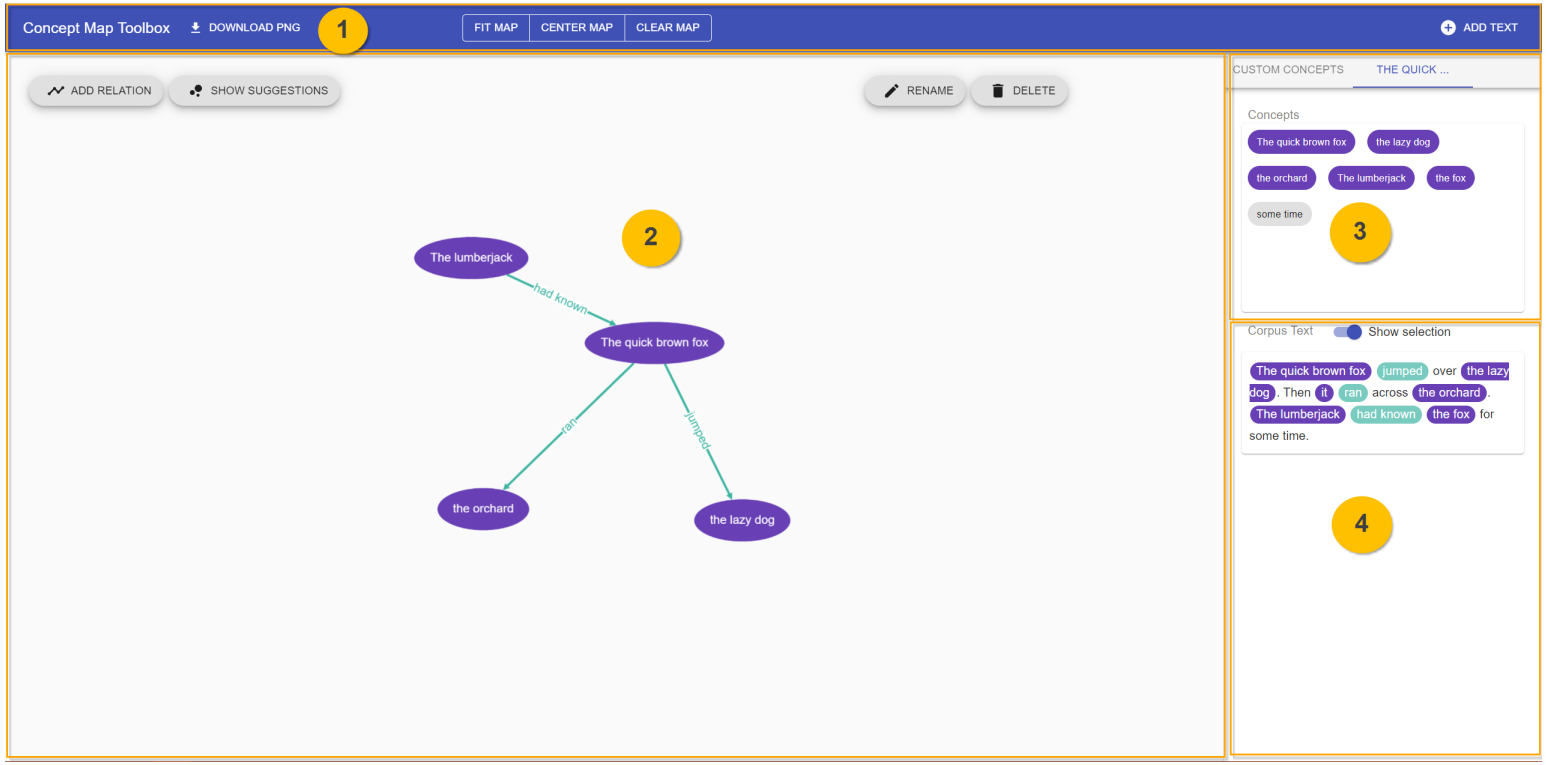

Figure 4.5: Overview of the user interface for the “Manual Concept Map Construction” step in the Semi-Automatic CMM approach. The following user interface elements can be seen here: 1) A “toolbar” for supplementary functions, 2) the concept map “canvas”, 3) the “suggested concepts sidebar” and 4) the “text area”.

The sidebar also contains a tab where all concepts that were added manually are visible. If more than one text is added to the concept map, each text together with its concepts can be selected in a different tab. In its current state, the semi-automatic CMM process is only designed for mining concepts from a single text. However, adding the handling of multiple texts to this process can be seen as part of possible future work.

4. In the "text area", the submitted text is displayed after it was processed. The main intention of this part of the user interface is that the underlying text for the concept map construction is always visible to the user. A crucial feature of the "text area" is the highlighting of selected concepts and relations. If a user adds a concept from the "suggested concepts sidebar" or a relation inside the "canvas", the added concept or relation is highlighted in the text. Additionally, if a user selects a concept with a click in the "canvas", the text for the currently selected concept is highlighted in the "text area" with a different color. If a concept belongs to a cluster, all concepts of the same cluster are highlighted as well. The feature is achieved with the extracted occurrence indices for each determined noun and verb phrase. Originally, regular expressions were considered for this feature, but it is hard with regular expressions to distinguish the same pronouns that belong to different clusters and subsequently differentiate when to highlight which of these pronouns. With available occurrence indices and cluster ids for each phrase, it is possible to only highlight selected concepts and clusters, even if the same words are used for different concepts in the text. The highlighting of the example text is also shown in Figure 4.7, where the concept "*The quick brown fox*" is selected in the "canvas" and together with all its cluster concepts, i.e., "*it*" and "*the fox*" highlighted in the "text area".

This highlighting feature was designed to give the user a sense of progress during the process of concept map construction. It shows which concepts of the text are already transferred to the concept map and also which of the possible concepts in the "suggested concepts sidebar" belong to which part of the text. This is achieved by highlighting the respective concepts and clusters when the user hovers over a concept in the "suggested concepts sidebar". A "show selection" toggle button can be used, if the user feels that the text is getting unreadable because too many parts of the text are already highlighted.

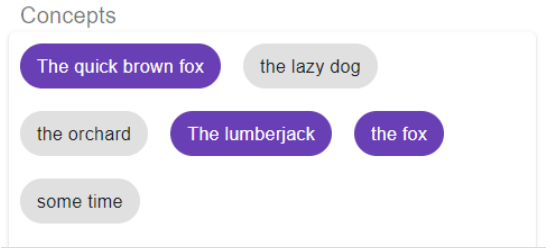

Figure 4.6: Suggested concepts sidebar.

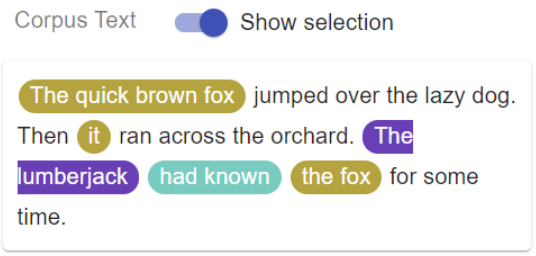

Figure 4.7: Text area with highlighted text.

4.4.2 Summarization and Topological Arrangement

The process of "Summarization" and "Topological Arrangement" begins when the user clicks the "add text" button, submits the text to the "Automatic Text Processing" pipeline and receives the pipeline's result. The extracted propositions from Listing 4.3 are loaded in the user interface after the "Automatic Text Processing" finishes. The extracted concepts are immediately visible in the "suggested concepts sidebar" and the submitted text is shown in the "text area". The following sequence of steps, the user has to undertake to create the concept map with the semi-automatic CMM approach, is illustrated in Figure 4.8.

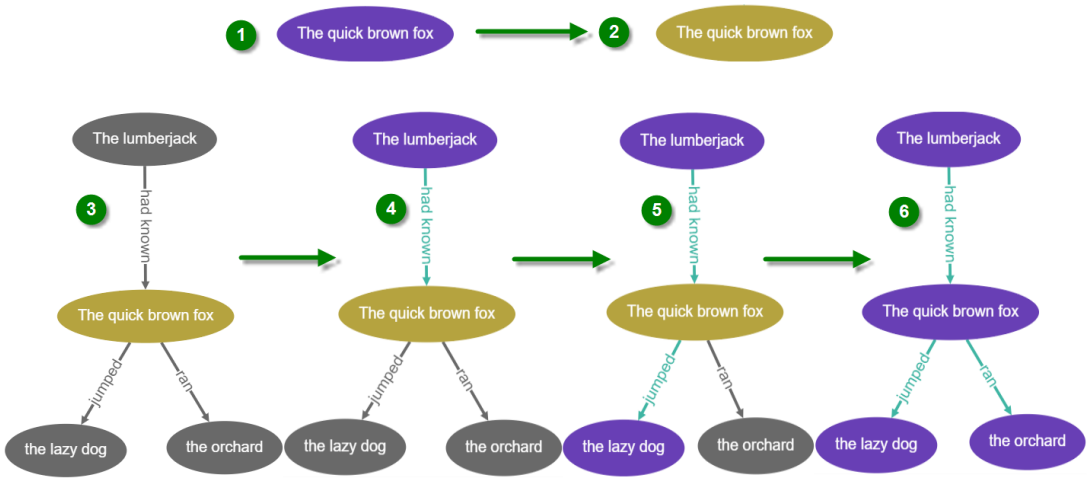

Figure 4.8: Creation of a concept map by selecting suggested concepts and relations.

The user can now begin with creating the concept map by reading the text in the "text area". Once a first concept is identified, the user can click the concept in the "suggested concepts sidebar", which adds the concept to the "canvas". We suppose the user selects "*The quick brown fox*" as a first concept (Step 1 in Figure 4.8). After the concept has been added, the user can select the concept in the "canvas" with a single mouse-click, which highlights the concept in an ochre color and marks it as selected (Step 2 in Figure 4.8). By clicking the "show suggestions" button, all concepts that are connected to the concept "*The quick brown fox*" appear in the "canvas", together with their labelled relations in a grey color, which marks them as only being suggestions (Step 3 in Figure 4.8).

If the user does not want to see any suggestions, they can be hidden again with another click on the "show suggestions" button. If, however, a concept seems suitable, the suggested concept or relation can be clicked, which adds it to the concept map and assigns a purple (concept) or turquoise (relation) color. We suppose the user selects the concept "*The lumberjack*" with the relation "*had known*". The "suggested concepts sidebar" in Figure 4.6 now shows that the concepts "*The quick brown fox*" and "*The lumberjack*" are already transferred to the concept map by appearing in a purple background. Additionally,

we can see that the concept "*the fox*" is marked as transferred as well, because it belongs to the same cluster as "*The quick brown fox*". Similarly, the transferred concepts and the relation "*had known*" are highlighted in the "text area" (Figure 4.7). If the user now clicks the concept "*The quick brown fox*" again and afterwards the "show suggestions" button, the remaining concepts "*the lazy dog*" and "*the orchard*" appear as suggestions in the "canvas" (Step 4 in Figure 4.8). Furthermore, we can see in the "text area" in Figure 4.7 that the currently selected concept "*The quick brown fox*" and all the concepts in its cluster are marked as selected with an ochre color, matching the selection in the "canvas".

In a similar manner, the user can now finish the concept map by selecting the concept "*the lazy dog*" (Step 5 in Figure 4.8) and afterwards the concept "the orchard" (Step 6 in Figure 4.8). It is now up to the user to apply "Topological Arrangement" of the concepts in the "canvas" with simple drag and drop maneuvers. Changing concept and relation labels and deleting or adding elements in the "canvas" can additionally be applied with further rearrangement of concepts, until the creator considers the concept map as finished.

Naturally, the chosen example for this semi-automatic CMM process is highly simplified for the purpose of showcasing the approach. Creating a real-world concept map involves many more applications of the later mentioned recurring tasks in an iterative manner. The user might also want to re-read the submitted text from time to time to contemplate on the concept map creation and while doing so might toggle the "show selection" button in the "text area" for easier reading. Having the extracted concepts and relations as close to the text as possible should also help to make the mental connection to the text in the "text area" and the concept in the "suggested concepts sidebar" and "canvas". If a topic or text is more complex and not all extracted suggestions are fitting, more manual concepts and relations could be created. In a nutshell, this overall process of building up the concept map with the extracted concept and relation suggestions can be seen as a combination of recurring tasks that involve the provided user interface components. These tasks can be subsumed as follows and repeated as often as necessary:

1. Read the submitted text in the "text area".
2. Select initial concepts from the "suggested concepts sidebar".
3. Choose further concepts and relations from the suggestions in the "canvas" with the "show suggestions" button.
4. Trace the progress of the concept map construction with highlighted concepts and relations in the "text area".
5. Toggle the highlighted text in the "text area" with the "show selection" button, if the text needs to be re-read and too many elements are highlighted.
6. Center and fit the concept map in the "canvas" with the buttons in the toolbar, if the zooming and panning features were used.

7. Add manual concepts and relations for more detailed elements.

8. Rename concepts and relations if necessary.

9. Apply Topological Arrangement of the concepts in the "canvas".

4.5 Implementation

The prototype for this semi-automatic CMM approach was implemented with a simple client-server architecture as a web-application. The server is the implementation for the "Automatic Text Processing" approach, whereas the client represents the user interface for the "Summarization" and "Topological Arrangement" parts.

Both client and server were deployed and hosted in the cloud application platform Heroku [HER] in order to be accessible over the web for the user evaluation. However, processing of longer texts (starting at approx. 1500 words) is not possible with the free account on Heroku [HER], because the free containers (dynos) are used for the server that only provide up to 512 MB RAM per user. The machine learning models that are used in the server consume almost all available RAM in the dynos. With this limitation, longer texts cannot be processed in the hosted container environment due to the exhaustion of the available RAM.

4.5.1 Server

The server was written in Python and exposes a single Http-POST method in a REST-API interface, which uses the web-application framework Flask [FLA]. This Http-POST method receives the text that should be processed and returns the extracted concepts and relations. A Http-POST method was used, because larger payloads (texts) should be transferred in the request's body and not in a Http-GET method's query parameter. Table 4.2 shows the used frameworks and libraries for the server implementation.

Framework / Library	**Version**	**Purpose**
Python	3.7.4	Programming language
Flask	1.1.1	REST API
spaCy	2.1.0	NLP framework
en_core_web_sm	2.1.0	spaCy language model
NeuralCoref	4.0.0	Coreference resolution
textacy	0.9.1	Verb phrase extraction

Table 4.2: Frameworks and Libraries used for the "Automatic Text Processing" server implementation.

