Automatic Construction of Concept Maps from Unstructured Text

by Saima Gul

Submitted to the Graduate School of Engineering and
Natural Sciences
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Sabancı University

May, 2022

Automatic Construction of Concept Maps from Unstructured Text

APPROVED BY:

Prof. Yücel Saygın (Thesis Supervisor)	.....
Prof. Hüsnü Yenigün	.....
Assoc.Prof. Kemal Kılıç	.....
Assoc. Prof. Tevfik Aytekin	.....
Asst. Prof. Rahim Dehkharghani	.....
DATE OF APPROVAL:	.....

To my Late Father "Mirza Abdul Quddous"

with love

Acknowledgments

All the praises and thanks to Allah, the most merciful and the most beneficent.

I want to express my most profound appreciation to my thesis advisor, Professor Yucel Saygin, for his continuous support in various forms throughout this journey. His technical guidance and encouragement played the most crucial role in attaining this milestone of my life. Also, I convey my sincere thanks to the committee members and my mentors Professor Husnu Yenigun and Associate Professor Kemal Kilic. They have always been a source of inspiration for me during my Ph.D. Studies at Sabanci University. Their valuable suggestions and feedback not only improved my research work. Besides, the courses they taught me enhanced my overall abilities and skills. I am also grateful for the helpful comments and for the valuable time of the other jury members Assoc. Prof. Tevfik Aytekin and Asst. Prof. Rahim Dehkharghani.

Moreover, I am sincerely grateful to my friend and co-author of one of my papers Stefan Rabiger for his technical guidance and valuable advice regarding evaluations of my methods. He helped me learn many useful lessons, which are extremely helpful for conducting quality research.

Furthermore, without the financial support of the Higher Education Commission of Pakistan, and Balochistan University of Information Technology, Engineering & Management Sciences (BUITEMS) Quetta, I could not even think about perusing my doctorate studies at Sabanci University. I am highly thankful for their financial grant and their faith in me regarding completing this goal. Many thanks go to all the people at Sabanci University for providing their assistance in different roles.

My loved ones have always been the source of energy and enthusiasm while facing the challenges of life. I am deeply indebted to the epitome of this energy, indeed my Parents, my late Father, Mirza Abdul Quddous, and my mother, Fatima Bibi. They are the true meanings of love and kindness for me. I firmly believe that without their unparalleled support, I could not have achieved anything in my life.

I am so thankful to my beautiful and kind sisters, Sanum Gul, Saba Gul, and Ruba Gul, for their unconditional support and help. Their assistance and love made my life easier and my objective achievable. I extend special thanks to my mother-in-law, Mumtaz Bibi, for being so loving and supportive and taking care of my family in my absence.

Finally, I am highly indebted to my family, my husband, Mohammad Yaqoob Khan, and my kids, Safa Khan and Ahmed Demir Khan suffered greatly during this long and tiring journey. Living with a spouse pursuing a doctorate degree is probably more tiring than doing the actual Ph.D. He helped me in any possible way he could. His patience, encouragement and unwavering assistance sustained me this far.

Thank you, everyone.

Automatic Construction of Concept Maps from Unstructured Text

Saima Gul

Computer Science and Engineering

Ph.D. Thesis, 2022

Thesis Supervisor: Prof. Yücel Saygın

Keywords: Concept extraction, Keyword extraction, Knowledge base, Unsupervised learning, DBpedia, Document Difficulty Prediction

Abstract

Acquiring knowledge about a topic is a complex process that requires an individual to gradually understand all concepts related to this topic. The learning experience for individuals can be enhanced by visualizing the key characteristics that serve as a guide for learning. Equipping learners with a concept map that orders concepts, represented as nodes, to be studied according to their prerequisite order, indicated by direct edges, helps them stay on track. However, existing algorithms for automatic prerequisite detection are too inaccurate, which reduces learners' trust in such maps as one assumes the prerequisite relation to be completely reliable. In this work we propose to replace prerequisite relations with less authoritative coverage relations, as they indicate only that one concept is broader and related to another one. Since most of the prerequisites of a given concept are less difficult than the concept itself, we argue that combining the coverage relation with concept's difficulty scores encodes similar semantics as the prerequisite relation. However, due to the coverage relation being less authoritative, potentially inaccurately detected coverage relations may be ignored by learners.Such relations are considered more like recommendations instead of facts. In turn, this change in perception about the reliability of the edges in the resulting concept map creates less frustration for learners. Further, two additional aspects, the unstructured nature and the abundance of the learning materials should also be considered for devising a salable method for concept map's construction.

With this in mind, this thesis aims to automatically construct from unstructured textual learning materials a concept map that encodes concept difficulty as node color and

connects concepts through coverage edges. To that end, we divide the problem into two subtasks: extracting concepts from unstructured text and constructing the concept map by estimating the difficulty of each extracted concept before inserting coverage edges among them. Specifically, we first develop an unsupervised method to extract concepts from unstructured textual learning materials and then compute a difficulty score for each of the identified concepts in the second subtask with a novel unsupervised method. We find that our concept extraction method is more accurate than existing state-of-the-art methods. To the best of our knowledge, we have proposed the first unsupervised method for finding the concept's difficulty. Our experiments demonstrate the feasibility of our proposed difficulty prediction method.It also provides evidence for our core assumption that prerequisites of a given concept tend to be easier than the concept itself, which renders our methodology viable. These findings imply that our proposed methodology yields concept maps for courses that help individuals navigate concepts more successfully in practice.

Ḳavram Haritalarının Yapılandırılmamış Metinlerden Otomatik Çıkarılması

Saima Gul

Bilgisayar Bilimi ve Mühendisliği

Doktora Tezi, 2022

Tez Danışmanı: Prof.Dr. Yücel Saygın

Anahtar Sözcükler: Kavram çıkarma, Anahtar kelime çıkarma, Bilgi tabanı, Denetimsiz öğrenme, DBpedia, Belge Zorluk Tahmini

Özet

Bir konu hakkında bilgi edinme, bireyin bu konuyla ilgili tüm kavramları aşamalı olarak anlamasını gerektiren karmaşık bir süreçtir. Bireyler için öğrenme deneyimi, öğrenme için bir rehber görevi gören temel özellikleri çizge şeklinde görselleştirerek geliştirilebilir. Öğrencileri, düğümler olarak temsil edilen kavramları, doğrudan kenarlarla gösterilen önkoşul sıralarına göre incelenecek şekilde sıralayan bir kavram haritası ile donatmak, doğru patikada kalmalarına yardımcı olur. Bununla birlikte, otomatik önkoşul tespiti için mevcut algoritmalar çok hatalı sonuçlar verebilmektedir, bu durum önkoşul ilişkisi tamamen güvenilir olduğu varsayıldığından öğrencilerin bu tür kavram haritalarına olan güvenini azaltır. Bu çalışmada, önkoşul yerine bir kavramın diğeriyle ilişkili olduğunu gösteren kapsama ilişkisini kullanmayı öneriyoruz. Belirli bir kavramın önkoşullarının çoğu, kavramın kendisinden daha az zor olduğu için, kapsama ilişkisini kavramın zorluk puanları ile birleştirmenin, önkoşula benzer anlamı kodladığını savunuyoruz. Ancak, önkoşul ilişkisinden farklı olarak kapsama ilişkisinin daha az kısıtlı olması nedeniyle, potansiyel olarak yanlış tespit edilen kapsama ilişkileri öğrenciler tarafından göz ardı edilebilir. Bu tür ilişkiler daha çok tavsiyeler olarak kabul değerlendirilebilir. Buna karşılık, ortaya çıkan kavram haritasındaki kenarların güvenilirliğine ilişkin algıdaki bu değişiklik, öğrenciler için daha az hayal kırıklığı yaratır. Ayrıca, kavram haritasının inşasına özel bir yöntem tasarlamak için iki ek husus, yapılandırılmamış doğası ve öğrenme materyallerinin bolluğu da dikkate alınmalıdır.