The NLP pipeline for the processing of the submitted text uses the NLP framework spaCy [SPA]. spaCy provides all the necessary tools to conduct tokenization, sentence

segmentation, POS tagging, dependency parsing, lemmatization and named entity recognition. Internally, spaCy uses neural networks for POS tagging, dependency parsing and named entity recognition [SPA]. It provides pre-trained models for this purpose, in a variety of languages and sizes. For English, three sizes of models - small, medium, and large - are available. For the prototype implementation, we chose the smallest model *en_core_web_sm*, because it provided the smallest footprint and shortest computation time with almost immediate execution of the proposed "Automatic Text Processing" pipeline. With a larger model, the computation time increased significantly and the small model seemed to provide good enough accuracy for the purpose of this thesis.

However, spaCy currently provides no native coreference resolution functionality. Therefore, a separate module called NeuralCoref [NEU], which can be loaded into spaCy, was used for coreference resolution. NeuralCoref uses a neural network based scoring system for coreferer that is an implementation of the work by Clark and Manning [CM16a][CM16b], which was explained in Section 4.3.2. This implementation builds on the parsing features of spaCy and deep-learning Python tools like Numpy [NEU]. NeuralCoref comes with a pre-trained model for English, but provides instructions and tools to train custom models if needed. The pre-trained model was used for the prototype implementation.

Another extension module for spaCy, which is called textacy [TEX], was utilized for extracting the verb phrases from the POS tags with a regular expression. Apart from that, it provides convenience methods that can be applied before or after the spaCy NLP pipeline, like filtering stop-words, extracting n-grams or noun chunks, or statistics about the text. For this thesis, the feature *pos_regex_matches* was chosen, which receives a regular expression and returns the matched tokens based on the annotated POS tags.

4.5.2 Client

The client was built as a web application in JavaScript that can be run in any modern browser. It uses current state-of-the-art JavaScript frameworks and proven user interface concepts, such as material design. Communication with the Http-POST method of the server is done with a simple call via the fetch API of current major browsers, like Chrome, Firefox or Safari. Table 4.3 shows the used frameworks and libraries for the client implementation.

Framework / Library	**Version**	**Purpose**
JavaScript	ECMAScript 2019	Programming language
React	16.10.2	JavaScript UI framework
Material-UI	4.5.1	UI components and layout
Cytoscape.js	3.11.0	Concept map drawing
react-cytoscape	1.2.0	React wrapper for Cytoscape.js

Table 4.3: Frameworks and Libraries used for the "Summarization" and "Topological Arrangement" client implementation.

The main library that was used to create the user interface is React [REA], which is currently one of the most popular front-end libraries for building user interfaces in JavaScript [STAb]. On top of that, Material-UI [MAT] provides pre-built user interface components for React in the style of material design. This library was chosen in order to apply a proven user interface concept to the whole user interface and ensure a consistent design. Overall, React in combination with Material-UI was a good choice for the prototype implementation, because this combination allows to create all kinds of simple and minimalist user interfaces with good performance in a reasonable amount of time. One example that was particularly enjoyable to implement with React's declarative style was the highlighting of the selected concepts and relations in the "text area".

Another important part of the user interface, the "canvas", where drawing and manipulation of the concept map happens, was not easily realizable with native React components. A special library was needed to enable the required drawing functionality on the "canvas". Instead of creating such a functionality from scratch with drawing libraries such as D3.js, a library called Cytoscape.js [CYTb] was used that provides all necessary functionalities to draw and manipulate graphs. Additionally, a React wrapper library [CYTa] for Cytoscape.js was utilized in order to be able to use it as a component with React. Although Cytoscape.js worked for most user interface interactions to create the concept map, it is actually a library for the visualization of networks. It had its shortcomings when placing and removing concepts in the "canvas" was necessary for displaying the temporary concept suggestions to the user. Basically, a *circle* network layout algorithm was employed to align the concept suggestions around the currently selected concept, which resulted in misplaced and overlapping concepts when many suggestions were given. This behaviour is shown in Figure 4.9.

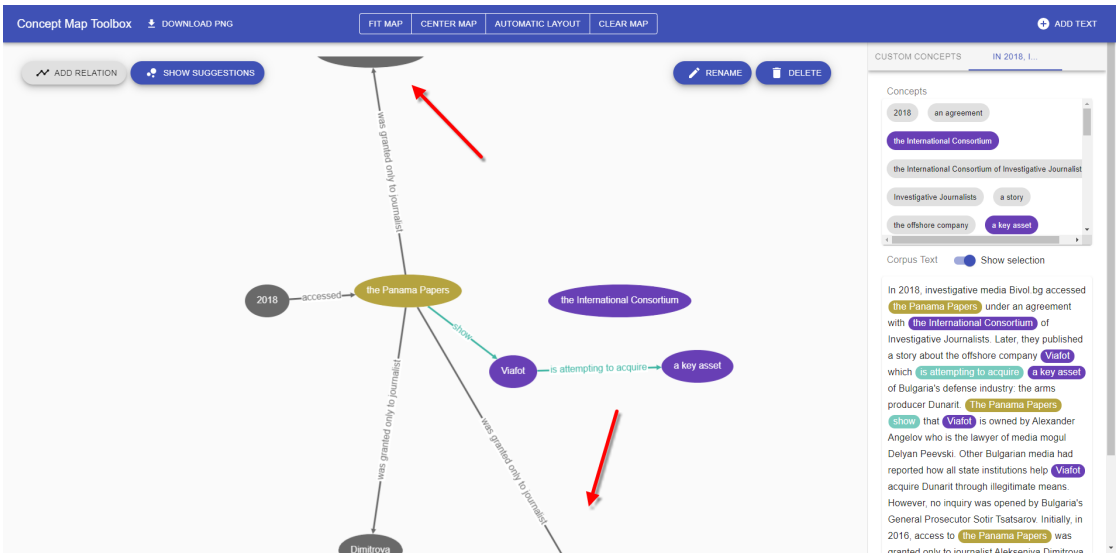

Figure 4.9: Misplaced concept suggestions with Cytoscape.js circle layout.

The problem was tedious for the user, because the concept had to be moved or searched in the "canvas". If our semi-automatic CMM approach would be released as a product, this aspect would certainly need to be improved on. If substantial improvements cannot be achieved with Cytoscape.js, a different technology would need to be used for this purpose.

4.6 Examples

In this section we try to mine two example concept maps. The first example in Section 4.6.1 represents a case where the semi-automatic CMM works as intended. It extracts helpful suggestions and the user interface supports the user in creating the concept map. Additionally, some other aspects of the semi-automatic CMM approach are highlighted. The second example in Section 4.6.2 explores how the approach scales with large documents that are several pages long.

4.6.1 Positive Example

As a first example, we tried to mine a short text that was also used as an example text for the user study:

"*Austria's financial market authority has announced that they will audit two Austrian banks that were mentioned in the Panama Papers: Raiffeisenbank International and Hypo Vorarlberg. It will be specifically examining whether the banks have complied with their obligation to prevent money laundering. Hypo Vorarlberg subsequently announced that while they have complied with all laws in the past, they are planning to retreat completely from the offshore sector.*"

After submitting the text, the "Automatic Text Processing" functionality extracted 17 concepts, including pronouns for the text highlighting, and 19 relations. The extracted concepts are shown in Figure 4.10. The most relevant concepts, like "*Austria's financial market authority*", "*the Panama Papers*", "two Austrian banks", "*Raiffeisenbank International*", "*Hypo Vorarlberg*", "*money laundering*", and "*the offshore sector*" could be extracted.

If we would insert "*Austria's financial market authority*" as the first concept, the system would suggest several relations to new concepts, which we could use for the further construction of the concept map. These relations can be seen in Figure 4.11. Some of the relations like "*will audit - two*" or "*Austria - has announced*" are less relevant, but were nevertheless extracted. This can be attributed to the intended higher recall.

Other relations like "*will audit - two Austrian banks*" or "*will be specifically examining - the banks*" could be exactly the relations, which a potential user would be looking for. The system correctly detected the coreference cluster *["Austria's financial market authority", "it", "they"]*, from which these two relations could be resolved for the concept "*Austria's financial market authority*". However, the system was not able to detect that "*two Austrian banks*" and "*the banks*" are the same concept.

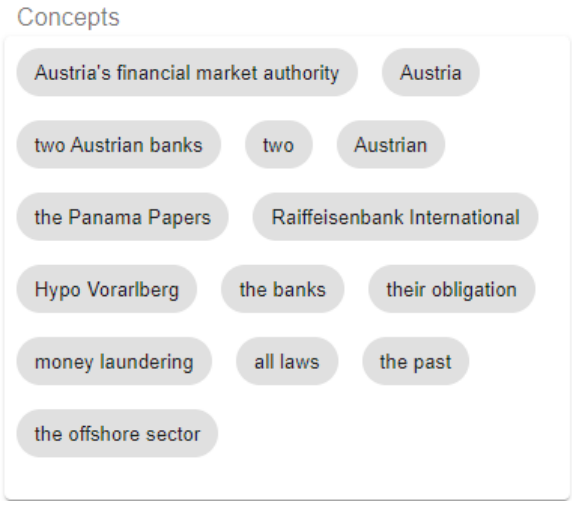

Figure 4.10: Extracted concepts for the first example text.

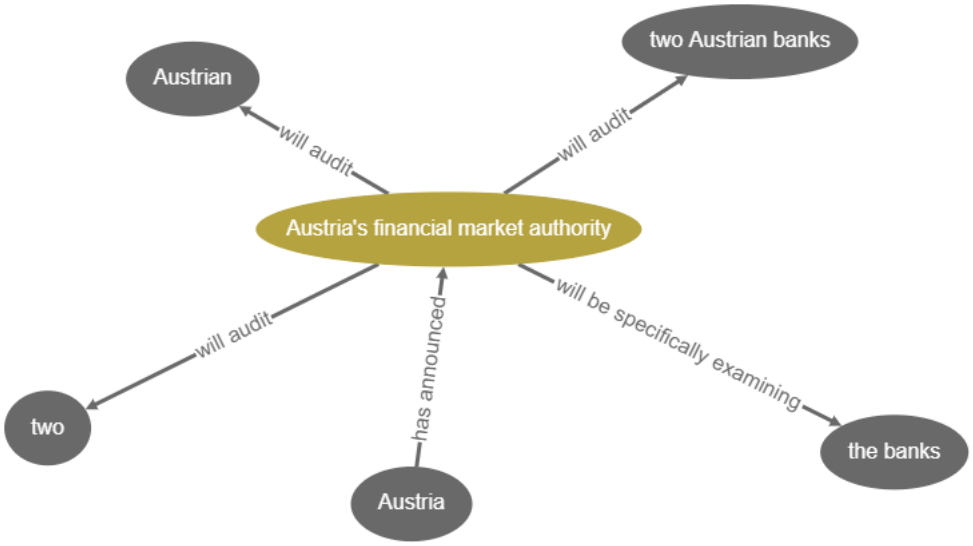

Figure 4.11: Extracted relations for the first example text and the concept "*Austria's financial market authority*".

We would now choose the concept "*two Austrian banks*" and show the suggestions for this concept in Figure 4.12. The nearest object to the verb phrase "*were mentioned*" in the text is "*the Panama Papers*", which the system correctly identified as a concept. Through the inclusion of named entities, "*Raiffeisenbank International*" and "*Hypo Vorarlberg*" could be obtained as additional concepts. Since all three of the relations to the new concepts in Figure 4.12 make sense, they could be added to the concept map with simple clicks on the suggested three concepts in the "canvas".

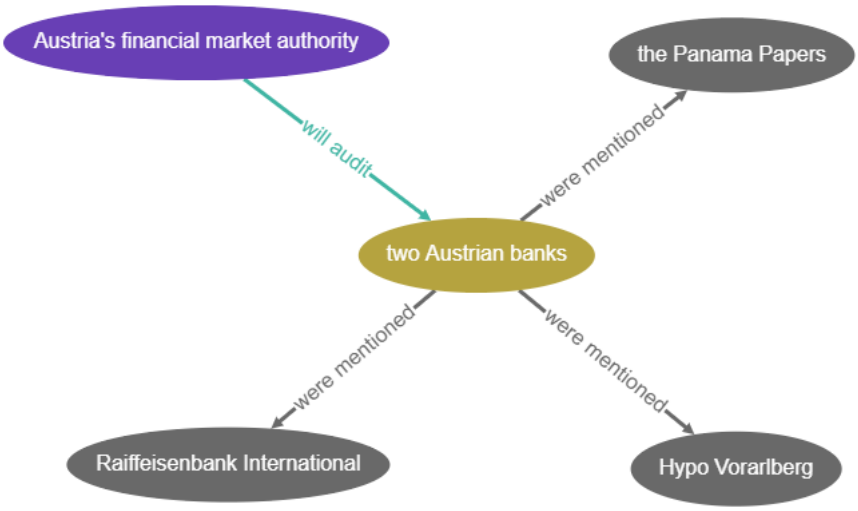

Figure 4.12: Extracted relations for the first example text and the concept "*two Austrian banks*".

Afterwards, we could complete the concept map by adding a relation that would indicate that "*Hypo Vorarlberg*" are planning to retreat from the offshore sector. However, in Figure 4.13 we can see the following highlighted coreference cluster that the model detected as a false positive: *["the banks", "they", "they"]*. We would have expected that "*they*" would build a cluster with "*Hypo Vorarlberg*". This shows that we use a machine learning model for coreference resolution that does not achieve 100% accuracy.

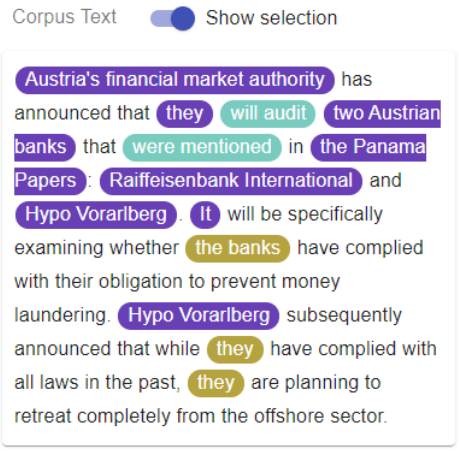

Figure 4.13: Text highlight for the first example text.

However, since the user interface provides functionality to conduct manual tasks, we could simply add the desired relation "*are planning to reatreat*" and concept "*the offshore sector*" manually. The finished concept map for this example is shown in Figure 4.14.

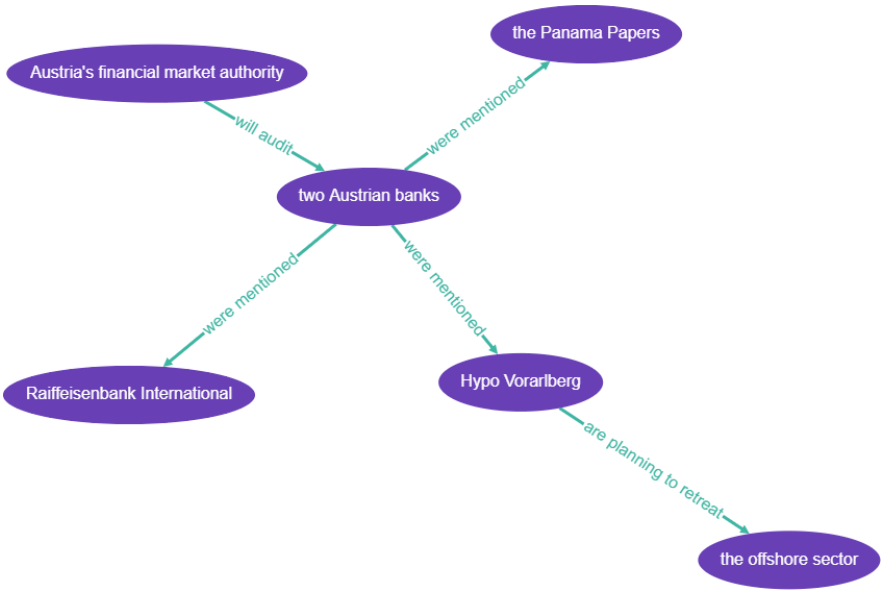

Figure 4.14: Finished concept map for the first example text.

4.6.2 Scaling Issues with Long Text Documents

For the second example, we used a deliberately long text and tried to exhaust the system. It contained 6580 words and was 42.614 characters or 9 pages long. The content of the example text is irrelevant for this experiment and any text of this length could be used. After submitting the text, the "Automatic Text Processing" functionality extracted 1201 concepts, including pronouns for the text highlighting, and 889 relations.

Unfortunately, the system degraded in processing time, user interface response time, and usability. Highlighting text in the "text area" or showing suggestions in the "canvas" took up to two seconds until finished. This made the system unresponsive and tedious to use. Performance degradation tends to start for documents that have a length longer than approximately 750 words or 5000 characters. Figure 4.15 shows the user interface with the second, long text example.

The following usability issues could be noted that can also be seen in Figure 4.15:

1. Some concept suggestions are shown outside of the canvas. This is the issue that was also reported in Section 4.5.2.
2. The mental connection between the "suggested concepts sidebar" and the "text area" gets lost, because the "suggested concepts sidebar" slips outside of the visible area. This problem could be easily fixed by docking the "suggested concepts sidebar" at the top and making only the "text area" scrollable.
3. Selected concepts in the "canvas" are not always visible as highlighted concepts in the "text area", due to the scrolling position of the "text area".

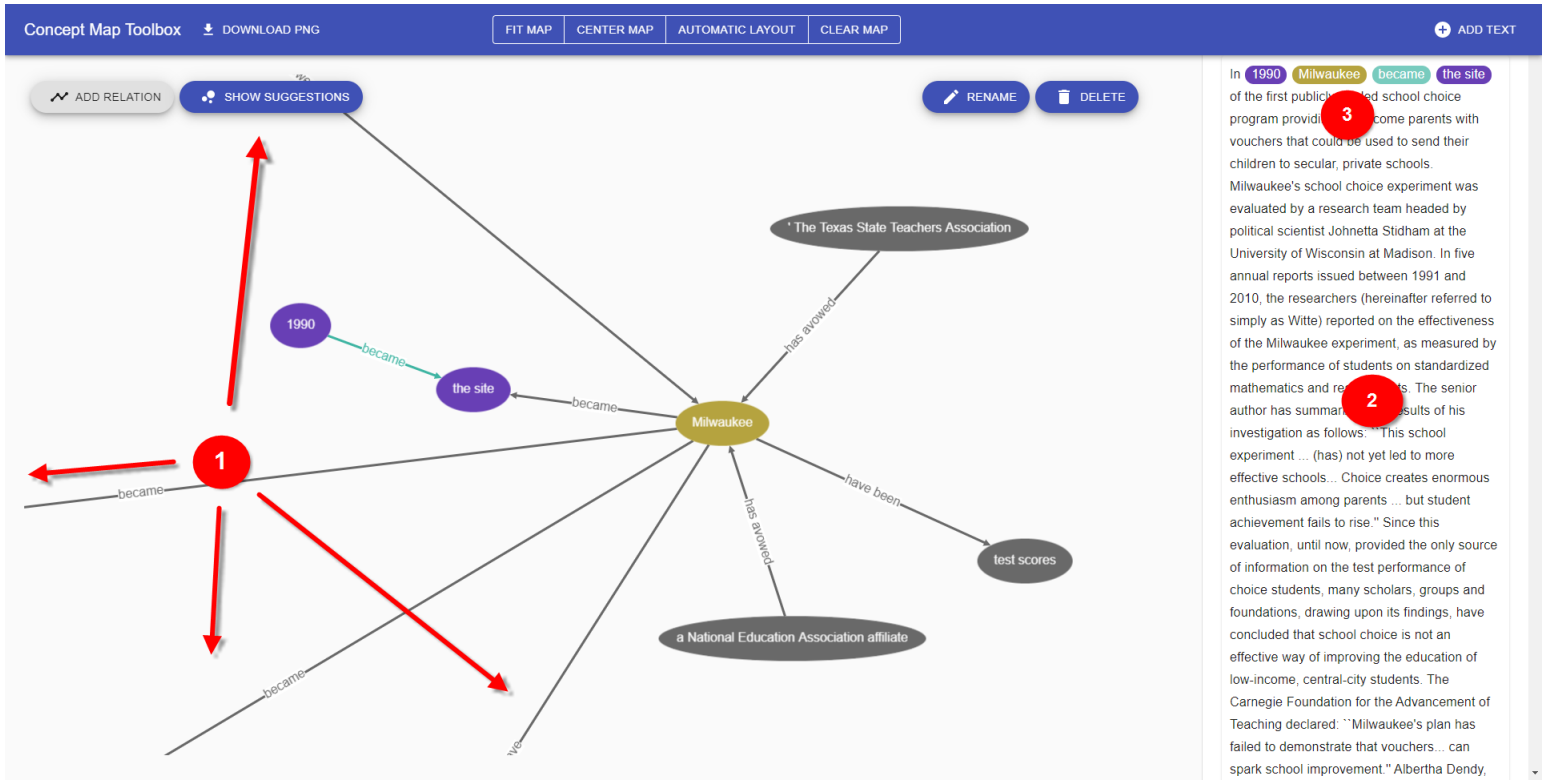

Figure 4.15: User interface overview with second, long example text.

CHAPTER 5

Evaluation and Results

When analyzing concept maps, several important aspects need to be taken into account that apply to their semantic nature as a representation of organized knowledge. The assessment and comparison of different concept maps is not simply a technical task in terms of measuring the similarity of two maps. The evaluation also depends on the objective quality of the concept maps in regard to accurately representing the semantic meaning of the topic or focus question, for which the maps were created for. This can be problematic, because the evaluation of concept maps by researchers is often highly subjective and dependent on the personal opinion of a minority of experts [ZKM12]. The focus can also be shifted to another aspect that measures the level of accuracy concerning user intention. Research shows that different creators, even with the same level of knowledge, create different concept maps [HOJS99]. Here, the concept map would need to accurately represent the knowledge model that the creator of the map had in mind. In order to evaluate the semi-automatic CMM approach on its usefulness, an evaluation method needs to be used that keeps these aspects in mind.

A standardized method for evaluating different CMM approaches that includes a measure for the special aspects discussed in the last paragraph would be a preferred way to conduct such an evaluation task. Unfortunately, no such standardized method exists to this date. Research nowadays mostly focuses on the fully automatic approaches for constructing concept maps. Hence, the evaluation methods that are used, stem from the evaluation of automatic construction. Another problem is that the evaluation methods that are used in existing research differ from each other and have therefore limited comparability [Fal19]. Nevertheless, some parts of those methods seem suitable to be used for the semi-automatic evaluation and to compare results from different (manual/semi-/automatic) approaches. Therefore, we have to derive an appropriate evaluation method for the semi-automatic approach that also allows to compare it to other approaches on the basis of existing methods in corresponding related work.

Initially, two hypotheses were proposed, which upon verification, could justify the usage of a semi-automatic CMM approach. The suggested hypotheses need to be considered as well, when choosing an evaluation framework for this thesis:

- H1: The semi-automatic approach can lead to a higher accuracy than the fully automatic one.
- H2: A semi-automatic approach is more efficient compared to a fully manual creation of a concept map.

Accuracy in *H1* is to be read as the accurate representation of the creators' mental model of the concept map, as stated before. *H1* is therefore verified as true when the semi-automatic approach yields a map with higher overlap than an automatic approach, in comparison to a gold standard concept map also created by the same user. A semi-automatic approach would, as a consequence, be a better tool for expressing subjective variations of concept maps created by different users than a fully automatic one.

The second hypothesis *H2* is to verify that a semi-automatic approach can deliver the higher accuracy that was proposed in hypothesis *H1*, while at the same time being more efficient in creating the concept maps than with a manual tool. The goal of a semi-automatic tool should be to assist a user in completing a task and offer support in sub tasks that require the most effort or are tedious if conducted manually. Consequently, *H2* can be verified as true if a semi-automatic tool provides meaningful support in creating a concept map and makes this process more efficient, and at the same time, less tedious than manually constructing the same map.

5.1 Evaluation Methods in Related Work

Falke [Fal19] groups the different existing evaluation methods that are currently used in research into the following three categories:

1. Automatic comparison against references,
2. Manual (comparative) quality judgements, and
3. Task-based (quantitative) evaluations in end user applications.

Some evaluation methods that use "Automatic comparison against references" can be juxtaposed to evaluation methods in the research area of the semantic web. The measures for graph matching in the semantic web, like the semantic distance, are based on semantic relations defined in an ontology such as OWL, which consists of a schema that defines which concepts and relations exist [GQ08]. Instead of measures that are based on *defined* concepts and relations, metrics have to be calculated in the evaluation of concept maps that utilize the structure of and the propositions contained in a constructed map, all

in comparison to a reference gold standard map [Fal19]. These automatic evaluation methods have the advantage that the experiments are cheap, fast, and repeatable, but have the disadvantage that the summarization of texts in a concept map cannot always be accurately compared, due to the still existing limitations in natural language understanding [Fal19]. In the semantic web, a concept and a relation are strictly defined and measures can be evaluated automatically without the pitfall of different semantic meanings. Automatic measures in concept map evaluation on the other hand might have different scores, simply because of the use of different words in a proposition, even if a proposition would have the same semantic meaning [Fal19].

One example of such a metric used in "Automatic comparison against references" would be ROGUE [Lin04], which measures overlapping word sequences and pairs of computer generated textual summaries against gold standard summaries created by experts. Additionally, automatic methods require the generation of test and evaluation data sets that sometimes need to be in a specific format to conduct the automatic evaluation.

Another example of a measure for calculating the distance between two concept maps was proposed by McClure et al. [MSS99]. The neighbourhood similarity of each concept is calculated for each concept in comparison to the same concept in the other compared map. This similarity is measured by taking the direct neighbours of one probed concept for each map. The Jaccard-Index is then calculated for these neighbours, which is explained in detail in Section 5.2.1. These neighbourhood Jaccard-Index similarities are then averaged to get the measure for the whole map.

"Manual (comparative) quality judgements" are subjective, qualitative reviews of created concept maps that are assessed by experts. They can also involve the comparison against a gold standard map. For instance, Zubrinic et al. [ZOS15] used this method through a quantitative online questionnaire and they let users grade automatically created concept maps in comparison to manually constructed concept maps, which served as the gold standard.

With "Task-based (quantitative) evaluations in end user applications", it can be determined, whether the created concept maps are actually useful in real world scenarios [Fal19]. As an example, Valerio et al. [VL12] set up an evaluation, where they tried to find out if students' reading comprehension skills can be improved by using concept maps. In this study, they presented either a text, a manually, or an automatically created concept map to students. The concept maps were created based on the presented text. On one of these presented options, the test students had to answer certain questions about the topic. With the provided answers, Valerio et al. [VL12] found out that neither manually nor automatically created concept maps did help in enhancing the accuracy of users' reading comprehension skills, but increased the speed of answering questions about the underlying topic. Evaluations of this kind are mostly tied to one experiment itself, because the methods try to evaluate specific properties of a certain research question and are therefore mostly not suitable to be used in other experimental setups [Fal19].

5.2 Evaluation Methodology

For this thesis, a combined evaluation method is used that can answer the proposed hypotheses and is also relatable to evaluation methods of the categories "Automatic comparison against references" and "Manual (comparative) quality judgements".

Because the approach is semi-automatic, test users are necessary for the evaluation of the prototype implementation. A set of five test users was chosen for this purpose. The users were first asked to create five concept maps, from five different texts, with a tool for manual concept map creation called "MindMup" [MIN]. The texts were randomly chosen texts from Wikipedia, based on the leaked documents about offshore companies, called "The Panama Papers" [PANb]. This topic was chosen because no user had previous knowledge about this topic, which should eliminate biased results. The maps created with the manual tool would serve as the gold standard, which represents the most accurate concept maps about the underlying texts in the users' opinion. The manual tool "MindMup" [MIN] also serves as a comparison for the semi-automatic approach in terms of usability and user experience, as well as improved assistance and help during creating the concept maps. The time the users needed to construct each map manually was measured as well.

In the second part of the study, the users needed to construct concept maps from the same texts, but this time with the semi-automatic prototype implementation. We tried to further reduce bias by setting the dates for the first and second part of the user study one month apart. The users were also not allowed to look at the previously created manual gold standard concept maps. Again, the time the users needed to create the maps was measured. Additionally, the users needed to answer a questionnaire, explained in Section 5.2.2, about their experience in using the manual tool in comparison to the semi-automatic prototype and conduct a "Heuristic Evaluation" of their experience, which is explained in detail in Section 5.2.3.

Finally, reference concept maps were created for each text with a tool that follows the fully automatic creation approach. The automatic functionality is part of the tool "InfoRapid KnowledgeBase Builder" [INF], which has a feature-rich tool-set to create maps and extend them both manually and with an automatic text analysis module. The tool was used for comparison with a caveat though, because the relations are not labeled. Since only concepts are used for the distance function (see Section 5.2.1), the tool should nevertheless suffice as an example of a fully automatic approach for comparison. Unfortunately, no automatic tool referenced in other research experiments could be used, because the tools were either research prototypes and/or not easily accessible within a reasonable amount of time and effort.

Altogether, the following 55 concept maps were created during the user evaluation:

- Manual concept maps: 5 texts by 5 different users = 25 maps
- Semi-automatic concept maps: 5 texts by 5 different users = 25 maps

- Automatic concept maps: 5 texts = 5 maps

These collected maps result in the subsequent comparisons:

- Each manual map compared to the corresponding semi-automatic map created by the same user (5 times accuracy and 5 times efficiency).
- Each semi-automatic map compared to its corresponding automatic map (5 times accuracy).