Bu bağlamda, bu tez ile, yapılandırılmamış metinsel öğrenme materyallerinden, kavram zorluğunu düğüm rengi olarak kodlayan ve kavramları kapsama kenarları aracılığıyla birbirine bağlayan bir kavram haritasını otomatik olarak oluşturmayı amaçlamaktayız. Bu amaçla, problemi iki alt işe ayırıyoruz: yapılandırılmamış metinden kavramların çıkarılması ve bunların arasına kapsama kenarları eklemeden önce çıkarılan her kavramın zorluğunu tahmin ederek kavram haritasının oluşturulması. Özetle kavramları çıkarmak için denetimsiz bir yöntem geliştirildi. Yapılandırılmamış metinsel öğrenme materyalleri kullanan ve yeni bir denetimsiz yöntemle ikinci alt işte tanımlanan kavramların her biri için bir zorluk puanı hesaplandı. Bu yolla kavram çıkarma yöntemimizin mevcut en son teknoloji yöntemlerden daha doğru olduğunu gösterdik. Bildiğimiz kadarıyla, kavramın zorluğunu bulmak için ilk denetimsiz yöntemi bu tezde önermiş olduk. Deneylerimiz, önerilen zorluk tahmin yöntemimizin uygulanabilirliğini göstermektedir. Ayrıca, belirli bir kavramın ön koşullarının kavramın kendisinden daha kolay olma eğiliminde olduğu gerçeği de metodolojimizi uygulanabilir kılan temel varsayımımız için kanıt sağlamaktadır. Bu bulgular, önerilen metodolojimizin, bireylerin pratikte kavramlar arasında daha başarılı bir şekilde yol almasını sağlayan kavram haritalarının oluşturulmasında faydalı olduğunu göstermektedir.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Knowledge acquisition about a specific topic is a complex process which requires an individual to gradually understand each concept related to this topic from learning materials. Hence, the effective representation of concepts with their interconnected relationships in the respective learning materials is of utmost importance for successful learning. This formal representation of knowledge is known as conceptualization, in which the set of concepts, entities, objects sharing the same objectives and relationships among them are defined [1]. Uncovering this initially hidden structure step by step from the learning material is the key challenge for individuals to master that topic. While the effectiveness of this learning process depends on an individual, it can be supported by means that emphasize the underlying conceptualization. Ontologies like DBpedia [2], which contains the same information as Wikipedia, but in a more structured way, are one such means, because they are an explicit specification of conceptualization [3] due to their ability to encode multiple types of relations among concepts, allowing a broad range of applications. Nevertheless, ontologies alone are insufficient due to the sheer amount of relations and concepts they contain. But in combination with concept maps [4], which represent concepts as nodes and edges indicate existing relationships, ontologies become promising resources for enhancing the learning experience. For one, because concept maps limit the amount of visualized data to avoid information overload [5] and to highlight specific relations among concepts. In particular, they may summarize key points from learning

materials or emphasize connections among seemingly unrelated concepts that challenge individuals' understanding about the reason why those connections exist. Ultimately, these visualizations help individuals realize their misconceptions, aid in deepening the understanding about connections among concepts and guide individuals toward a better learning experience. One popular type of relationship is the prerequisite relation which indicates the concepts that must be known in advance to understand a specific concept. The concepts to be taught in the course are shown as nodes and a directed edge points from a basic concept to an advanced one if the advanced one expects knowledge about that particular basic concept. On the one hand, this enables a smoother learning experience for individuals and independent learners as a prerequisite graph already encodes specific learning paths that individuals may follow to master a course. But at the same time this relation communicates to individuals that they must obey these paths, which can be problematic if any of those relations is incorrect - be it either a missing prerequisite relation between two concepts or an unnecessary one that is not required for understanding a concept. Concept maps with prerequisite relations are typically created either manually by experts or automatically by exploiting metadata from structured textual documents like author-assigned keywords from scientific articles [6], the order in which concepts are presented in books [7], because it is challenging to extract relevant information solely from unstructured textual documents. Therefore, these methods are only applicable to a tiny fraction of available learning materials. Despite using metadata, the performance of existing methods for prerequisite detection produce too many incorrect predictions [8, 9, 10]. We argue that the coverage relation, which shares some commonalities with the prerequisite relation, is more suitable to guide individuals through a course because an edge pointing from concept c_i to c_j only indicates that c_i is more abstract than c_j while also being related to c_j, but not that c_i must be studied before c_j. This way individuals would be more forgiving in case of incorrect coverage relations than in the case of incorrect prerequisite relations. Equipping individuals with additional information in the concept map increases the chance of them being able to discern if a relation represents solely a coverage relation, i.e., it is an artifact in the data, or potentially an actual prerequisite relation. One

helpful indicator to discern both types could be the difficulty of a concept if we assume that advanced concepts tend to be more difficult than their prerequisites. By encoding concept difficulty as node color in the concept map and connecting concepts according to the coverage relation, individuals can explore courses by traversing course concepts in the order of the coverage relation while disregarding potentially noisy coverage edges due to node colors indicating concept difficulty. In other words, in this thesis we explore the possibility of approximating prerequisite edges with a combination of coverage edges and node color denoting concept difficulty. Although the idea of estimating concept difficulty is intuitive, the task known as predicting conceptual text complexity [11] has been introduced only lately. Hence, this field is still unexplored with only a limited number of supervised methods and benchmark datasets being available.

With this in mind, the goal of this thesis is to construct automatically a concept map from unstructured learning materials depicting related concepts while encoding concept difficulty by node color. Assuming unstructured textual learning materials makes the methods to be developed flexible and exploit the abundance of available materials. This amount of data also requires our methods to scale well, thus they must adopt an unsupervised approach. To that end, we divide the task into two subtasks: 1) extracting concepts from learning materials and 2) constructing the resulting concept map by computing a concept difficulty score to be used for encoding concept difficulty and by determining coverage edges among the concepts. More specifically, for the first subtask only those concepts are extracted from learning materials that have a corresponding Wikipedia article as this ensures the concepts' presence in DBpedia, which is important for the second subtask, because we extract features that leverage hierarchical information from DBpedia for estimating concept difficulty.

For the first subtask, we propose COBEC (Context-Based Extraction of Concepts), a new unsupervised method for concept extraction. COBEC only assumes that all concepts are related to a common overarching topic, which is not restrictive, given that learning materials for a course are all related to that course. While different unsupervised and supervised approaches exist, none of them take the context, in which a concept is ex-

tracted, into account. For example, while the extracted concept ”Local Area Network” is irrelevant in the context of the topic ”Data Structures”, it is highly relevant in the context of ”Wireless Network Security”. Similarly, context may be exploited to determine in which sense a concept is used as concepts might be ambiguous. For example, if a learning material mentions the concept ”Process”, one has to take the context into account to disambiguate the sense in which ”Process” is used - is the learning material referring to it in the sense of ”Chemical Process”, ”Biological Process”, or in a different sense? If multiple terms related to biology occur as context in the learning material, it is more likely that ”Process” refers to ”Biological Process”. Therefore, one should first disambiguate a concept before determining its relevance with respect to the context. We demonstrate that idea allows COBEC to outperform existing state-of-the-art methods on different datasets. Hence, COBEC identifies the most important concepts with regard to the context.

CODIF (Concept Difficulty), our unsupervised approach addressing the second subtask takes these concepts as input to estimate their difficulty, while requiring each concept to have an unstructured textual description as well as an entry in Wikipedia/DBpedia. Features used in that computation cover three main aspects, namely the content of the concept description, the position of a concept in the hierarchical structure of DBpedia, and the amount of background information required to understand that concept. While methods exist for assessing the difficulty of questions or exams based on the responses of students in assessments, there is limited work on estimating the difficulty of entire documents, especially when only a concept description is available.

We demonstrate in our experiments that our unsupervised method yields promising results for estimating concept difficulty, while also showing the general feasibility of our proposed methodology for constructing concept maps by providing evidence that our assumption of a concept's prerequisites being not more difficult than the concept itself is reasonable. As part of our experimental evaluation on both subtasks, we created five new benchmark datasets in total - three for the first subtask and two for the second one. While the three new datasets can be used for testing methods that only require unstructured texts, the two new datasets for the second subtask are the first ones to cover the educational do-

main.

In summary, our main contributions are:

1. We propose an unsupervised method for extracting context-based concepts from unstructured text.

2. We present a novel mechanism to infer and update the context information from the input documents and by demonstrating that leveraging the inferred context information improves the quality of the extracted candidate concepts.

3. We propose a promising unsupervised method for predicting the difficulty of each concept based only on its textual description.

4. We show that a concept's prerequisites tend to be not more difficult than the concept itself.

5. We release three new benchmark datasets for concept extraction on unstructured textual learning materials and two for predicting the difficulty of concepts based on their textual descriptions.

Chapter 2 explains the related work, before Chapter 3 introduces common concepts that are needed for understanding the proposed methods thereafter. Moreover, it describes the two subtasks to be addressed in this thesis. Chapter 4 explains COBEC, our proposed method for concept extraction from learning materials. Afterwards the same chapter elucidates CODIF, our unsupervised method tackling the second subtask for assigning each extracted concept a difficulty score, which forms the basis for computing the coverage relation. In Chapter 5 the experiments and results for both subtasks are reported. Last but not least, the thesis is concluded in Chapter 6 with a summary and outlook into future research avenues.

Chapter 2

Related Work

In this chapter we describe existing methods separately for the first subtask of concept extraction in Section 2.1 and for the second subtask of predicting the difficulty of concepts in Section 2.2. Relevant for the second subtask is also the automatic construction of concept maps, which is analyzed in Section 2.3.