After all three versions of the created concept maps were available, it was possible to evaluate hypothesis *H1* by calculating the similarities for both the semi-automatic, as well as the automatically created maps against the gold standard manual maps. The similarity of the maps to the gold standard should verify which maps are closer to the ground truth and the intended meaning of the test users. *H1* states that the semi-automatic maps should have a significantly higher similarity to the gold standard than the fully automatic maps. The measures taken for this resemble metrics used in "Automatic comparison against references" and are explained in Section 5.2.1. For the evaluation it was not feasible though, to take the measurements automatically, because only one of the systems under test, i.e., the semi-automatic prototype, would have a fully accessible API.

With the time measurement for creating the concept maps, the questionnaire and the "Heuristic Evaluation" it should be possible to assert whether hypothesis *H2* can be verified as true. Additionally, measurements for precision and recall were calculated to ensure that the semi-automatic features of the prototype actually support or cause the supposed increase of efficiency.

The answers for these, mostly qualitative, evaluation techniques were consolidated and should provide good insights to the usefulness of the semi-automatic approach. The techniques used here belong to techniques of "Manual (comparative) quality judgements", because users qualitatively judge their experience of employing a semi-automatic approach in comparison to a manual gold standard.

5.2.1 Measurements and Calculations

Accuracy

The measure of the concept map similarity is based on the neighbourhood similarity proposed by McClure et al. [MSS99], which is described in Section 5.1. A slightly different similarity metric was used for this thesis, without calculating a direct neighbourhood similarity. The neighbourhood similarity of McClure et al. [MSS99] is only possible, if the same set of concepts is used for every map. This requirement was not fulfilled in the test setup, because the test users were not restricted to using a fixed set of concepts for neither the manual nor the semi-automatic concept maps. Instead, the map similarity was calculated by using the Jaccard-Index similarity over the whole set of concepts

available in the manual, semi-automatic, and automatic approaches respectively. The Jaccard-Index similarity was calculated for the sets of concepts used in manual (A) vs. semi-automatic (B) and manual (A) vs. automatic (B) approaches:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}. \tag{5.1}$$

In order to conduct this calculation, concepts from both sets had to be matched, even though no predefined sets of concepts were given to construct the concept maps. This was done with a simple sequence of steps: At first, all possible concepts were extracted manually from each text as detailed as possible. If a concept was used in a map, it was added to the set of concepts, A or B, for each respective map. If a concept was part of a compound concept in the map, it was also added to the set of concepts for this map. With these steps, sets of concepts were created, which could be matched, intersected, and unions of sets could be built. The resulting Jaccard-Index similarity gives a measurement about the accuracy of the semi-automatic and the automatic approach in comparison to the manual gold standard.

To show an example of this process, we consider the compound concept "UK Prime Minister David Cameron". This compound concept would be split up into the more detailed concepts "United Kingdom", "Prime Minister", and "David Cameron". One user might now use the single compound concept "UK Prime Minister David Cameron" in his concept map. The second user might use the two concepts "Prime Minister" and "David Cameron" as single concepts and not use the concept "United Kingdom" at all.

Now, the sets A and B can be created. The set A for the first users' concept map would then contain all three detailed concepts $A = \{$*"United Kingdom", "Prime Minister", "David Cameron"*$\}$ and set B for the second users' map would contain only two concepts: $B = \{$*"Prime Minister", "David Cameron"*$\}$. With the intersection of both sets $A \cap B$ and the union of both sets $A \cup B$, we can calculate the Jaccard-Index $J(A, B)$ as follows:

$$A \cap B = \{\textit{"Prime Minister", "David Cameron"}\}. \tag{5.2}$$

$$A \cup B = \{\textit{"United Kingdom", "Prime Minister", "David Cameron"}\}. \tag{5.3}$$

$$|A \cap B| = 2. \tag{5.4}$$

$$|A \cup B| = 3. \tag{5.5}$$

$$J(A, B) = \frac{2}{3} = 0.67. \tag{5.6}$$

Precision and Recall

Precision and recall were calculated to measure the usefulness of the NLP pipeline for extracting relevant concept and relation suggestions. For each analyzed text in the NLP pipeline, the number of suggested concepts and relations was recorded. Additionally, for each user and each created map, the number of used concepts and relations, as well as the concept and relation suggestions that were actually chosen by each user, were noted.

From these values, the concept precision (**CP**) was calculated by dividing the used concept suggestions (**UCS**) by the number of extracted concept suggestions (**ECS**). Note that in **ECS** extracted pronouns were not counted, since they are not shown as suggestions to the user and only used for highlighting in the "text area":

$$CP(UCS, ECS) = \frac{|UCS|}{|ECS|}. \tag{5.7}$$

The same calculation was done for the relation precision (**RP**) by dividing the used relation suggestions (**URS**) by the number of extracted relation suggestions (**ERS**):

$$RP(URS, ERS) = \frac{|URS|}{|ERS|}. \tag{5.8}$$

The concept recall (**CR**) was calculated by dividing the used concept suggestions (**UCS**) by the number of used concepts (**UC**) in each respective map:

$$CR(UCS, UC) = \frac{|UCS|}{|UC|}. \tag{5.9}$$

Again, the same calculation was conducted for the relation recall (**RR**) by dividing the used relation suggestions (**URS**) by the number of used relations (**UR**) in each respective map:

$$RR(URS, UR) = \frac{|URS|}{|UR|}. \tag{5.10}$$

To show an example of these calculations, we can again use the sample text:

"*The quick brown fox jumped over the lazy dog. Then it ran across the orchard. The lumberjack had known the fox for some time*".

The semi-automatic CMM prototype would extract the concepts (**ECS** without the pronoun concept "it") and relations (**ERS**) that can be seen in Table 5.1. A user could create the concept map that can be seen in Figure 5.1. Counting the used concept (**UCS**) and used relation suggestions (**URS**), as well as the used concepts (**UC**) and used relations (**UR**) from Figure 5.1, we can record the values in Table 5.1.

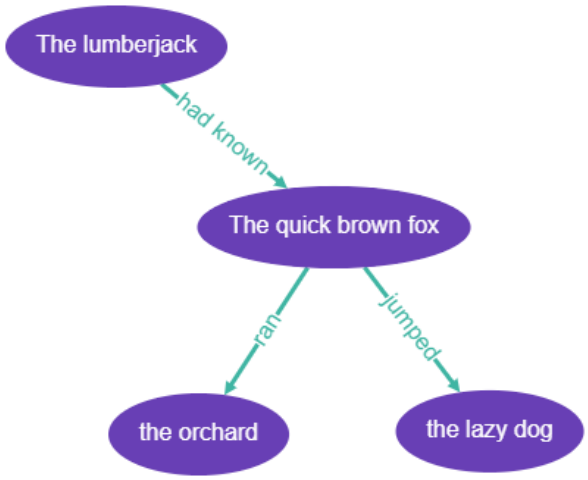

Figure 5.1: Sample concept map for the calculation of concept and relation precision and concept and relation recall.

	Count	Description
Extracted Concept Suggestions (**ECS**)	6	"The quick brown fox", "the lazy dog", "the orchard", "The lumberjack", "the fox", "some time"
Extracted Relation Suggestions (**ERS**)	6	("The quick brown fox") "jumped" ("the lazy dog") ("The lumberjack") "had known" ("the fox") ("The quick brown fox') "ran" ("the orchard") ("The lumberjack") "had known" ("The quick brown fox") ("the fox") "jumped" ("the lazy dog") ("the fox") "ran" ("the orchard")
Used Concept Suggestions (**UCS**)	4	"The quick brown fox", "the lazy dog", "the orchard", "The lumberjack"
Used Relation Suggestions (**URS**)	3	("The quick brown fox") "jumped" ("the lazy dog") ("The lumberjack") "had known" ("the fox") ("The quick brown fox") "ran" ("the orchard")
Used Concepts (**UC**)	4	"The quick brown fox", "the lazy dog", "the orchard", "The lumberjack"
Used Relations (**UR**)	3	("The quick brown fox") "jumped" ("the lazy dog") ("The lumberjack") "had known" ("the fox") ("The quick brown fox") "ran" ("the orchard")

Table 5.1: Recorded counts for the sample concept map to calculate concept and relation precision, as well as concept and relation recall.

Using the counts in Table 5.1, we can calculate the precision and recall measures as follows:

$$CP(UCS, ECS) = \frac{4}{6} = 0.67. \tag{5.11}$$

$$RP(URS, ERS) = \frac{3}{6} = 0.5. \tag{5.12}$$

$$CR(UCS, UC) = \frac{4}{4} = 1. \tag{5.13}$$

$$RR(URS, UR) = \frac{3}{3} = 1. \tag{5.14}$$

5.2.2 Questionnaire

The following questions were asked each user after completing the test concept maps with the semi-automatic prototype:

Key	**Question**	**Possible Answers**
Q1	Have you heard of the term “concept map” before?	yes / no
Q2	Have you heard of the term “mind map” before?	yes / no
Q3	What is your experience level in user interface design?	1-4
Q4	Was it more or less efficient for you to construct a concept map in comparison to doing it fully manually?	more / less
Q5	How much did the concept suggestions in the sidebar help?	1-4
Q6	How much did the highlighting in the text help?	1-4
Q7	How much did the concept suggestions in the canvas help?	1-4
Q8	How much did the relation suggestions in the canvas help?	1-4
Q9	How was the experience with using the “Concept Map Toolbox”?	freetext

Table 5.2: Questions and possible answers for the questionnaire.

The possible answers 1-4, with 4 being the best score, in the Table 5.2 for *Q3* are broken down into:

1. No experience at all,
2. Have heard the term “user interface” before,
3. Have designed a user interface myself, and
4. Have knowledge in user interface design, and designed several user interfaces.

The possible answers 1-4, with 4 being the best score, in the Table 5.2 for *Q5* - *Q8* are broken down into:

1. Not at all,
2. Helped me a little,
3. Helped me more than a few times, and
4. Helped me significantly.

5.2.3 Heuristic Evaluation

Another important part of the user evaluation was to measure the usability of the prototype application and to gain additional insights into possible usability problems and reveal features that may be crucial for efficient semi-automatic concept map creation. For this purpose, a discount usability engineering method, called "Heuristic Evaluation", originally developed by J. Nielsen [Nie92], was used. With this method, it is possible to find up to 80% of usability problems in a user interface, by employing only five expert evaluators that are familiar with user interface design and discovering usability problems. Nine of the ten usability heuristics by J. Nielsens [Nie94] were used to conduct the heuristic evaluation. The following heuristics needed to be inspected:

HEU1 - Visibility of system status: It should always be visible to the user in which status the system currently resides. For instance, if the system currently loads data and is blocked, the user should see an indicator for the current load action.

HEU2 - User control and freedom: Users should be able to easily undo and redo their user interactions.

HEU3 - Consistency and standards: Graphic elements and controls should have the same meaning and use the same terminology in different parts of the user interface.

HEU4 - Error prevention: Errors should be prevented as much as possible. Interactions that could produce errors should be displayed accordingly.

HEU5 - Recognition rather than recall: While users operate the user interface, they should only be exposed to relevant information for the current task at hand. They should not be forced to recall information from, e.g., a previous dialog and should rather be able to recognize important information based on the current context.

HEU6 - Flexibility and efficiency of use: Users that have more experience in using a system should have advanced interaction possibilities that enable faster navigation and completion of tasks, like keyboard shortcuts, advanced controls, or user macros.

HEU7 - Aesthetic and minimalist design: A user interface should be limited to the most necessary controls and elements. Unnecessary additional elements introduce visual clutter and can lead to limiting the users mental capacity and even introduce distraction from the task to fulfill.

HEU8 - Help users recognize, diagnose, and recover from errors: If errors do occur, it should be easy for users to understand the error and return to normal system use, without requiring expert knowledge or understanding expert terminology.

HEU9 - Help and documentation: It should be possible to use the system without documentation and the user interface should generally be self explanatory. But, if help is needed, the documentation should be easily accessible and visible.

One heuristic **HEU10 - Match between system and the real world** was left out from the heuristic evaluation on purpose. An examination of the match to the real world, in terms of user interface elements and real world conventions, did not seem to be necessary to be evaluated for such an abstract prototype application, for which the only task is to construct concept maps.

After the second user trial, the prototype was to be carefully checked against fulfilling the heuristics through the experience gained by the test users. The heuristics were to be investigated by giving a score of 1 - 4, with 4 being the best possible score, in order to generate a quantitative measurement. Furthermore, each evaluator gave a free text statement on the compliance of the prototype with the heuristics, if necessary, and was encouraged to propose additional improvements on all aspects of the implementation.

5.3 Results

Three example concept maps are shown below, which were created during the evaluation phase for the semi-automatic prototype. Figure 5.2 shows a map that was manually created with the tool “MindMup” [MIN]. The map in Figure 5.3 was created with the semi-automatic prototype and in Figure 5.4 with “InfoRapid KnowledgeBase Builder” [INF].

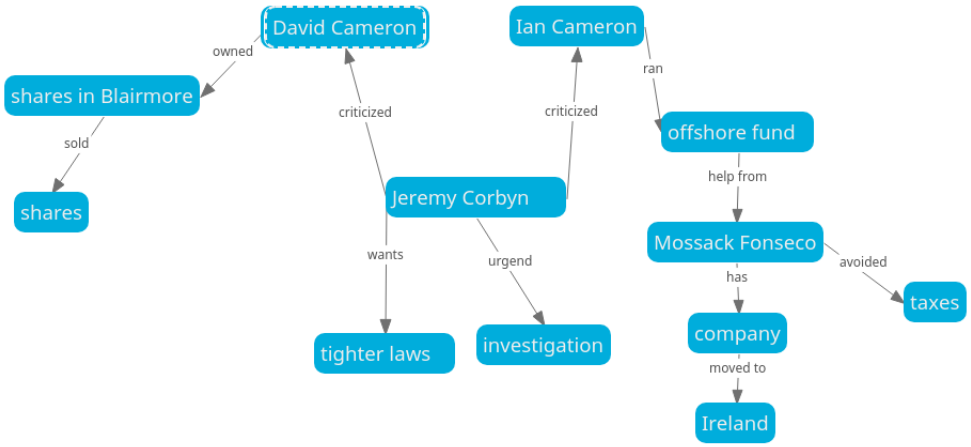

Figure 5.2: Manually constructed concept map with “MindMup” - Text 5, User 4.