2.1 Concept Extraction

Keyphrase extraction is a popular task with a multitude of existing supervised and unsupervised methods. One common idea for unsupervised methods is to limit the type of part of speech that represent candidate concepts, i.e., promising phrases that could or could not turn out to be concepts. For example, [12] consider only noun phrases as candidate concepts and based on their properties, such as frequency or length, they are regarded as concepts or not. Likewise, [13] rely on extracting noun phrases - either nouns or compound nouns - from parse trees to identify concepts. Therefore, any concepts violating that assumption cannot be extracted, whereas our method makes no assumption about the grammatical structure of concepts. In general, candidate concepts could be represented by any type of part of speech. To accomplish this, [14] propose to score the candidates based on phraseness, i.e., how likely a candidate represents a concept, and informativeness of a candidate concept instead of relying on part of speech. Other works pursue a similar

avenue by ranking candidate concepts based on different unsupervised heuristics. One such popular heuristic is TF-IDF (term frequency-inverse document frequency) which was proposed by [15]. It assigns high scores to terms that occur frequently in a specific document, but rarely in other documents. In contrast to our method, TF-IDF does not take the context into account, but we use it as a baseline method in our experiments. Another idea to rank concepts in an unsupervised manner is to model the extracted concepts as nodes in a graph and to score their importance according to graph-based properties like the centrality of a concept in the graph, where more central concepts receive higher scores. Popular methods include TextRank [16], TopicRank [17], and MutltipartiteRank [18]. As opposed to these methods, we model the context information instead of the concepts as a graph and rank the concepts based on that context information, which is ignored in the aforementioned graph-based methods. We use them as baselines in our experiments. In [19] the authors extend TextRank by assigning non-uniform weights to links according to the co-occurrence of the involved terms, but conceptually the model remains the same, thus it suffers from the same drawbacks as TextRank and we do not include it as another baseline. YAKE!, proposed by [20], is a state of the art unsupervised method that extracts five statistical features to detect keyphrases. It performs well on different datasets about various topics which use different languages, although it does not take the context of keyphrases into account when ranking them. Since it outperforms other unsupervised methods, we include it as a competitive baseline. A different unsupervised approach is to cast the problem of detecting keyphrases as a market basket analysis problem to apply frequent pattern mining techniques, e.g., in [21]. The authors represent concepts as n-grams and assume that if an n-gram represents a concept, neither its prefix nor its postfix of length n-1 may represent a concept. Therefore a concept is either included or omitted. However, this heuristic could easily miss concepts, e.g., if ”Neural network” is a concept ”Neural network optimization” could not be a separate concept, whereas in our method both could represent separate concepts. [22] propose a two-phase method for extracting concepts from research paper titles by combining a probabilistic generative model and grammar rules tailored to short texts. While it is theoretically possible to extend this

method to texts of variable length, this avenue has not been explored yet. In contrast, our method naturally deals with textual documents of arbitrary length. The work closest to ours is described in [23], where the authors utilize Wikipedia to filter out concepts, which are represented by n-grams, that have no Wikipedia article. The remaining n-grams are ranked according to TF-IDF and other heuristics to regard only the top-ranked n-grams as extracted concepts. While our method follows a similar strategy, we adopt DBpedia [2] which is a knowledge base and provides a structured way to access information that is available on Wikipedia and other platforms like Yago [24]. In contrast to [23], our heuristic for ranking n-grams considers context information. Overall, previous unsupervised methods for keyphrase extraction utilize different heuristics to rank candidates. However, these heuristics do not take explicitly context information into account. In contrast, our method exploits the fact that we know in advance that the set of unstructured textual documents is related to the central topic. Therefore, we model context information explicitly to rank candidates based on that information.

In terms of supervised methods, many supervised approaches based on statistical features exist [25, 26, 27, 28, 29]. However, they require labeled, domain-specific datasets which are expensive and time-consuming to obtain. While deep learning methods suffer from the same drawbacks, they tend to provide more accurate results at the expense of more training data. [30] introduce a recurrent neural network architecture to extract keyphrases from tweets, which is unsuitable for our specific scenario because learning materials could be arbitrarily long depending on the resource (book versus lecture slides) creating challenges for this type of network like the well known vanishing gradient problem [31]. [32] propose an attention-based LSTM to extract keywords from text documents. This supervised model considers the current sentence as the context of a word to decide if it is part of a keyphrase or not. However, context information beyond the current sentence is not considered as opposed to our method. Nevertheless, KEA [29] is a popular baseline as it utilizes two domain-agnostic features, one of them being TF-IDF. Despite being a supervised method, we also utilize KEA as a baseline to assess COBEC.

2.2 Conceptual Text Complexity

The task of predicting the difficulty of an entire document has been first formally described in [11], where the authors argue that work up to this point has only considered lexical and syntactical features, while discarding the textual content itself. In line with that argument, the authors refer to this task as predicting conceptual text complexity. According to this definition of the task, most prior work addresses a different task, namely predicting different types of difficulty. For example, in [33] the authors determine the difficulty of a concept based on a single feature, which is the position of a concept in an ontology. They argue that the deeper a concept is in the hierarchical structure of an ontology, the more difficult it is. However, this may lead to ties where multiple concepts sit at the same level in the ontology. In [34], the researchers focus on determining the difficulty of quiz questions, but not on estimating the difficulty for entire textual documents. The particular features used were popularity, selectivity and coherence, which are defined on an ontology. All works up to this point only address the task of predicting conceptual text complexity partially. Nevertheless, we leverage the aforementioned features in our work as they are defined on ontologies. The most similar work to ours are [11] and [35]. [11] introduces the conceptual text complexity task and proposes a supervised model that leverages 13 features that were obtained by constructing a graph through linking detected entities from the texts according to the existing relationships among their DBpedia articles. A follow-up study by the same authors [35] adds also syntactical and lexical features to the existing 13 features, which results in an enhanced performance. In contrast to both of these works, our method works in an unsupervised fashion, thus we do not consider them as baseline methods.

Query difficulty prediction is another task that is related to our problem. A well-performing supervised method is explained in [36], where the authors demonstrate that predicting the difficulty of topics associated with a query improves the ability of their system to rate the difficulty of queries. The 32 features extracted for topic detection are of lexical and syntactical nature, but we do not utilize any of them.

Another factor that affects the difficulty of a document is the required background in-

formation to understand a concept. Identifying the background information is also closely related to the task of finding the prerequisite relation between two given concepts. While the former task focuses on identifying which knowledge is required for understanding, the latter task deals with determining if there is a prerequisite relationship between a given pair of concepts. Since prerequisite detection is a popular task, we focus specifically on works that utilize knowledge bases and web taxonomies. A supervised machine learning approach is proposed in [37] by using Wikipedia as a main knowledge base to extract features which are used by different classifiers to establish the prerequisite relationship among learning objects or course concepts. While some of our extracted features are similar to the ones from [37], we adopt an unsupervised approach which cannot be compared with a supervised one. [38] and [39] leverage student assessments and test data for identifying prerequisites. However, these approaches are not applicable if assessments are unavailable, which is the case in our problem setting. Another study [9] proposes to utilize the Wikipedia link structure to identify pairwise prerequisite relationships between concept pairs based on a single metric known as Reference Difference (RefD). This method is unsupervised and based on the idea that if a related set of concepts of concept c_i refers more frequently to concept c_j than the related set of concepts of c_j refers to c_i, c_j is more likely the prerequisite of c_i. Otherwise c_i is more likely a prerequisite of c_j. In our method we consider background information as an aspect that impacts the overall difficulty of a concept. Therefore, we include RefD amongst other features related to background information for determining concept difficulty.

2.3 Automatic Construction of Concept Maps

There are several methods for constructing concept maps from text documents. The method proposed in [6] assumes that the input documents, here research articles, are structured, as this method relies on the presence of keywords assigned by the authors of the articles for constructing the concept maps instead of extracting the keywords as well. While the the authors of [7] construct concept maps from books, their method exploits

structured metadata from the books such as the table of contents. In contrast to these works, our method is more flexible as it does not assume the presence of any metadata in the input documents. In other words, it is agnostic to the type of textual learning materials used as input documents.

Chapter 3

Preliminaries and Problem Definition

This chapter introduces the basic concepts and terminology that is used throughout the thesis. First, to avoid confusion we clarify the meaning of key terms to be used in the remainder in Section 3.1. Then, in Section 3.2, we provide the relevant details that we leverage from DBpedia in our work. Finally in Section 3.3, we define each subproblem associated with the automatic construction of concept maps.

3.1 Terminology

In theory the terms *concept*, *keyword*, and *keyphrase* have different meanings. While a *keyword* is highly specific as it is a single word that describes relevant parts of the content of a document, a *keyphrase* has the same function as a *keyword*, but is comprised of multiple keywords. A set of keywords and keyphrases describe a *concept*, which tends to refer to an abstract notion in the document (”love”), although it could also describe concrete things (”book”). Note that a concept might or might not be explicitly stated in a document. For example, the keyword ”Addition” and the keyphrase ”Matrix Multiplication” both describe specific aspects of the concept ”Matrix Operators”, which might or might not appear explicitly, in a document. Moreover, a document might contain multiple concepts, e.g., the aforementioned document could also contain the keywords ”Rows”, ”Columns”, and the keyphrase ”Dimension of a Matrix” describing the concept ”Matrix”.