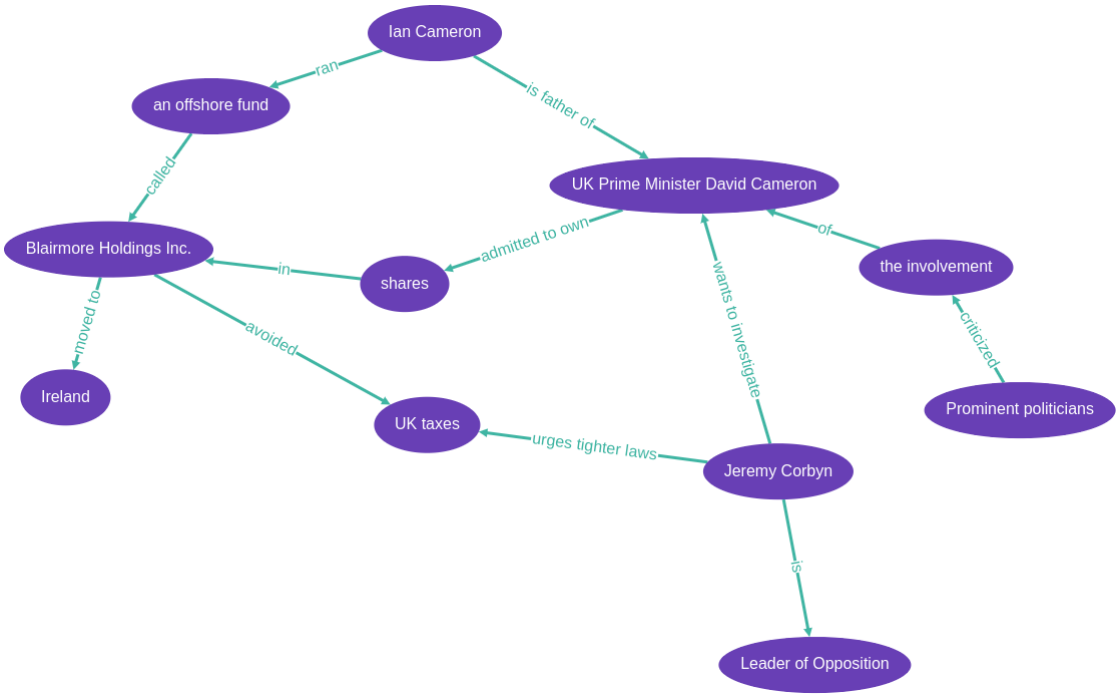

Figure 5.3: Concept map constructed with semi-automatic prototype - Text 5, User 4.

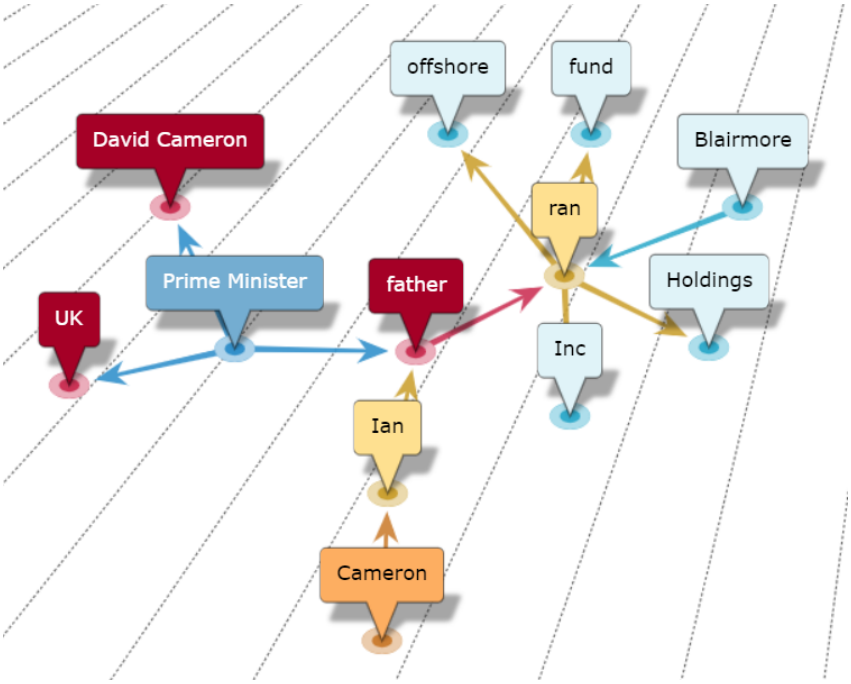

Figure 5.4: Automatically constructed concept map with “InfoRapid KnowledgeBase Builder” - Text 5.

5.3.1 Validating H1

Calculations for the similarity of the concept maps created with the semi-automatic and the automatic approach, in comparison to the manual gold standard, can be seen in Table 5.3. *U1M1* is to be read as user one, map one.

	Semi-Automatic vs. Manual	**Automatic vs. Manual**	**Difference**
U1M1	0.76	0.31	0.45
U2M1	0.68	0.27	0.41
U3M1	0.89	0.28	0.61
U4M1	0.56	0.33	0.23
U5M1	0.71	0.38	0.33
Avg. M1	**0.72**	**0.31**	**0.41**
U1M2	0.61	0.26	0.35
U2M2	0.61	0.24	0.37
U3M2	0.50	0.23	0.27
U4M2	0.55	0.39	0.16
U5M2	0.44	0.22	0.22
Avg. M2	**0.54**	**0.27**	**0.27**
U1M3	0.73	0.20	0.53
U2M3	0.52	0.14	0.38
U3M3	0.65	0.17	0.48
U4M3	0.65	0.29	0.36
U5M3	0.68	0.10	0.58
Avg. M3	**0.65**	**0.18**	**0.47**
U1M4	0.90	0.27	0.63
U2M4	0.80	0.44	0.36
U3M4	0.80	0.30	0.50
U4M4	0.80	0.27	0.53
U5M4	0.67	0.33	0.34
Avg. M4	**0.79**	**0.32**	**0.47**
U1M5	0.83	0.32	0.51
U2M5	0.53	0.46	0.07
U3M5	0.65	0.35	0.30
U4M5	0.44	0.27	0.17
U5M5	0.63	0.33	0.30
Avg. M5	**0.62**	**0.35**	**0.27**
Avg. Total	**0.66**	**0.29**	**0.37**

Table 5.3: Jaccard-Index similarity of all evaluated concept maps.

The results clearly show that the semi-automatic approach can generate maps with a higher similarity of 0.66 to the manual maps on average than the automatic approach with an average similarity of 0.29. That amounts to an average similarity difference of 0.37, which points to a considerable advantage in terms of accurately representing the user intention for the semi-automatic prototype implementation. A boxplot representation of the similarity measures, as seen in Figure 5.5, visually confirms the findings. Additionally, a students' t-test with $\alpha = .05$ was conducted to test the significance of the results, where the null hypothesis states that the accuracy means of the manual and semi-automatic approaches are the same. The 25 concept maps created with the semi-automatic approach ($M = 0.66$, $SD = 0.13$) compared to the 25 concept maps created with the automatic approach ($M = 0.29$, $SD = 0.09$) demonstrated a significantly higher accuracy, $t(48) = 12.18$, $p < .001$.

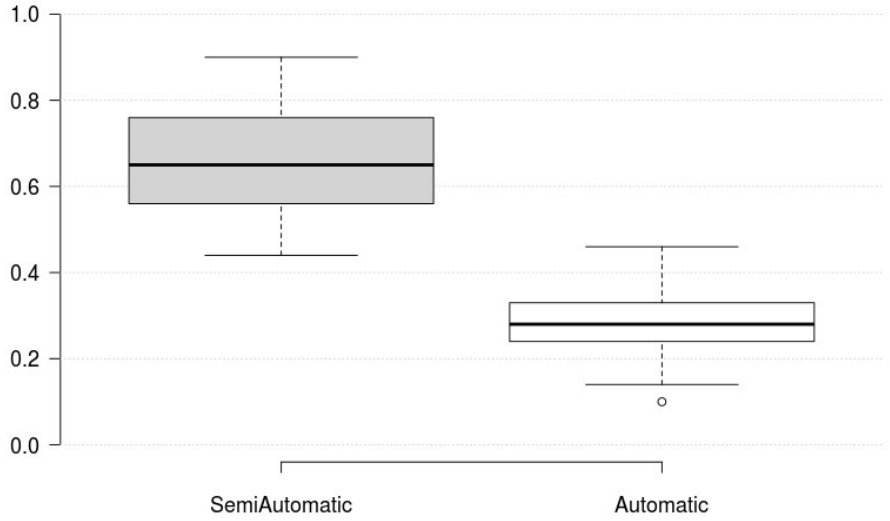

Figure 5.5: Boxplot of accuracy of semi-automatically and automatically created concept maps compared to manually created concept maps.

One could argue that even an average accuracy of 0.66 is not really high and that the same users should be able to create almost identical concept maps, if using a manual tool or a semi-automatic tool. This may be on the one hand attributed to the fact that there was an intended one month time gap between the creation of the manual gold standard maps and the recreation with the semi-automatic prototype implementation. Therefore, users had to reread and relearn the contents of the test texts for concept map creation. On the other hand, we know that a user does not even always create exactly the same concept map for the same underlying topic [HOJS99]. Another aspect could be that the users were biased to select different concepts because of the already suggested concepts to choose from. Although they had the option to create their own concepts manually, it was more convenient to choose from the suggested concepts in most instances, which is also reflected in Section 5.3.2.

Overall, *H1* is supported with the implemented semi-automatic prototype. Tools for concept map creation that implement a semi-automatic approach can lead to a higher accuracy than fully automatic approaches, if reflecting the user intention as best and efficient as possible is the main purpose.

5.3.2 Validating H2

Precision and Recall

The main features of a semi-automatic concept map creation approach are meaningful suggestions of concepts and relations that can be used to construct the maps from the underlying text document. In order to present the user with many meaningful concepts and relations to choose from, we want to make sure to return as many relevant results as possible and not miss important concepts or relations, even when they will not be used in the final concept map.

In other words, recall assesses how many of the concepts and relations in the user maps were created from suggestions, while precision assesses how many suggested concepts and relations were used in the map. Therefore, a high recall is more important than a high precision. Table 5.4 shows the results for precision and recall.

	CP	**RP**	**CR**	**RR**
Avg. Text 1	0.25	0.09	0.71	0.19
Avg. Text 2	0.33	0.14	0.85	0.36
Avg. Text 3	0.32	0.04	0.88	0.15
Avg. Text 4	0.54	0.09	1.00	0.25
Avg. Text 5	0.31	0.10	0.92	0.32
Avg. Total	**0.35**	**0.09**	**0.87**	**0.25**

Table 5.4: Average concept precision (CP), relation precision (RP), concept recall (CR) and relation recall (RR).

Precision results are rather low, with 0.35 CP and 0.09 RP. This is more or less what was expected from the NLP pipeline, because the goal was not to return only correct results, but provide the user with many possibilities to choose from. Therefore, the number of returned correct concepts and relations is low compared to the overall returned results from the text mining NLP algorithm.

Recall results show a good score for the concepts of 0.87 CR, which was intended. This means that almost 90% of the concepts in the users' concept maps were suggested by the semi-automatic prototype. The relation recall is below expectations, with the RR score being 0.25. This means that only a quarter of the extracted relations were actually considered by the users and could be used in their maps.

The precision and recall results indicate that the semi-automatic prototype can indeed deliver meaningful concept suggestions for the map creator. A consistent extraction of

meaningful relation suggestions, however, cannot be verified. Considering hypothesis *H2*, this could lead to the conclusion that suggesting meaningful concepts is the more crucial feature for making a semi-automatic concept map creation approach more efficient than a manual approach. Another explanation would be that the part of the CMM algorithm for extracting relation suggestions is simply not performing well and suggests relations which do not fit to the section of the concept map the user is currently building. The part of the CMM algorithm to extract meaningful relations and build *subject-verb-object* triples is also certainly one of the most challenging parts of creating and summarizing textual data to concept maps.

Duration

Table 5.5 shows the length and word count of the sample texts and the average duration it took all users to create maps with the manual tool and the semi-automatic prototype implementation.

	Text Length in Characters	**Word Count**	**Avg. Duration Manual in Seconds**	**Variance Manual**	**Avg. Duration Semi-Automatic in Seconds**	**Variance Semi-Automatic**
Text 1	622	103	640	5487	612	23451
Text 2	913	137	1028	402795	654	50580
Text 3	843	129	939	138415	716	25973
Text 4	461	67	349	11761	233	4085
Text 5	622	100	676	123307	442	25709
Avg. Total			**726**	**173605**	**531**	**53417**

Table 5.5: Text length in characters and word count. Average duration (in seconds) for manual and semi-automatic concept map creation. Variance for duration of manual and semi-automatic concept map creation.

The results show that, on average, the map creation time could be reduced by 195 seconds or 3 minutes and 15 seconds, if using the semi-automatic prototype. This results in an average time saving of 27%. A boxplot comparison of the manual and semi-automatic durations in Figure 5.6 visualizes the findings. A students' t-test with $\alpha = .05$ shows the significance of the results, where the null hypothesis states that the means of the manual and semi-automatic creation duration are the same. The 25 concept maps created with the semi-automatic approach (M = 531, SD = 231) compared to the 25 concept maps created with the manual approach (M = 726, SD = 417) exhibited a significant reduction of the creation duration, $t(48) = 2.05$, $p = .046$.

For validating hypothesis *H2*, this means that the semi-automatic approach is more efficient than a manual approach, if construction duration is considered. *H2* is therefore supported. Qualitative results below can reinforce these findings.

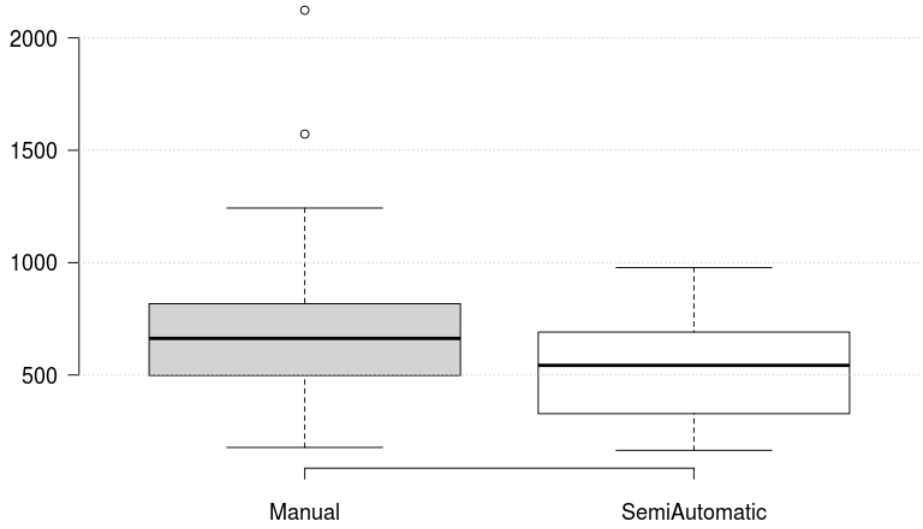

Figure 5.6: Boxplots of completion times (in seconds) for manual and semi-automatic concept map creation.