The document itself may be about the *topic* ”Linear Algebra”, thus we use the term *topic* in this chapter to refer to a more abstract concept. As can be seen from the examples, keywords or keyphrases like ”Matrix Multiplication” or ”Addition” can also be concepts but not the other way around. However, in our particular scenario documents represent learning materials. Thus, the concepts that they explain to learners must be explicitly stated in the text. Therefore, the main difference between *keyword/keyphrase* and *concept* has disappeared and thus we use these three terms interchangeably in the remainder of this paper. Similarly concept extraction and keyphrase extraction are used interchangeably. Nevertheless we distinguish between a *concept* and a *topic*, as the latter is broader. Another term we use frequently, *n-gram* refers to a phrase comprising *n* words. For example, ”Dimension of a Matrix” is a 4-gram. Thus, when *tokenizing* text into n-grams, i.e., splitting the text on a whitespace into words, one moves a sliding window of size *n* across the text to obtain all text sequences comprising *n* adjacent words. For example, if the sentence ”I like learning.” is tokenized into 2-grams, the resulting bigrams would be ”I like” and ”like learning”.

3.2 DBpedia

We use DBpedia as our knowledge base as it is an up to date knowledge base for Wikipedia, where each Wikipedia article has a corresponding DBpedia entry. DBpedia is stored as a directed graph encoded according to the RDF model [40] where nodes correspond to DBpedia entries and different types of edges (=predicates) link these nodes to describe different kinds of connections between these nodes. These predicates are extracted from Wikipedia articles, thus DBpedia is a means to access information, that is available in Wikipedia, in a more structured way via SPARQL queries [40], which enable efficient graph matching to retrieve subgraphs. In this work we utilize the following DBpedia properties:

- dct:subject - describes the categories in which that concept is used
- skos:broader - retrieve the parent categories of a category

- dbo:abstract - extracts the first lines of a Wikipedia article
- rdfs:label - extracts the associated Wikipedia page title/label for a concept, it might be lexically similar to the concept or could be different.
- dbo:wikiPageWikiLink - Wikipedia link structure plays an important role for extracting a variety of information in different problem domains. These annotations are performed by the contributors of Wikipedia articles according to the Wikipedia's "Manual of Style", which requires to select the important concepts and assign link to the related articles. As mentioned in relevant work this information has been utilized by many researchers to find the prerequisite relationship among topics or concepts. Based on the similar hypothesis we used DBpedia's dbo:wikiPageWikiLink property to extract the reference links associated with a concept and to further investigate that which sort of information they could provide which could help us to predict the difficulty of a concept.

By combining the DBpedia properties dct:subject and skos:broader one may traverse the hierarchical structure of DBpedia ontology.

3.3 Problem Definition

The automatic construction of a concept map from unstructured text is primarily divided into two subtasks, first the extraction of concepts and second the computations of difficulty score of each concept and identification of relationships among them. Based on these two subproblems we define our problem statements as follows.

3.3.1 Extraction of Concepts from Unstructured Text

In our scenario we are given a set of unstructured text documents $D = \{d_1, ...d_n\}$ that describe a set of concepts $C = \{c_1, ...c_m\}$, $m \geq 1, n \geq 1$. The goal is to identify the most relevant concepts from C. Each concept $c_i \in C$ is assigned a relevance score R_i, where higher scores indicate higher relevance. We cast this problem as a ranking task where we

retrieve those l concepts $c_j \in C$ at the k top ranks with the highest l relevance scores R_j, $l \geq k$. Our proposed method, COBEC, assumes that the concepts $c_i \in C$ share a mutual underlying topic T, i.e., they describe certain aspects of T.

For example, in the educational domain, T could represent the course topic and d_i would correspond to the learning materials describing the different concepts c_i to be taught as part of the course curriculum. Therefore, each extracted candidate concept c_p from d_i is related to T and describes it to some extent, which is expressed by a weight w_p. We refer to the set of all these tuples (c_p, w_p) as context information.

3.3.2 Computation of Difficulty Scores and Identification of Relations

In our scenario we are given a set of concepts $C = \{c_1, \dots c_m\}, m \geq 1$. All concepts c_i have a corresponding Wikipedia or DBpedia entry. The goal is to automatically construct a concept map comprised of nodes and edges, where nodes correspond to concepts from C and a directed edge from concept c_i to c_j indicates that c_i is broader than c_j and that c_i is related to c_j. We simply refer to this relation as coverage relation. We assume that c_i explains the bigger picture and provides a context in which the more specific c_j is discussed. In that sense c_i might be a prerequisite of c_j, but it could also only be related to c_i. Coverage is measured according to the hierarchical structure of DBpedia. More specifically, directed edges are inserted from c_i to c_j if c_i is is a parent, grandparent or great-grandparent of c_j, as it is known that DBpedia categories quickly become too abstract [41]. Moreover, the color of a node c_i in the concept map encodes its difficulty score $ds(c_i)$ according to our proposed unsupervised method CODIF, which results in lower scores for simpler concepts and in higher scores for more difficult concepts. In general, difficulty scores can take any value in the range $0 \leq ds(c_i) \leq 1$.

Chapter 4

Methodology

In this chapter we will explain our proposed solutions for addressing the two subtasks presented in Sections 3.3.1 and 3.3.2, respectively. First, Section 4.1 presents the methodology for extracting concepts in an unsupervised manner from learning materials. With these identified concepts at hand, Section 4.2 explains how to compute their difficulty scores for constructing a concept map automatically.

4.1 COBEC

Thinking in terms of concepts in the educational domain offers benefits for learning and teaching. Thinking about teaching materials for a course in terms of concepts enables instructors to design their classes accordingly such that they emphasize the connections between concepts [42]. This, in turn, makes it easier for learners to understand and remember the essence of a course. Similarly, it empowers learners to be curious and explore connections to seemingly unrelated concepts by themselves. This example underscores the vital role that concepts play in learning. But it leads to the question: how to identify such concepts for learners? Usually unstructured documents - be they lecture slides, books, or subtitles of videos - are suggested or provided by instructors as learning materials. However, reading them all is prohibitively time-consuming for learners, especially with the rising demand of online education, where learners hold responsibility for un-

derstanding taught concepts sufficiently well. Thus, ideally one would provide learners with a summary containing the most relevant concepts that are discussed in learning materials so that they can judge individually whether it is beneficial to study them or not. Alternatively concepts could also be visualized as concept maps [4], where concepts are represented by nodes and edges denote existing relationships, to emphasize the connections between concepts, e.g., [6] or [7].

To date a multitude of methods have been proposed to extract relevant concepts or keyphrases. This indicates not only the importance of the task, which is also relevant in many domains other than education, but also its difficulty. While some methods are tailored to a specific type of learning materials, as they rely on additional metadata like author-assigned keywords or a table of contents, others require a labeled corpus for training. However, in this work we argue that all of these methods neglect one important aspect, which is the context in which a concept is extracted. For example, while the extracted concept ”Local Area Network” is irrelevant in the context of the topic ”Data Structures”, it is highly relevant in the context of ”Wireless Network Security”. Similarly, context may be exploited to determine in which sense a concept is used as concepts might be ambiguous. For example, if a learning material mentions the concept ”Process”, one has to take the context into account to disambiguate the sense in which ”Process” is used - is the learning material referring to it in the sense of ”Chemical Process”, ”Biological Process”, or in a different sense? If multiple terms related to biology occur as context in the learning material, it is more likely that ”Process” refers to ”Biological Process”. Therefore, one should first disambiguate a concept before determining its relevance with respect to the context. With that in mind, in this chapter we propose COBEC (Context-Based Extraction of Concepts), a novel unsupervised method for extracting concepts from a set of unstructured, i.e., only raw text is available, textual learning materials. It combines natural language processing (NLP) techniques with a knowledge base, DBpedia in our case. COBEC extracts the relevant concepts from a set of input documents in two phases: (1) by extracting candidate concepts from unstructured text documents, then disambiguating and ranking these concepts based on the inferred context information and (2)

by enriching this context information with the help of DBpedia. Through the utilization of DBpedia our method is also capable of discovering out-of-vocabulary (OOV) words as context information, i.e., phrases that do not occur in the set of input documents. Likewise it naturally groups semantically similar concepts together in the rankings, i.e., semantically similar keyphrases receive the same ranks. In our experiments we demonstrate that COBEC is competitive when compared with seven popular supervised and unsupervised methods on four datasets about different topics in education. On three of these datasets it outperforms its competitors. Last but not least we publish two new datasets about the topics "Data Mining" and "Operating Systems" that were both extracted from books. To the best of our knowledge, there are no suitable datasets available for this type of learning material, thus we hope to advance the field with these new benchmark datasets. In summary, our main contributions are:

1. We propose an unsupervised method for extracting context-based concepts from unstructured text

2. We present a novel mechanism to infer and update the context information from the input documents.

3. We demonstrate that leveraging the inferred context information improves the quality of the extracted candidate concepts.

4. We make two new benchmark datasets for concept extraction available.

In Section 4.1.1 we introduce the methodology for extracting concepts in an unsupervised manner.

4.1.1 Methodology

First we present a short overview of DBpedia, a core component of COBEC, before we briefly outline COBEC in general and then delve into its details.

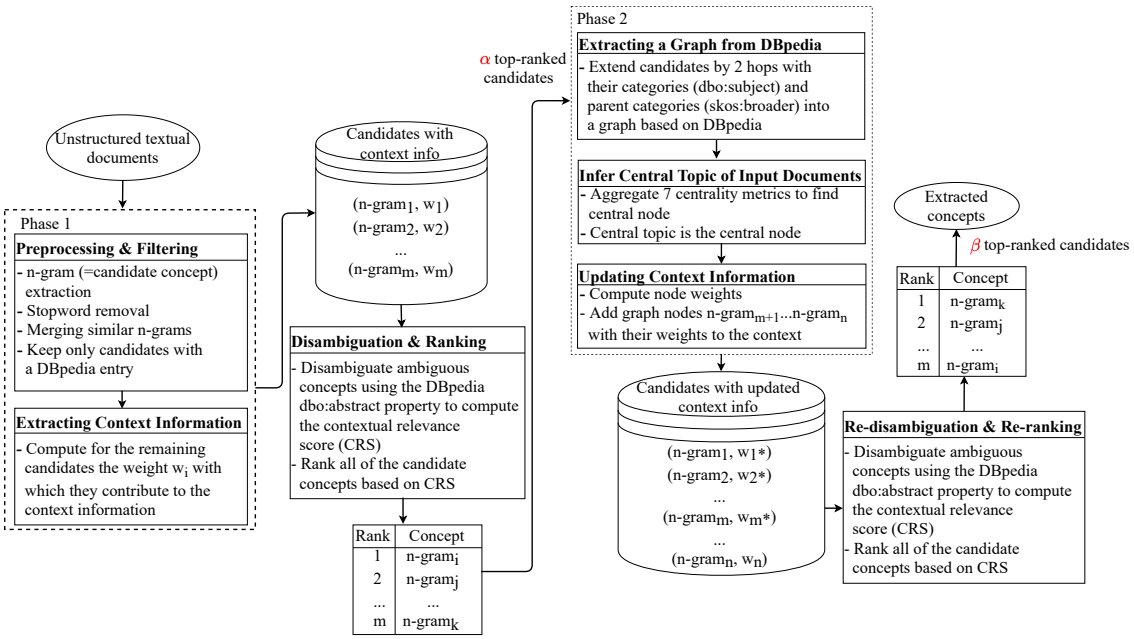

Figure 4.1: Workflow of COBEC to extract all concepts at the β top ranks from a set of unstructured textual input documents.

4.1.2 Overview

Fig. 4.1 depicts an overview of COBEC's workflow. Our method comprises two phases and represents concepts with n-grams. The aim of the first phase is to extract candidate concepts and rank them according to their relevance w.r.t. topic T based on the initial context information. As part of the first phase, unstructured textual input documents $d_i \in D$ are preprocessed using common NLP techniques to identify n-grams which correspond to candidate concepts $c_i \in C$, i.e., n-grams that potentially represent concepts (see Section 4.1.3). This set of candidates is filtered through a knowledge base, DBpedia in our case, to retain only those candidates which have a DBpedia entry (see Section 4.1.3).

All remaining candidates at this stage are regarded as context information related to the underlying shared topic T, where T is assumed to exist as COBEC's performance benefits if this assumption holds, but there is no hard requirement for T's existence. Yet design decisions in Sections 4.1.5 and 4.1.7 were made with the idea in mind that T exists. If that is not the case, COBEC's performance could deteriorate.

The context information stores tuples of candidates and a weight reflecting their con-

tribution towards the context (see Section 4.1.4). Since some of these candidates have multiple DBpedia entries, they are disambiguated s.t. each candidate is mapped to exactly one DBpedia entry, before ranking them according to their relevance w.r.t. topic T represented by the context information (see Section 4.1.5). Then the top-ranked candidates serve as input for the second phase.

Since the quality of disambiguating and ranking candidate concepts depends mainly on the reliability of the available context information, the goal of the second phase is to enrich this data with the help of DBpedia and to re-disambiguate and re-rank the candidates afterwards. At this stage the context information is comprised solely of n-grams from D which tend to represent specific concepts. Therefore, our idea is to complement these with more abstract n-grams, which are potentially OOV words, from DBpedia that bias the context information towards topic T. For this phase to succeed, only the α top-ranked candidates are considered because these are most likely describing T. These top-ranked candidates serve as input for DBpedia to construct a DBpedia subgraph that comprises a) the candidates, b) the candidates' categories according to DBpedia, and c) also the categories' parent categories as nodes (see Section 4.1.6).

Based on the assumption that the α top-ranked candidates describe T implicitly, the goal is to infer T from the subgraph. Since all candidates describe T, we assume that the most important node in the subgraph is closely related to T. The centrality of a node can be interpreted as a proxy for its importance. Thus, the term most similar to T is the central node in the subgraph (see Section 4.1.7). Similarly, other abstract words in the subgraph close to the central node are likely to describe T sufficiently accurately. Thus, to steer the context towards this central node and therefore towards T, the nodes from the subgraph are added to the context information, s.t. the central node contributes the most (see Section 4.1.8). Then all candidates are re-disambiguated and re-ranked according to Section 4.1.5 and all n-grams at the $\beta = k$ top ranks correspond to the l most relevant concepts according to Section 3.3.1.

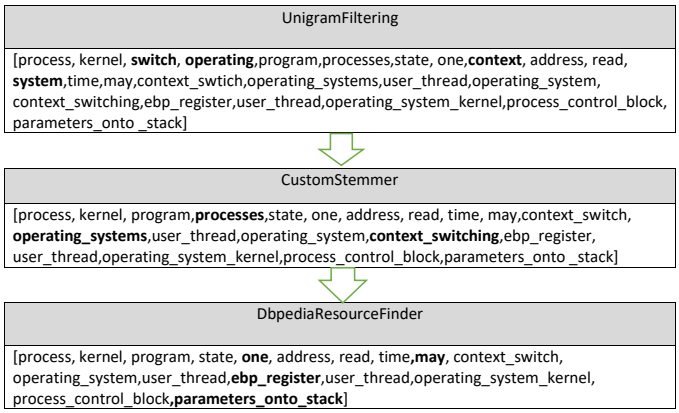

Figure 4.2: Filtering process for a set of candidate concepts sampled from a tokenized input document about ”Operating Systems”. Candidates in bold font are filtered out at their respective stage: first unigrams are removed if they are part of longer frequent n-grams, then semantically similar n-grams are merged after data cleaning and lastly n-grams with no DBpedia entry are discarded.

4.1.3 Preprocessing and Filtering Candidate Concepts

First the unstructured textual documents $d_i \in D$ are tokenized into n-grams representing the initial candidate concepts $c_i \in C$. From C, n-grams representing stopwords are removed using a stopword list comprising common terms. Counting the occurrences of the remaining candidates c_i in D yields a set of tuples $\tau = \{(c_i, f_i), \dots\}$ comprising a candidate c_i and its respective frequency f_i. Removing short, infrequent n-grams from τ (and therefore also from C) according to a frequency threshold f_t reduces noise. Our heuristic is based on the idea that a unigram ($n = 1$) is meaningful and thus should be retained if and only if it occurs at least f_t times more often than any larger n-gram which contains this unigram somewhere.

Formally, this translates to: given $(c_i, f_i) \in \tau$ and $(c_j, f_j) \in \tau$, where c_i is a unigram and c_j corresponds to an n-gram of higher order, i.e., $n \geq 2$, that contains c_i at some position, then c_i is retained if and only if $f_i > f_j + f_t$. If there are multiple n-grams c_j of higher order that contain c_i, f_j refers to most frequently occurring c_j.

The remaining n-grams from C are merged according to two rules, s.t. only a single n-

gram is present for semantically similar n-grams. First, a plural token is filtered out if it is also present in singular form. Second, the present participle of a regular verb is discarded, if a version without it exists, e.g., “context-switching” is removed if “context-switch” is already present. From the remaining candidates in C, we filter out those with no matching DBpedia entry. Note that no stemming is applied because no matching DBpedia entries exist for stemmed n-grams.

The whole filtering process is illustrated in Fig. 4.2 for a given tokenized input document about the topic ”Operating Systems”. With the help of the minimum frequency threshold f_t, the n-grams ”switch”, ”operating”, ”context” and ”system” are removed as they appear in longer n-grams. Then merging semantically similar n-grams leads to the removal of multiple n-grams, e.g., ”operating systems” is removed because its singular form already exists. Similarly, there is no DBpedia entry for ”may”, hence it is excluded from C. Likewise ”parameters onto stack” is filtered out due to no matching DBpedia entry. The latter example also illustrates that using DBpedia as a proxy for concepts is a reasonable choice since ”parameters onto stack” itself does not represent a concept, although it is relevant in ”Operating Systems” as part of other concepts.