Questionnaire

The questionnaire results can be seen in Table 5.6.

	U1	**U2**	**U3**	**U4**	**U5**	**Overall / Average**
Q1	no	no	no	no	no	**no**
Q2	yes	yes	yes	yes	yes	**yes**
Q3	4	4	4	1	3	**-**
Q4	more	more	more	more	more	**more**
Q5	4	4	4	4	4	**4**
Q6	4	3	3	2	4	**3.2**
Q7	3	3	3	3	2	**2.8**
Q8	2	2	4	3	2	**2.6**

Table 5.6: Questionnaire results.

Results for **Q1** and **Q2** show that none of the test users had heard of the term "concept map" before (**Q1**), but all users knew the term "mind map" (**Q2**). This suggests that "concept map" is not a common term outside of research and "mind map" is the term used by most people. Also, none of the test users were associated with research of concept maps.

Answers for **Q3** represent the experience level of the test users with user interface design. Three of the test users were experts in user interface design and had in-depth knowledge

and designed several user interfaces. They were also aware of the common usability heuristics by J. Nielsen [Nie94]. One user had designed user interfaces, but was not an expert in user interface design and was not aware of any usability principles. The fifth user had no experience in user interface design and can be considered a naive user in terms of usability. Overall, with three usability experts and two more or less naive users, the experience requirements were fulfilled for conducting the heuristic evaluation.

Every user answered in **Q4** that it was more efficient to create the concept maps with the semi-automatic prototype than with the manual tool. Two users additionally answered that the increase in efficiency was mostly because of the suggested concepts and another one mentioned that concept map creation was also more "fun" than with the manual tool. Considering hypothesis *H2*, these results, although highly subjective but consistently good, reassure the finding of the duration results and in consequence the validation of *H2* as being *true*.

Questions **Q5 - Q8** refer to certain user interface elements that represent the key features of the semi-automatic prototype. Additionally, they should answer how much each of those features contributed to the usefulness of the prototype.

The concepts in the "suggested concepts sidebar" were unanimously scored in **Q5** (Figure 5.7) with the best possible score of 4 by all users and it helped them significantly in building their concept maps. As stated before, extracted concepts that serve as the basic building blocks for a concept map, seemed to be the most helpful feature of a semi-automatic concept map creation tool. One significant quote from a user for this question was: "*I tried to always use the suggested concepts, because then I did not have to write them manually*".

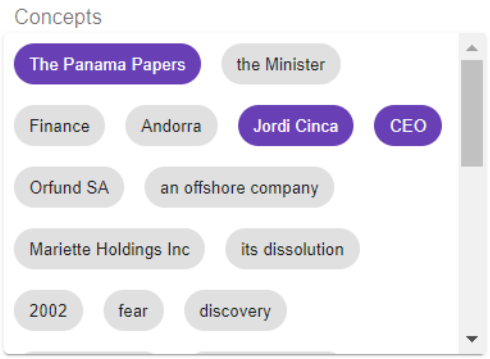

Figure 5.7: Q5 - Suggested concepts sidebar.

Results about the highlighted concepts and relations in the analyzed text can be seen in question **Q6** (Figure 5.8) and show a slightly more differentiated picture. Two users answered that the text highlighting helped them significantly, whereas it helped two other users more then a few times and it helped one user only a little. This results in an average score of 3.2, which is still a good score and means that the feature was overall useful. Three users stated independently that the text highlighting of already

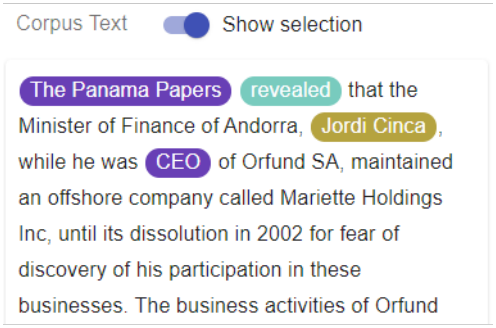

Figure 5.8: Q6 - Highlighted text.

used concepts and relations represented some sort of progress in building the concept map. Another added that, "*although I find the text highlighting useful, I sometimes had to disable the highlighting to read the text*". A different user also specifically mentioned that "*the highlighting of the concepts helped me, but the possibility to turn it off was really important for me*".

Q7 asked about the usefulness of showing newly suggested concepts directly in the map, if one concept is selected in the map. The suggested concepts can be seen in grey color in Figure 5.9.

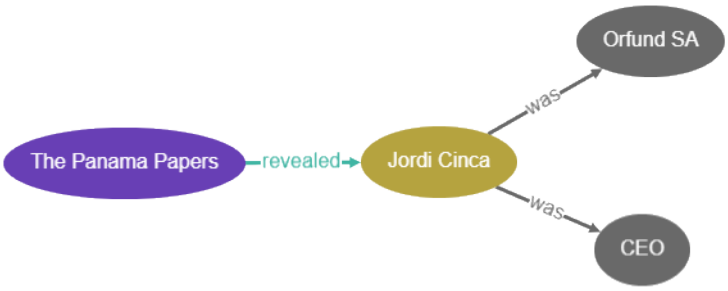

Figure 5.9: Q7 - Concept suggestion in the concept map.

Four out of five users, who answered with a score of 3, had felt that this feature helped them more than a few times, but not significantly. One user answered that it only helped a little (2), which results in an average score of 2.8. Altogether, this means that the feature was useful, but not at all times, when the users were creating their maps. For one user the concept suggestions were less helpful in the beginning, but more helpful at a later stage of building the map, when there were already more concepts and relations added to the map. Whereas for another user, the concept suggestions were better as a starting point. This particular user stated: "*They were helpful, but not always correct. If there were already a lot of concepts in the map, they were less helpful, but when a new*

node was suggested that had not already many relations to the existing map, I could use it. As a starting point for the map, the suggestions were really good". The user who gave the lowest score of 2 shared a similar experience: "*Sometimes the suggestions were really good, but sometimes did not help at all*". One user felt that the feature may be unnecessary for users that know the topic well: "*The suggestions of the concepts in the graph, I think, are really important when you don't understand the text. But may be a little obtrusive if you know exactly what you want to do*". These experiences align with the concept precision and recall results, where the concept suggestions are not always made to be precise, but instead offer a variety of options to choose from. Furthermore, the feature could also stand in the users' way, if the user knows exactly which concept map he or she wants to build and which step he or she wants to take next, while iteratively creating the map.

Figure 5.10: Q8 - Relation suggestion in the concept map.

Results for **Q8**, which examined the feature of suggesting new relations inside a map if at least two concepts are already selected, show a slightly worse, but similar, outcome to the results of **Q7**. A suggested relation is visible in grey color in Figure 5.10. Three users answered that the relation suggestions helped them a little (2), whereas one answered they had helped them more then a few times (3), and one that they had helped them significantly (4), with an average score of 2.6. In consequence, the feature could be considered as useful only in some instances. Two users answered independently that for them, often the relation label was right, but in the wrong direction. Additionally, one user answered that he wanted the label of the relation written slightly different than it was proposed: "*I often had to manually create relations because I wanted to name them in my way. And also the direction was not right for me sometimes*".

These statements also align with the relation precision and recall results, which reveal that not many suggested relations were actually used in the final maps of the users. The statements may provide an explanation for the rather low score for suggested relations. Users often want to express their propositions in a specific way. If relations are extracted that have the same semantic meaning as their intended relation, but use a different verb, the users often still want to use their specific verb. Additionally, they might want to use the extracted direction of the relation in reverse, even though it was extracted in the syntactically correct direction.

Q9 was an open question, where users could freely add additional findings from their experience using the semi-automatic prototype. The answers to the open question confirm the overall positive sentiment for working with the semi-automatic prototype:

Q9U1

- "*It works but was a little cumbersome in the beginning. After a few maps it started to work better. I discovered some usability problems and potential improvements. Over all I thought it was more efficient, especially the suggested concepts.*"
- "*The UI was definitely more intuitive than the manual tool. The features that are necessary to construct the map were available and nothing more.*"
- "*It is cool that the tool offers an ability to manually create a concept map and also the semi-automatic features. Both ways are possible.*"

Q9U2

- "*I would use the tool again. It is definitely more efficient than the manual tool. I always tried the suggestions, which helped me a lot, even though I could not find the right relation and concept every time.*"

Q9U3

- "*It was cool that the concepts are already provided (suggested) on the right side and it was definitely more efficient than with the manual tool.*"
- "*I was a little confused at the start that I had too many choices for the same concept. If I wanted to add a different concept, I have the ability to add it manually. Reducing this to one concept to choose from would have been better for me.*"

Q9U4

- "*I thought it was good that I could directly work with and see the text.*"
- "*The suggested concepts were really helpful and that I could use them immediately.*"
- "*It is a good tool to filter out the most important part of a text. I would use the tool again and I think it would be a good way to support and facilitate learning.*"

Q9U5

- "*I think it works well, but I also encountered several issues. For instance: It would have been better if there would be only one suggested concept for one concept in the text, like only "the opposition leader Jeremy Corbin" or only "Jeremy Corbin".*"
- "*The suggestions were really helpful for me.*"
- "*The tool is extremely helpful when I have a text that I don't know at all.*"

Heuristic Evaluation

Results for the heuristic evaluation can be seen in Table 5.7 and attest the semi-automatic prototype a very good usability with an average score of 3.5 over all usability heuristics. All but three scores are equal to or higher than 3, which indicates only minor usability problems. Encountered usability issues and user comments are listed below for each usability heuristic in detail. Some comments are to be taken with a caveat, because the semi-automatic tool is simply a prototype and not a finished product. Therefore, some convenient features that would normally be part of a polished software product, could not be implemented into the prototype in time.

	U1	**U2**	**U3**	**U4**	**U5**	**Overall**
HEU1	4	3	3	3	4	**3.4**
HEU2	4	3	4	4	3	**3.6**
HEU3	3	4	3	4	4	**3.6**
HEU4	4	4	4	3	4	**3.8**
HEU5	3	4	3	4	4	**3.6**
HEU6	3	2	3	4	3	**3.0**
HEU7	4	2	4	4	4	**3.6**
HEU8	4	4	4	4	4	**4.0**
HEU9	2	3	3	3	4	**3.0**
Total	**3.4**	**3.2**	**3.4**	**3.7**	**3.8**	**3.5**

Table 5.7: Heuristic evaluation results.

HEU1 - Visibility of system status achieved an average score of 3.4. A UI element the users specifically mentioned for this heuristic was the display of the analyzed text at the right side along with highlighting already selected concepts and relations. It was positively mentioned that the text for the concept map was always visible, while building the map:

- "*It was really good that I always saw the text and did not have to jump between different (browser) tabs.*"

Issues users found had to do with the lack of visibility of the possible actions, when clicking the "show suggestions" and "add relation" buttons:

- "*It was initially not clear to me that I had to click on a concept, when I clicked on the "show suggestions" button in order to see the concept and relation suggestions. Although it was better that only suggestions for the currently selected concept were shown.*"
- "*When clicking the "add relation" button, the cursor should maybe change to indicate that it is possible to do something different.*"

Another user, who is a professional in graphic design, mentioned the used colors for the selected concepts and relations in the text highlighting:

- "*The colors in the text were not always clear to me (what did they symbolize).*"

A different user suggested that it would be better to underline selected concepts and relations in the text instead of highlighting them with a colored background:

- "*Maybe underline the selected concepts and relations, because the text gets somewhat unreadable when many concepts are selected.*"

HEU2 - User control and freedom was evaluated with an average score of 3.6, which suggests no serious usability issues for this heuristic. The prototype has no explicit undo and redo features, but other options to undo previously executed steps, like deleting a concept. User comments reflect these findings:

- "*Undo was not necessary for me, until you have an automatic layout (feature), then you need an undo action.*"
- "*I did not miss much.*"

However, the prototype lacked the functionality to save the current state of the application, which one user noted specifically:

- "*The ability to save my progress was missing for me.*"

HEU3 - Consistency and standards achieved a good average score of 3.6. This indicates only minor usability problems. Issues that the users mentioned were:

- "*When renaming a concept and I don't actually rename the concept and click "OK", the dialog should accept it.*" There are two ways to address this problem: Give users visual feedback that nothing changed and therefore the "OK" button does not submit the dialog or let the user click "OK", without changing anything.
- A subtle consistency error in the shape of user controls was detected by one user: "*The form of the map controls had the same shape as the suggested concepts and the concepts in the canvas had a different shape.*"
- For one user, it was not clear that the currently selected concept was colored ocre in the map and in the highlighted text: "*The only thing I did not understand was at map 4, where the "It" concept was colored in ocre.*"

HEU4 - Error prevention achieved a score of 3.8. The prototype overall has a good stability, but one user was able to provoke an error that crashed the application, while constructing one map, and therefore gave a slightly lower score of 3.