4.1.4 Extracting Context Information

The remaining candidates after Section 4.1.3 form the initial context information C_{info}, i.e., we assume that they describe the underlying topic T. But the context information also includes the degree to which each candidate contributes to the context. Therefore, the context information comprises tuples (c_i, w_i), where c_i represents a candidate concept from C and weight w_i its contribution to the context information. w_i is calculated as:

$$w_i = |c_i| \tag{4.1}$$

where $|c_i|$ corresponds to the number of words in c_i.

At this point only n-grams with matching DBpedia entries remain in C_{info}, since we employed DBpedia as a filter to remove meaningless n-grams in Section 4.1.3. With that in mind, Eq. 4.1 assigns higher weights to the longer remaining n-grams as they carry

(a) (b) (c)

Figure 4.3: The context information for ”Process” is visualized as a word cloud, where a bigger font size of an n-gram implies that it has a higher contribution to the context. In a) the context information for ”Process” in ”Chemistry” is shown, in b) the context information for ”Process” in ”Biology” is depicted, and in c) the context information for ”Process” in ”Computing” is illustrated.

more contextual information and are more domain-specific than shorter ones, thus they describe the underlying topic T more accurately and should therefore have higher weights.

For example, the bigram ”operating system” is more descriptive than the unigrams ”operating” and ”system” individually.

Overall, the context information is a set of tuples containing the n-grams with their weights as described above. Since we assume that all n-grams are related to the underlying topic T. Therefore, the context information approximates T, which is shown in Fig 4.3. In Fig 4.3(a) the context information describes the underlying topic ”Chemical process”, whereas in Fig 4.3(b) it describes ”Biological Process”, and in Fig 4.3(c) it describes ”Process” in ”Computing” without necessarily mentioning these topics explicitly.

4.1.5 Ranking and Disambiguating Concepts

Since COBEC assumes that the context information describes the underlying topic T, that all candidate concepts have in common, we utilize the context information for disambiguating n-grams, i.e., to map each candidate to a single DBpedia entry. For ex-

ample, the n-gram ”Process” has 43 matching DBpedia entries. If used in the context of ”Operating Systems”, the correctly disambiguated word sense in this case would be ”Process_(computing)”, whereas in the context of ”Law” the appropriate sense could be ”Legal_process”. COBEC disambiguates candidate concepts by computing the contextual relevance score (CRS) as follows:

$$CRS_i = \frac{\sum_{c_j \in C_{info} \cap DBP_i} w_j}{|DBP_i|} \tag{4.2}$$

where w_j is the weight assigned to candidate concept c_j in C_{info}, and DBP_i corresponds to c_i's abstract in DBpedia which is processed and tokenized according to Section 4.1.3, except that remaining n-grams in DBP_i do not need a matching DBpedia entry, i.e., stopwords and infrequent n-grams are removed, and semantically similar n-grams are merged, which leaves only $\{1, 2, 3\}$-grams. The abstract is accessed through the dbo:abstract property of c_i's DBpedia entry.

In other words, we sum up the weights of all those n-grams from the abstract that match an n-gram in the context information. Since the length of the abstracts varies for different entries, the score is normalized by the length of the abstract. Therefore, CRS_i of a candidate concept c_i will be high if its DBpedia abstract contains many n-grams that have high weights in the context information. If there are multiple DBpedia entries matching a candidate c_i, CRS_i is computed for each of the entries and the one with the maximum CRS_{i*} is used as the word sense of c_i. In theory there could be multiple CRS_{i*}, which is resolved by selecting one of the tied CRS_{i*} randomly, although this has never happened in any of our experiments as it would indicate that two DBpedia entries describe the same candidate. Once all candidates are mapped to exactly one DBpedia entry, they are ranked w.r.t. the context information according to CRS_i. Thus, Eq. 4.2 is used both for disambiguating and ranking a candidate concept and it also corresponds to R_i as described in Section 3.3.1.

Note that for ranking CRS_i may introduce ties if two different candidate concepts either share the same tuples in the context information or tuples with the same weight. Thus, these two candidates most likely describe either the same concept or they are at least

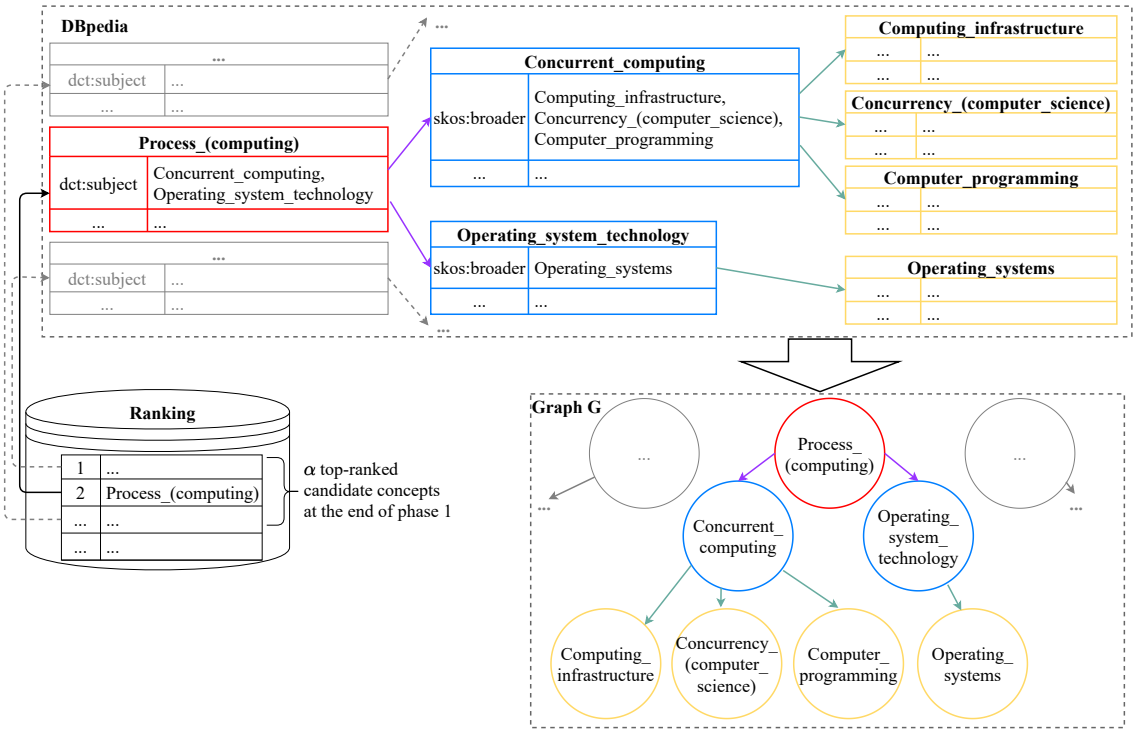

Figure 4.4: Constructing the directed graph G with DBpedia for the candidate concept "Process_(computing)" by traversing its corresponding DBpedia graph leveraging the properties "dct:subject" and "skos:broader".

semantically similar. For this reason COBEC does not split ties when ranking candidates.

4.1.6 Extracting a Graph from DBpedia

For steering the context information towards topic T with the α top-ranked candidates from the first phase, more abstract n-grams from DBpedia are added to the context information. One suitable type of information provided by DBpedia is the dct:subject property for a candidate concept, which describes the categories in which that concept is used. These categories are hierarchically structured in DBpedia, meaning that one can also retrieve the parent categories of a category by utilizing the corresponding skos:broader property in DBpedia. Thus, we construct a directed graph G_i from DBpedia by extracting for each of the α top-ranked candidate concepts its categories (dct:subject) and their respective parent categories (skos:broader). In other words, for each candidate we traverse

the

DBpedia graph for exactly two hops to construct its directed graph G_i. Directed edges in G_i are inserted from a candidate to its categories and from these to their parent categories. Note that creating such a graph G_i for each candidate yields a set G^* of α disconnected graphs, namely $G^* = \{G_i, \ldots, G_\alpha\}$. However, since those α top-ranked candidates tend to share common DBpedia nodes, which is either a category (dct:subject), a parent category (skos:broader), or both, some of the directed graphs $G_i \in G^*$ are connected with each other forming a larger connected graph. In practice, most $G_i \in G^*$ will be connected due to the α top-ranked candidates being similar, which increases their probability of sharing common DBpedia nodes, but this is not enforced by our graph construction method. At the end of creating G^*, we select the largest connected component and refer to this directed graph as G in the remainder. Since we assume that all α top-ranked candidates refer to T, it is most likely that those in the largest connected component describe T. We note that G could contain cycles as part of the construction process. The reason for not traversing the DBpedia graph beyond two hops is that the DBpedia categories quickly become too abstract. This decision is motivated by the findings in [41], in which the researchers found that relevant concepts lie close to each other, i.e., few hops apart, in the DBpedia graph.

We illustrate the construction process of G in Fig. 4.4 for the candidate concept "Process", which was disambiguated in the first phase according to Eq. 4.2 as "Process_(computing)" by DBpedia since the input documents are about the topic "Operating Systems". "Computing", which corresponds to the category information in DBpedia, is added by DBpedia to the name to uniquely identify the word sense of "Process". Then we extend "Process_(computing)" by two hops to construct G.

From its DBpedia entry COBEC retrieves the dct:subject property to access all categories in which "Process_(computing)" occurs. After retrieving their DBpedia entries - "Concurrent_computing" and "Operating_system_technology" - we extract their parent categories from their entries according to the information present in the skos:broader property. Therefore, the parent categories of the former one are "Computing_infrastructure",

”Concurrency_(computer_science)”, ”Computer_programming” and for the latter one there is only one, namely ”Operating_systems”. G now includes all these nodes - the concept, its categories and their parent categories with directed edges indicating how these nodes were traversed in DBpedia. This procedure is repeated for all α top-ranked concepts to construct G.

4.1.7 Inferring the Central Node in the Graph

COBEC leverages the central node in G to assess the relevance of each node. The central node corresponds to the central topic which represents the underlying topic T most accurately. To determine the central topic, we combine multiple centrality metrics as centrality can be defined in multiple ways and each metric covers a different aspect of it. To compute centrality metrics in G, we focus on its largest connected component and convert it into an undirected graph since edge orientation provides no additional information in our setting. We refer to the undirected version of G as $\tilde{G}$, where all directed edges are replaced with undirected ones. As centrality metrics we adopt degree centrality, Katz Centrality, Eigenvector Centrality, PageRank Centrality, Betweenness Centrality, Closeness Centrality and Information Centrality which are all described in [43]. These seven metrics comprise our set of centrality metrics CM to estimate the node centrality in $\tilde{G}$. The centrality score CS_c of a node c in $\tilde{G}$ is computed as:

$$CS_c = \sum_{i \in CM} \frac{CM_i(c)}{rank_{ci}} \tag{4.3}$$

where $CM_i(c)$ denotes the centrality value computed by centrality metric i for node c and $rank_{ci}$ corresponds to the rank assigned to c by centrality metric i. This way CS_c will be high if a centrality metric assigns a high centrality value to c and ranks it among the most central nodes. The central topic then corresponds to the node with the highest CS_c.

4.1.8 Updating Context Information

To update the context information from Section 4.1.4, we add each node c_i from $\tilde{G}$, but first its weight w_i is determined. With the identification of the central node according to

Section 4.1.7, we weigh the contribution of each node in $\tilde{G}$ towards the context information as some of these nodes might introduce noise. Since we assume that the central node represents the underlying topic T most accurately, it contributes the most to the context information, while nodes that are farther, i.e., more hops, away from the central node have a smaller contribution. To that end, COBEC traverses $\tilde{G}$ in breadth-first order starting at the central node, to which we refer as $root$, to avoid cycles. Then each visited node c_i is added to the context information as a tuple (c_i, w_i) with contribution w_i, where w_i is computed as

$$w_i = N - hops(root) \tag{4.4}$$

$hops(root)$ measures the number of hops that node c_i is away from the node $root$ and $N =$ number of nodes in $\tilde{G}$. Thus, nodes that are farther away from $root$ receive lower, yet positive weights that contribute to the context information. For example, if there are three nodes $root$, c, d in $\tilde{G}$, then $w_{root} = 3$. For the other two nodes there are two cases. Either c and d are directly connected with $root$, then their weights are set to $w_c = w_d = 2$. Otherwise all three nodes form one of two paths: either (1) $root$ is connected to c which, in turn, is linked to d or (2) $root$ is connected to d, which shares an edge with c. In case (1) the weights are $w_c = 3 - 1 = 2$ and $w_d = 3 - 2 = 1$, while in case (2) the resulting weights are $w_d = 3 - 1 = 2$ and $w_c = 3 - 2 = 1$. The weights assigned by Eq. 4.4 to n-grams extracted from DBpedia tend to be higher than those assigned to n-grams in the first phase according to Eq. 4.1 because typically there are more nodes in $\tilde{G}$ than n-grams c_i are long, i.e., $N > |c_i|$. As a result, the n-grams extracted from DBpedia will steer the context information towards topic T which, in turn, will affect the re-disambiguation and re-ranking of candidate concepts at the end of the second phase. This is a deliberate choice because context information extracted from DBpedia is potentially more reliable than the n-grams from the first phase. Finally, most of the abstracts in DBpedia start the definition sentence of an article with the respective category of the concept. Thus, when capturing that concept's category with DBpedia, it should contribute substantially to the updated context information, which, in turn, will affect both the ranking and disambiguation process afterwards. In case c_i is already present in

the context information from the first phase, the tuple with the larger weight w_i will be preserved, which is usually the one re-computed by Eq. 4.4.

Note that it is possible that c_i represents out-of-vocabulary words, i.e., words that do not occur in any of the input documents D, which is the case if a node in $\tilde{G}$ does not appear in D. After updating the context information with all tuples (c_i, w_i) from $\tilde{G}$, $c_i \in \tilde{G}$, all candidate concepts C are re-disambiguated and then re-ranked according to Section 4.1.5. All those candidates at the β top ranks are the final output of COBEC, which corresponds to the l most relevant concepts according to Section 3.3.1.

4.2 CODIF

Individuals can easily get overwhelmed and confused by information from different learning materials covering the same course concepts. One solution to this problem is equipping them with tools that highlight the key ideas, that is the most relevant concepts and how they are related to each other, in order to guide individuals' learning experience. Concept maps [4] have this capability by visualizing the relevant information as a graph. More specifically, they represent concepts as nodes and relationships as edges. Hence, they are flexible and can depict different information depending on the purpose. One popular type of relationship is the prerequisite relation which indicates the concepts that must be known to understand a given concept. Therefore, concepts maps visualizing this type of relation are also known as prerequisite graphs. Prerequisite graphs are either constructed manually by experts or by exploiting metadata from structured textual documents like author-assigned keywords from scientific articles [6], the order in which concepts are presented in books [7], because it is challenging to extract relevant information solely from unstructured textual documents. This implies that these methods leverage only a tiny fraction of available learning materials. Another aspect to consider is the semantic authority and reliability that the prerequisite relation communicates to individuals - this relation is not a recommendation but a fact that individuals have to obey. While this interpretation gives individuals the confidence to master a course concept by concept, it does not

allow for errors, at least individuals would expect this relation to be correct without any exceptions. But in practice, predicting prerequisite relations turns out to be challenging and has resulted in too many incorrect predictions to date [7, 8, 9, 10, 44, 45, 46, 47, 48]. If individuals suffer from those inaccuracies while using a prerequisite graph as a guide for mastering a course, their trust in the graph will vanish quickly, e.g., if it contains unnecessary relations while missing some crucial prerequisite relationships.

Hence, our goal is to construct a concept map for guiding individuals which expresses similar semantics as the prerequisite relation, namely a partial order for studying the concepts, while being less authoritative. This way it is more transparent to individuals that the relation is only a recommendation, which they can disobey at times. Another requirement for our method is allowing not only structured textual learning materials, but also unstructured ones. This provides more data to extract more accurate relations: while a large subset of structured learning materials might adopt a similar approach to explain a course, unstructured materials could be more diverse, and combining multiple learning materials would therefore lead to more accurate relations among concepts. And the last requirement is the ability of our method to scale to larger datasets from a myriad of domains, which favors an unsupervised approach over a supervised one. With these requirements in mind, the goal in this chapter is to construct a concept map based on unstructured learning materials for a course in an automated fashion. Our concept map is centered around two aspects, namely replacing the prerequisite relation by a coverage relationship, and encoding concept difficulty as node color in the resulting map. With this color information, individuals are equipped to make better decisions when it comes to deciding the next concept to study. For example, given a subgraph of related concepts, it would be more intuitive to study the simpler concepts first. However, estimating the difficulty of concepts has been introduced only recently as a separate task known as predicting conceptual text complexity [11]. Hence, only a limited number of methods exist. Similarly, only few benchmark datasets exist and none of them is from the educational domain. Therefore, we propose a unsupervised method, called CODIF (Concept Difficulty), that assigns each concept a difficulty score from 0 (easy) to 1 (hard) based on the concept's textual descrip-