HEU5 - Recognition rather than recall also achieved a good average score of 3.6. However, there were a few problems that the users encountered, but mostly deemed not as critical usability issues, since the lowest score given for this heuristic is 3. One problem that was still persistent in the prototype was the not optimal way of inserting selected concept suggestions into the current concept map. Suggested concepts are not inserted at the same position in the map every time, which turned out to be an issue for most test users:

- "*When selecting a suggested concept from the list on the right side, it would be good to always have the same position for a newly inserted concept.*"
- "*The concepts should always be inserted near my visible view port, sometimes I had to search them.*"

Another problem was that it was not clear for every user how to actually insert a text to the tool to get analyzed, since the button is not positioned optimally:

- "*The "add text" button should be (in the area) where the text will get inserted.*"
- "*It was not clear to me in the beginning that I had to click the "add text" button, to insert a text.*"
- "*The position of the "add text" Button was not optimal for me. Maybe a positioning nearer to the text would be good.*"

A major flaw that the prototype still has, is that manually inserted concepts and relations, or renamed concept and relations, lose the coupling to the concept and relation suggestions and to the highlighting in the text:

- "*If I rename a concept just a little (maybe just change a part of it), the suggestions should still work.*"

One user mentioned that it would be better to use a different control for the "show suggestions" button, because the button changes the way a concept behaves in the concept map, when it is clicked (it shows suggestions in addition to being selected):

- "*The "show suggestions" button maybe would be better as a checkbox, because it changes the behaviour when clicking on a concept.*"

HEU6 - Flexibility and efficiency of use is one of the heuristics with the lowest score of on average 3. Hence, the users had the most proposals for possible improvements concerning this heuristic, since they became proficient in using the prototype during the course of the trial and demanded advanced user control features. The encountered problems and suggested enhancements are listed below without much further explanation, since all user comments have a valid point:

- "*I would like to submit all my actions with pressing the enter key, this was not possible with most user interface elements.*"
- "*I want to be able to rename concepts and relations with a double click.*"
- "*I always want to drag and drop the suggested concepts to the canvas. In the beginning, it was not clear to me that I had to click them.*"
- "*It should be possible to enlarge or resize the area for suggested concepts.*"
- "*I would suggest an improvement, when I try to select similar concepts. I would prefer concepts which have more relations, so an indicator on how many relations are available, like a number after the concept name, would be helpful.*"
- "*Much more keyboard actions and shortcuts would be really helpful. / Keyboard shortcuts would be essential if this tool would be released.*"
- "*Moving the relations directly would improve the tool in my opinion.*"
- "*Copying relations or adding the same relation to multiple concepts at the same time.*"
- "*The ability to resize nodes would be great.*"
- "*"Magnetic" alignment of Concepts would be a cool feature.*"
- "*Maybe adding relations with a right click would be better, because the mouse cursor is nearer to my scope of action.*"
- "*Sometimes I missed the feature to move existing relations to other nodes.*"
- "*If I click on the text it would be helpful that a suggested concept (in the sidebar) is selected or marked.*"

One user, who gave one of the lowest scores (2), summarized the overall problem with this heuristic precisely:

- "*The problem with graph tools is that they are never efficient enough. There are always features like reversing the direction of relations, changing the size of nodes that may be important. But it may be out of scope of this study and could be part of the future work.*"

HEU7 - Aesthetic and minimalist design and the requirement to minimize visual clutter was one of the key concepts for developing the user interface of the semi-automatic prototype application. This can also be seen in the good score of on average 3.6. Overall, the users thought that the user interface was minimalist and incorporated only the most necessary controls and elements:

- "*I like that there were little choices in the user interface. Only the necessary elements were there. But nothing more. In the manual tool I was at first overwhelmed with the choices.*"
- "*The user interface was rather simple and not too overloaded.*"

The graphic design professional thought that the colors for the user interface elements were nothing special, but were also not a bad choice. Interestingly, one other user, who gave the lowest score of 2, thought that the choice of colors was not optimal:

- "*Minimalist yes, aesthetic no - the colors were not optimal.*"

HEU8 - Help users recognize, diagnose and recover from errors achieved the best possible score of 4. Except one crash of the application, mentioned in **HEU4**, no other errors or exceptions were encountered by the users. Also, the user that encountered the crash, was able to reload the application without help and therefore also answered with the best possible score of 4.

HEU9 - Help and documentation also achieved one of the lowest scores with an average of 3. It is important that a user interface needs as little help and documentation as possible and can ideally be used without any. Nevertheless, most users missed a small introduction to all the relevant features of the prototype. For some users this was more important than for others:

- "*At first, I did not recognize the option to show suggestions. Maybe an optional tutorial dialog at the start would be nice.*"
- "*If there would be a short introduction or mini tutorial it would be good, otherwise help is not needed.*"
- "*A documentation was not really necessary because the system was easy to use.*"
- "*An introductory mechanism would be helpful that explains all the features.*"
- "*Maybe a small intro or wizard would have been good, but I did not need a documentation, because everything was self explanatory.*"

CHAPTER 6

Conclusion

In this last chapter, a summary of the research, findings and evaluation results of this thesis are given in Section 6.1. Additionally, potential improvements on the semi-automatic CMM approach and the created prototype implementation, as well as further research possibilities, are described in Section 6.2.

6.1 Summary of Findings

In this thesis, a new semi-automatic CMM approach on unstructured text was introduced. This approach aims to reduce the tediousness of fully manual concept map creation and make the process more efficient, while preserving the user-intended subjectivity of the created concept maps, other than a fully automatic source-text based system. The main contributions of this thesis to scientific research in the area of CMM can be summed up as follows:

1. A web-based semi-automatic CMM tool was developed that extracts concept and relation suggestions from a single text document and allows a user to efficiently create and extend a concept map in a user interface.
2. Results from a user study, which show that semi-automatic CMM can deliver a higher accuracy than fully automatic CMM compared to a manually created gold standard, but is more efficient than fully manual concept map creation.

For this approach, we defined a new CMM process. This process consists of two main steps and constitutes the semi-automatic nature of the approach: “Automatic Text Processing” and “Manual Concept Map Construction”.

The “Automatic Text Processing” step analyzes a submitted text with a state-of-the-art NLP pipeline and extracts concepts and relations as propositions. This extraction is

realized with custom algorithms that use the NLP annotations that were created by the NLP pipeline and coreference mentions, which were extracted with a neural network based Mention-Ranking coreference scoring model [NEU].

The extracted concepts and relations are presented as suggestions in the user interface of the "Summarization" and "Topological Arrangement" parts of the second "Manual Concept Map Construction" step. The user interface was intended to facilitate a minimalist and clutter free manual creation of concept maps, while providing the user with extracted concept suggestions to jump-start the construction of a concept map. Additionally, the suggestions can be shown and chosen in the "canvas" of the user interface, when the map is already filled with concepts and relations. Another crucial part of the semi-automatic CMM approach is the displaying and highlighting of the underlying text in the "text area", to always maintain a visual connection between the "canvas", the concept and relations suggestions and the underlying text.

The semi-automatic CMM approach was implemented as a prototype web-application with state-of-the-art web technologies in a simple client-server architecture. We implemented this prototype in order to validate the developed semi-automatic CMM approach. It currently provides a usable and good-performing system for text documents that are composed of up to 750 words. For longer text documents, the system has some scaling issues. Performance and usability tend do degrade with longer texts. Furthermore, it still has some problems with the positioning of concept suggestions in the "canvas" and matching of manual concepts and relations to the extracted ones.

A user evaluation with five users was conducted that combines qualitative and quantitative measures for verifying the usefulness of the proposed semi-automatic CMM approach. The five test users first each manually created a set of five concept maps from sample texts, which served as the gold standard. Afterwards, concept maps from the same texts were created with the semi-automatic CMM prototype implementation. Additionally, concept maps from the same sample texts were created with a fully automatic tool.

The resulting concept maps were manually compared and similarity of the concept maps was measured with a Jaccard-Index for the used concepts. Results show that the concept maps created with the semi-automatic approach are on average about 37% more accurate with respect to the gold standard than the concept maps created with the fully automatic approach. In other words, the semi-automatic CMM approach can deliver the intended higher accuracy in comparison to a fully automatic approach, if capturing user intention in concept map creation is required.

We also validated that a semi-automatic CMM approach can be more efficient than a manual approach. Efficiency of the semi-automatic approach, in comparison to manually creating concept maps, could on average be improved considerably by a time saving of 27%. Achieving a high recall for the extracted concept and relation suggestions was another goal in this regard, to ensure that the "Automatic Text Processing" delivers many relevant results. Concept and relation precision were low as expected, because we did not want to show only clearly correct results, but many suggestions to choose from,

to the user. Concept recall was very good, with on average almost 90% of suggested concepts actually being used in the finished concept map by the user. Relation recall, however, was low with only 25% of extracted relations used in the final concept maps. All in all, it can therefore be said that the developed semi-automatic approach can provide a higher efficiency with its supporting features than a manual creation approach, although improvements need to be made for the suggested relations.

This can also be supported by the users' experience, who generally felt that using the semi-automatic prototype was more efficient and less tedious than to manually create the concept maps. These insights were gained from the answers to the questionnaire, along with additional scores for the most important user interface components. The most useful feature were the concept suggestions in the "suggested concepts sidebar", which users singled out as particularly helpful. The overall usability of the user interface was rated as very good by the users during the heuristic evaluation. They mostly had improvement suggestions for enhanced handling of the concept-map graph-elements in the "canvas" and some convenience features, regarding slightly different placement of user interface components or better keyboard navigation in the user interface components.

Summing this thesis up, it can be said that achieving the intended semi-automatic balance between automatic text processing and manual concept map construction was successful. It was important to find a good trade-off, so that the created concept maps with automatically extracted suggestions still reflect the mental model that the user had in mind, while removing many tedious parts of a manual process. However, there are still parts that were not implemented satisfactorily, like the suggested relations, or the scaling issues with longer text documents.

6.2 Future Work

We analyzed the advantage of a semi-automatic approach in comparison to a fully automatic approach, if we want to reflect the mental model of a learner in a concept map, where the underlying text is the knowledge to learn. What was not investigated, was the impact on meaningful learning if a learner uses such a semi-automatic approach. It could be that the concept and relation suggestions influence the memorization of the learned long-term knowledge, in comparison to the learner manually extracting all relevant concepts and relations out of a text document.

Although the generalization and topological arrangement of the concepts were left to the user in this semi-automatic CMM approach, a mechanism to pre-compute and incorporate the generalization levels would be an improvement. The nature of concept maps, as originally proposed by Novak and Cañas [NC06], included generalized concepts. Therefore, some sort of supporting feature, in order to direct the user to choosing the most generalized concepts, could be introduced. For instance, one possibility would be to prioritize concepts with a higher generalization level in the "suggested concepts sidebar", making it easier to choose the most relevant concepts of a topic. Another way would be

to let the user choose a generalization level and based on that choice let concepts of the currently selected level be suggested.

The suggested relations were less used by the users than expected. Improvements could be made by choosing a different strategy on how to extract the verb phrases. One way would be to try other regular expressions or a completely different method, like using dependency parsing or the proposed predicate-argument technique by Falke [Fal19].

There are several ways to address the scalability issues of the prototype implementation. At first, the "suggested concepts sidebar" needs to be docked at the top, right below the "toolbar". This would immediately prevent that this user interface element is not visible for longer texts. Another improvement would be to show only those concepts in the "suggested concepts sidebar" that are present in the currently visible text section in the "text area". Limiting the rendering of the concepts to only the ones relevant to the currently visible text section would certainly improve the user interfaces' responsiveness. Additionally, the number of concepts in the "suggested concepts sidebar" could be reduced by showing only one concept for a cluster. The other concepts of the cluster could then be shown on demand.

Another feature that was not finished in time for the prototype implementation, was the connection of manually created concepts and relations to the extracted suggestions. While a preliminary solution was implemented, it was decided that this feature would not be used for the prototype, because it still suffered from some severe bugs. Manually created concepts and relations would need to be connected in such a way, that if their strings are equal, they would have to be matched to concept or relation suggestions. The difficult part in this regard would be connecting them to the right coreference clusters.

The prototype implementation was also prepared to work with multiple text documents. Multiple texts can be added for "Automatic Text Processing" and the extracted concepts and relations can be used in the "canvas". However, connecting the extracted concepts and relations to those that are already loaded in the user interface is a more challenging problem, since besides the concepts and relations, the coreference clusters also have to be matched. A mechanism to connect these elements, which at the same time assures that the coreference clusters and assigned concepts and relations are semantically correct, would need to be developed.

If the prototype was intended to be used in another than a scientific environment, some usability issues would need to be resolved and convenience features added. Locations of user interface elements, like the "add relation" button, would need to be reconsidered. Further improvements would have to be made on the placements of concept suggestions in the "canvas", which are sometimes placed outside the "canvas". This could be done by choosing a different placement algorithm or a different library for the drawing of the "canvas" altogether. Enhanced keyboard navigation and renaming of concepts and relations, by typing if the respective element was selected, were also perceived as important missing functions by the users. Finally, a saving functionality for created concept maps would need to be implemented or even better, a connection to the CmapTools servers.

List of Figures

List of Tables

List of Algorithms

Bibliography

[AC16] Camila Z. Aguiar and Davidson Cury. Automatic construction of concept maps from texts. In *Proceedings of the Seventh International Conference on Concept Mapping*, 2016.

[Aus63] David P. Ausubel. *The Psychology of Meaningful Verbal Learning*. Grune & Stratton, 1963.

[BKL09] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly, 2009.

[BM06] Karsten Böhm and Lutz Maicher. Real-time generation of topic maps from speech streams. In *Charting the Topic Maps Research and Applications Landscape*, pages 112–124, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[CCF+03] John Coffey, Mary Carnot, Paul J. Feltovich, Robert Hoffman, Joan Feltovich, Alberto Cañas, and Joseph Novak. A summary of literature pertaining to the use of concept mapping techniques and technologies for education and performance support. *(No. Technical Report submitted to the US Navy Chief of Naval Education and Training). Pensacola, FL: Institute for Human and Machine Cognition*, 01 2003.

[CCH+05] Alberto J. Cañas, Roger Carff, Greg Hill, Marco M. Carvalho, Marco Arguedas, Thomas C. Eskridge, James Lott, and Rodrigo Carvajal. Concept maps: Integrating knowledge and information visualization. In *Knowledge and Information Visualization*, volume 3426 of *Lecture Notes in Computer Science*, pages 205–219. Springer, 2005.

[CCL18] Alberto J. Cañas, Roger Carff, and James Lott. eCmap: An embeddable web-based concept map editor. In *Proceedings of the Eighth International Conference on Concept Mapping*, 09 2018.

[CF12] April Colosimo and Megan Fitzgibbons. Teaching, designing, and organizing: Concept mapping for librarians. *Partnership: the Canadian Journal of Library and Information Practice and Research*, 7, 06 2012.

[CHC01] Marco Carvalho, Rattikorn Hewett, and Alberto Cañas. Enhancing web searches from concept map based knowledge models. In *Proceedings of the 5th World Multi-Conference on Systemics, Cybernetics and Informatics*, pages 69–73, 07 2001.

[CHC+04] Alberto J. Cañas, Greg Hill, Roger Carff, Niranjan Suri, James Lott, Tom Eskridge, Gloria Gómez, Mario Arroyo, and Rodrigo Carvajal. CmapTools: A knowledge modeling and sharing environment. *Concept Maps: Theory, Methodology, Technology. Proceedings of the First International Conference on Concept Mapping*, 1:125–133, 2004.

[CKWC08] Nian-Shing Chen, Kinshuk, Chun-Wang Wei, and Hong-Jhe Chen. Mining e-learning domain concept map from academic articles. *Computers & Education*, 50(3):1009 – 1021, 2008.

[CM16a] Kevin Clark and Christopher D. Manning. Deep reinforcement learning for mention-ranking coreference models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2256–2262, Austin, Texas, November 2016. Association for Computational Linguistics.

[CM16b] Kevin Clark and Christopher D. Manning. Improving Coreference Resolution by Learning Entity-Level Distributed Representations. *arXiv e-prints*, page arXiv:1606.01323, June 2016.

[CRE] Creately - The Visual Workspace For Team Collaboration. `https://creately.com/`. Accessed: 17.08.2020.

[CYTa] plotly/react-cytoscapejs: React component for Cytoscape.js network visualisations. `https://github.com/plotly/react-cytoscapejs`. Accessed: 17.08.2020.

[CYTb] Cytoscape.js: Graph theory (network) library for visualisation and analysis. `https://js.cytoscape.org/`. Accessed: 17.08.2020.

[DO12] Rebecca Dridan and Stephan Oepen. Tokenization: Returning to a long solved problem: A survey, contrastive experiment, recommendations, and toolkit. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 378–382, 07 2012.

[DOM] Mindomo - Collaborative mind maps, concept maps, outlines and Gantt charts. `https://www.mindomo.com/`. Accessed: 17.08.2020.

[Fal19] Tobias Falke. *Automatic Structured Text Summarization with Concept Maps*. PhD thesis, Technische Universität Darmstadt, 2019.

[FLA] Flask. `https://palletsprojects.com/p/flask/`. Accessed: 17.08.2020.

[GG08] Vita Graudina and Janis Grundspenkis. Concept map generation from owl ontologies. In *Proceedings of the Third International Conference on Concept Mapping*, 2008.

[GQ08] Jike Ge and Yuhui Qiu. Concept similarity matching based on semantic distance. In *SKG*, pages 380–383. IEEE Computer Society, 2008.

[GWLS17] Thomas Geymayer, Manuela Waldner, Alexander Lex, and Dieter Schmalstieg. How sensemaking tools influence display space usage. In *EuroVis Workshop on Visual Analytics*, June 2017.

[HC11] Hen-Hsen Huang and Hsin-Hsi Chen. Pause and stop labeling for chinese sentence boundary detection. In *International Conference Recent Advances in Natural Language Processing, RANLP*, pages 146–153, 01 2011.

[HCF01] Robert Hoffman, John Coffey, and Kenneth Ford. Storm-lk: A human-centered knowledge model for weather forecasting. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 45, 10 2001.

[HER] Heroku: Cloud Application Platform. `https://www.heroku.com/`. Accessed: 17.08.2020.

[HOJS99] H.E. Herl, H.F. O. Jr., and J. Schacter. Reliability and validity of a computer-based knowledge mapping system to measure content understanding. *Computers in Human Behavior*, 15:315–333, 1999.

[INF] InfoRapid KnowledgeBase Builder Web Edition. `https://inforapid.org/webapp/login.php`. Accessed: 17.08.2020.

[JM19] Daniel Jurafsky and James H. Martin. *Speech and Language Processing (3rd Edition Draft)*. Unpublished, Publisher of 2nd Edition: Prentice Hall, USA, 2019.

[KCB10] Juliana Kowata, Davidson Cury, and Maria Claudia Silva Boeres. A review of semi-automatic approaches to build concept maps. In *Proceedings of the Fourth International Conference on Concept Mapping*, 2010.

[KKE16] Peter Kraker, Christopher Kittel, and Asura Enkhbayar. Open knowledge maps: Creating a visual interface to the world's scientific knowledge based on natural language processing. *027.7 Zeitschrift für Bibliothekskultur / Journal for Library Culture*, 4(2):98–103, 2016.

[KR18] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, 01 2018.

[KS06] Tibor Kiss and Jan Strunk. Unsupervised multilingual sentence boundary detection. *Computational Linguistics*, 32(4):485–525, 2006.

[LHH19] Hobson Lane, Cole Howard, and Hannes Hapke. *Natural Language Processing in Action*. Manning Publications, 2019.

[Lin04] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS)*, Barcelona, Spain, July 25-26 2004.

[LL94] Shalom Lappin and Herbert J. Leass. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561, 1994.

[LLZ13] Yang Liu, Yanyan Li, and Zhiqiang Zhang. Designing an intelligent interactive tool for scaffolding concept map construction. In *ICHL*, volume 8038 of *Lecture Notes in Computer Science*, pages 280–289. Springer, 2013.

[LMR04] David Leake, Ana Maguitman, and Thomas Reichherzer. Understanding knowledge models: Modeling assessment of concept importance in concept maps. *Cognitive Science - COGSCI*, 01 2004.

[LPY15] Seulki Lee, Youkyoung Park, and Wan C. Yoon. Burst analysis for automatic concept map creation with a single document. *Expert Systems with Applications*, 42(22):8817 – 8829, 2015.

[MAT] Material-UI: A popular React UI framework. `https://material-ui.com/`. Accessed: 17.08.2020.

[MDP06] Paweł Matykiewicz, Włodzisław Duch, and John Pestian. Nonambiguous concept mapping in medical domain. In *Artificial Intelligence and Soft Computing – ICAISC 2006*, pages 941–950, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[MEI] MindMeister - Online-MindMapping-Software. `https://www.mindmeister.com/`. Accessed: 17.08.2020.

[MHK19] Golsa Mirbagheri, Mojdeh Hakimian, and Ahmad Agha Kardan. Using a Recommender System to Suggest Educational Resources and Drawing a Semi-Automated Concept Map to Enhance the Learning Progress. In *Proceedings of The International Conference on E-Learning in the Workplace 2019 (ICELW)*, 06 2019.

[Mil95] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995.

[MIN] MindMup - Free Online Mind Mapping. `https://www.mindmup.com/`. Accessed: 17.08.2020.

[MMS93] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: the penn treebank. *Comput. Linguist.*, 19(2):313–330, June 1993.

[MSB+14] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[MSS99] John R. McClure, Brian Sonak, and Hoi K. Suen. Concept map assessment of classroom learning: Reliability, validity, and logistical practicality. *Journal of Research in Science Teaching*, 36:475–492, 1999.

[NA06] John Nesbit and Olusola Adesope. Learning with concept and knowledge maps: A meta-analysis. *Review of Educational Research*, 76:413–448, 09 2006.

[NC06] Joseph D. Novak and Alberto J. Cañas. The theory underlying concept maps and how to construct and use them. Research report 2006-01 Rev 2008-01, Florida Institute for Human and Machine Cognition, 2006.

[NdMG+16] Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA).

[NEU] NeuralCoref 4.0: Coreference Resolution in spaCy with Neural Networks. `https://github.com/huggingface/neuralcoref`. Accessed: 17.08.2020.

[Nie92] Jakob Nielsen. Finding usability problems through heuristic evaluation. In *CHI '92: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 373–380. ACM, 1992.

[Nie94] Jakob Nielsen. Enhancing the explanatory power of usability heuristics. In *CHI '94: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 152–158, New York, NY, USA, 1994. ACM Press.

[NM91] Joseph D. Novak and Dismas Musonda. A twelve-year longitudinal study of science concept learning. *American Educational Research Journal*, 28(1):1117–153, 1991.

[OPC01] Ana Oliveira, Francisco Câmara Pereira, and Amílcar Cardoso. Automatic reading and learning from text. In *Proceedings of the International Symposium on Artificial Intelligence*, pages 302–310, 2001.

[PANa] Panama Papers - The Power Players / Visualizations of the Offshore Leaks database. `https://www.icij.org/investigations/panama-papers/the-power-players/`. Accessed: 17.08.2020.

[PANb] Panama Papers - Wikipedia. `https://en.wikipedia.org/wiki/Panama_Papers`. Accessed: 17.08.2020.

[PAR] Universal Dependencies - PART POS Tag. `https://universaldependencies.org/u/pos/PART.html`. Accessed: 17.08.2020.

[Pau17] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508, 2017.

[POC00] Francisco Pereira, Ana Oliveira, and Amílcar Cardoso. Extracting concept maps with clouds. In *Argentine Symposium of Artificial Intelligence*, 07 2000.

[Por80] Martin F. Porter. An algoritm for suffix stripping. *Program: electronic library and information systems*, 14(3):130–137, 03 1980.

[POS] ACLWEB - POS Tagging (State of the art). `https://aclweb.org/aclwiki/POS_Tagging_(State_of_the_art)`. Accessed: 17.08.2020.

[QJHL13] Iqbal Qasim, Jin-Woo Jeong, Jee-Uk Heu, and Dong-Ho Lee. Concept map construction from text documents using affinity propagation. *Journal of Information Science*, 39(6):719–736, 2013.

[RDOS12] Jonathon Read, Rebecca Dridan, Stephan Oepen, and Lars Solberg. Sentence boundary detection: A long solved problem? In *COLING*, pages 985–994, 12 2012.

[REA] React - A JavaScript library for building user interfaces. `https://reactjs.org/`. Accessed: 17.08.2020.

[SD18] Bhargav Srinivasa-Desikan. *Natural Language Processing and Computational Linguistics*. Packt Publishing, June 2018.

[SHRS09] Drahomíra Spoustová, Jan Hajič, Jan Raab, and Miroslav Spousta. Semi-supervised training for the averaged perceptron POS tagger. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 763–771, Athens, Greece, March 2009. Association for Computational Linguistics.

[SPA] spaCy - Industrial-Strength Natural Language Processing. https://spacy.io/. Accessed: 17.08.2020.

[STAa] Extract verb phrases using spaCy. https://stackoverflow.com/questions/47856247/extract-verb-phrases-using-spacy. Accessed: 17.08.2020.

[STAb] The State of JavaScript Framewrks 2019. https://2019.stateofjs.com/front-end-frameworks/. Accessed: 17.08.2020.

[TCRR10] Yuen-Hsien Tseng, Chun-Yen Chang, Shu-Nu Chang Rundgren, and Carl-Johan Rundgren. Mining concept maps from news stories for measuring civic scientific literacy in media. *Computers & Education*, 55(1):165–177, 2010.

[TEX] textacy: NLP, before and after spaCy. https://github.com/chartbeat-labs/textacy. Accessed: 17.08.2020.

[VC08] Jorge J. Villalón and Rafael A. Calvo. Concept map mining: A definition and a framework for its evaluation. In *Web Intelligence/IAT Workshops*, pages 357–360. IEEE Computer Society, 2008. 978-0-7695-3496-1.

[VC09] Jorge J. Villalón and Rafael A. Calvo. Concept extraction from student essays, towards concept map mining. In *2009 Ninth IEEE International Conference on Advanced Learning Technologies*, pages 221–225, 2009.

[VL06] Alejandro Valerio and David Leake. Jump-starting concept map construction with knowledge extracted from documents. In *Proceedings of the Second International Conference on Concept Mapping*, pages 296–303, 01 2006.

[VL12] Alejandro Valerio and David Leake. 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, 09 2012.

[WCLK08] W.M. Wang, C.F. Cheung, W.B. Lee, and S.K. Kwok. Mining knowledge from natural language texts using fuzzy associated concept mapping. *Information Processing & Management*, 44(5):1707 – 1719, 2008.

[YB18] Vikas Yadav and Steven Bethard. A survey on recent advances in named entity recognition from deep learning models. In *COLING*, pages 2145–2158. Association for Computational Linguistics, 2018.

[ZGH11] Amal Zouaq, Dragan Gasevic, and Marek Hatala. Ontologizing concept maps using graph theory. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, SAC '11, page 1687–1692, New York, NY, USA, 2011. Association for Computing Machinery.

[ZKM12] Krunoslav Zubrinic, Damir Kalpic, and Mario Milicevic. The automatic creation of concept maps from documents written using morphologically rich languages. *Expert Syst. Appl.*, 39(16):12709–12718, 2012.

[ZN08] Amal Zouaq and Roger Nkambou. Building domain ontologies from text for educational purposes. *IEEE Transactions on Learning Technologies*, 1(1):49–62, 2008.

[ZOS15] Krunoslav Zubrinic, Ines Obradovic, and Tomo Sjekavica. Implementation of method for generating concept map from unstructured text in the Croatian language. In *2015 23rd International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 220–223. IEEE, 2015.

Appendix

User Study Texts

Text 1: The Panama Papers revealed that the Minister of Finance of Andorra, Jordi Cinca, while he was CEO of Orfund SA, maintained an offshore company called Mariette Holdings Inc, until its dissolution in 2002 for fear of discovery of his participation in these businesses. The business activities of Orfund had ties to the blood diamond trade, and the refining and sale of African gold. This company closed shortly before the civil war in Ivory Coast.

The opposition demanded his resignation as a result. In response Cinca said that "if their connection would affect the Government of Andorra, will step down". Still, he did not.

Text 2: In 2018, investigative media Bivol.bg accessed the Panama Papers under an agreement with the International Consortium of Investigative Journalists. Later, they published a story about the offshore company Viafot which is attempting to acquire a key asset of Bulgaria's defense industry: the arms producer Dunarit. The Panama Papers show that Viafot is owned by Alexander Angelov who is the lawyer of media mogul Delyan Peevski. Other Bulgarian media had reported how all state institutions help Viafot acquire Dunarit through illegitimate means. However, no inquiry was opened by Bulgaria's General Prosecutor Sotir Tsatsarov.

Initially, in 2016, access to the Panama Papers was granted only to journalist Alekseniya Dimitrova. However, the spokesperson of the Ministry of Foreign Affairs (Bulgaria) raised concern that Dimitrova worked for the communist secret services, so her work with the papers may be biased.

Text 3: On April 6, 2016, Italy's Procura of Turin ordered the Guardia di Finanza to investigate the 800 Italians contained in the Panama Papers' documents.

Former long-time Prime Minister Silvio Berlusconi, who had been already convicted for tax evasion and expelled from the Parliament, was included in the papers. Other notable people whose names are mentioned in the Papers include entrepreneurs Luca Cordero di Montezemolo, Flavio Briatore, Adriano Galliani, and actor Carlo Verdone.

An investigation by ICIJ partner The Namibian found that the imprisoned mafioso Vito Roberto Palazzolo shielded his finances from Italian, Namibian and South African

authorities with shell companies in the British Virgin Islands set up by a German banker in Hong Kong, Wolf-Peter Berthold, which they also used to transfer control of Palazzolo's assets to his son.

Text 4: Austria's financial market authority has announced that they will audit two Austrian banks that were mentioned in the Panama Papers: Raiffeisenbank International and Hypo Vorarlberg. It will be specifically examining whether the banks have complied with their obligation to prevent money laundering. Hypo Vorarlberg subsequently announced that while they have complied with all laws in the past, they are planning to retreat completely from the offshore sector.

Text 5: Ian Cameron, the late father of UK Prime Minister David Cameron, ran an offshore fund called Blairmore Holdings Inc. through Mossack Fonseca, that avoided UK taxes for 30 years. His company moved to Ireland after David Cameron became Prime Minister. On April 6, Cameron admitted that he had owned shares in Blairmore, but said he sold his shares before becoming PM. Prominent politicians criticized the involvement of the Cameron family in the scandal. Leader of the Opposition Jeremy Corbyn urged an immediate independent investigation into the tax affairs of Cameron's family as well as tighter laws on UK tax avoidance.