tion. To accomplish this goal, CODIF leverages features from three domains, namely the content of the concept description, the position of a concept in the hierarchical structure of DBpedia, and the amount of background information required to understand that concept. For CODIF to work, it assumes the existence of a DBpedia/Wikipedia entry per concept, otherwise some features cannot be extracted. Using COBEC (Context-Based Extraction of Concepts) [49] or similar unsupervised methods for concept extraction, we can extract concepts from unstructured textual concept descriptions that are used for constructing our concept maps. However, in this chapter we limit ourselves to the scenario where concepts have already been identified and the goal is to visualize relationships among those concepts while coloring the concepts according to their difficulty score. We select the coverage relation between pairs of concepts to decide if an edge exists. More precisely, a directed edge from concept c_i to concept c_j is inserted if and only if c_i is more abstract in the DBpedia hierarchical structure than c_j, while c_j must be within three hops, which is motivated by the findings in [41], where the researchers found that relevant concepts lie only few hops apart in the DBpedia graph. The direction of the edge implies that c_i is potentially simpler than c_j and the key motivation for this assumption is that a more abstract concept is more common, thus it occurs in more contexts which makes it more likely for individuals to be already familiar with the basic ideas from other contexts. Moreover, c_i sets the context for c_j which makes it easier to understand the more specific c_j. Therefore, c_i is either a prerequisite of c_j or c_i is at least related to c_j. Combining this information with the node color, i.e., concept difficulty, equips learners with the ability to better identify prerequisite relations, because we assume that prerequisites are not harder than the concept itself. Thus, any coverage relations violating this assumption represent less likely prerequisite relations and can potentially be ignored by learners. As part of the evaluation of CODIF, we also create two benchmark datasets for the educational domain, which are intended to spur research about predicting conceptual text complexity, which is an important task given the abundance of available text documents on the internet that overwhelm people. Hence, a better categorization in terms of their difficulty level or accessibility would be beneficial to all types of interested individuals. Last but not least,

we showed that the key assumption of our concept map visualization, namely that a concept's prerequisites tend to be not harder to learn than the concept itself, holds in practice, which implies that our idea of approximating prerequisite relations by a combination of coverage relations and node color indicating concept difficulty is feasible.

In summary, our main contributions are:

1. We propose a promising unsupervised method for predicting conceptual text complexity.

2. We demonstrate that our extracted set of features is sufficient to estimate conceptual text complexity.

3. We show that a concept's prerequisites tend to be not more difficult than the concept itself.

4. We make two new benchmark datasets for predicting conceptual text complexity available.

The remainder is structured as follows. First, in Section 4.2.1 we clarify what we mean by the term "difficulty". Discussing the methodology for constructing a concept map, that encodes concept difficulty as the node color, follows in Section 4.2.2.

4.2.1 Notion of Difficulty

Before explaining the details about how to estimate concept difficulty, we discuss existing definitions of difficulty and then clarify what we understand by this term in the remainder of this chapter.

The notion of concept difficulty is subjective by nature, because it is affected by multiple factors. The first factor is language [50]. Considering only language, one may define a difficult concept as "a difficult concept is one that is based on a primitive concept". Similarly, one may say that a concept is simple if it is represented by a single lexeme, which is the most basic unit in a language to convey meaning, and otherwise it is complex. A large set of linguistic features has been proposed over the years to capture the

linguistic aspects of concept and they have been grouped into morphological features, syntactical features, and semantic features [51]. Closely related to language is also the reading difficulty, which is affected by multiple aspects which are categorized into syntactic depth, relational semantics, and prospective ambiguity [52]. A different approach for defining difficulty has been proposed in [53], where simple concepts are defined as those that preserve their meaning across different domains. This factor is known as specificity. The content of the document describing a concept also contributes to difficulty: from an individual's perspective more technical computations or more required mathematical understanding make a concept more difficult as these concepts are less common and do not have many applications across different domains [54]. Similarly, if the description of a concept is long, it contains more information which makes it more difficult to understand [55]. Background knowledge is also an aspect affecting concept difficulty: if individuals possess the expected background knowledge, they tend to find learning a new concept easier than others who lack some of that background knowledge [56]. Related to this idea is also the factor known as simultaneously learning, according to which simple concepts are those that can be learned sequentially instead of in parallel [57].

These different definitions and ideas clearly establish that concept difficulty cannot be objective as not only features related to the description of a concept contribute to its difficulty, but also individual factors like background knowledge which varies from individual to individual. To reduce subjectivity in the remainder of this work, we clarify what we mean when talking about concept difficulty. Most importantly, we assume that individuals share a similar set of visible and hidden factors, e.g. background knowledge, that affect how they perceive the difficulty of a concept. While this assumption permits a small number of individuals to disagree on the difficulty of concepts, combining the opinions of all individuals will converge toward the true difficulty of every concept regardless of individual disagreement. In other words, in our discussions hereafter we assume that only features of a concept's description differ, while differences in individuals, which could change the perception of the concept's actual difficulty, are ignored.

Table 4.1: Extracted features.

Feature Category	Feature Name
Content	Mathematical Complexity
Specificity	Granularity
	Related Knowledge
	Ambiguity
Background Knowledge	DBpedia Hierarchy
	RefD

4.2.2 Methodology

We use both DBpedia and Wikipedia as resources to extract features to predict the difficulty of concepts. In Section 4.2.3 we explain how the features are extracted and computed. We aggregate those features to compute a difficulty score in Section 4.2.4, which can be used as a basis for automatically constructing a concept map by encoding nodes with a color according to their difficulty score as explained in Section 4.2.5.

4.2.3 Feature Extraction

In this section, we propose several Wikipedia and DBpedia-based features to compute a difficulty score for a concept. The features are grouped into three categories based on what they measure. Table 4.1 provides an overview of all the features that we extract. Content-related features are extracted from the Wikipedia article of a concept, whereas the specificity of a concept is estimated by its position in the hierarchical structure of DBpedia. Last but not least, features related to the required background knowledge to understand a concept are derived from the concept's DBpedia page and Wikipedia article, respectively.

Content-related Features

Arguably the most common indicator of a concept's difficulty is based on how the concept is described in text. While prior works largely on lexical features as an indicator of difficulty other aspects affecting the difficulty of a concept are listed in Section 4.2.1. In fact, language might not be the best indicator. At least in our preliminary experiments the computation of readability scores like the Dale-Chall readability formula [58] or the common Flesch-Kincaid grade level formula [59] did not correlate well with the true concept difficulty. While more sophisticated lexical approaches could be more useful, an analysis of different approaches is beyond the scope of this thesis. All extracted features from this category are extracted from Wikipedia. For content analysis specifically, one aspect to consider is how mathematical an explanation is, as captured by our feature *Mathematical Complexity*. While it might be more useful for concepts related to STEM (Science, Technology, Engineering, and Mathematics) topics, it might fail for certain concepts related to Social Sciences that involve no equations.

Mathematical Complexity $MC(c_i)$ counts the occurrences of different mathematical symbols, notations, formulas, and equations in the description of concept c_i. We assume that encountering many mathematical expressions in a concept description makes understanding more challenging. In particular, the following exhaustive list of expressions serve as proxies for detecting mathematical expressions: "\mathbf", "\frac", "\lambda", "\max", "\star", "\operatorname", "\mapsto", "\quad", "\tfrac", "\varepsilon", "\langle", "\rangle", "\geq", "\leq", "\mathcal", "\sigma", "\prime", "\min", "\log", "\Omega", "\Theta", "\lim", "\infty", "\tfrac".

Specificity-related Features

Features in this category are centered around the idea that learning a more general concept is simpler than a more specific one. One reason for this assumption is that broader concepts tend to be more common and thus they might be already known from other contexts in which individuals familiarized themselves with the basic ideas. In contrast, more specific concepts are more relevant for experts and they thus require more knowledge. All

Assume that a hash function selects each array position with equal probability. If m is the number of bits in the array, the probability that a certain bit is not set to 1 by a certain hash function during the insertion of an element is

$$1 - \frac{1}{m}.$$

If k is the number of hash functions and each has no significant correlation between each other, then the probability that the bit is not set to 1 by any of the hash functions is

$$\left(1 - \frac{1}{m}\right)^k.$$

We can use the well-known identity for e^{-1}

$$\lim_{m \to \infty} \left(1 - \frac{1}{m}\right)^m = \frac{1}{e}$$

to conclude that, for large m,

$$\left(1 - \frac{1}{m}\right)^k = \left(\left(1 - \frac{1}{m}\right)^m\right)^{k/m} \approx e^{-k/m}.$$

Figure 4.5: A section of Wikipedia's article for the concept "Bloom Filters" shows the mathematical background information required to fully understand the detailed implementation of this concept.

specificity-related features are extracted from DBpedia.

For category information DBpedia provides a well-defined class structure for each DBpedia entry. By default if a concept has a DBpedia Uniform Resource Identifier (URI) it shows its existence as a resource page in the ontology. However, a subset of these resource pages are DBpedia categories, i.e. a concept acts also as a broader category. This may even happen at the lowest granularity level of the class structure - some concepts on the lowest level still represent valid categories in DBpedia and any concept that represents a category can be considered more general than others that only represent the concept itself. For instance, the concepts ”Merge sort”, ”Sorting algorithm”, and ”Algorithm” all are valid resource pages in DBpedia, but the latter two concepts are categories as well and therefore they are more general. The linguistic feature which is used by DBpedia to distinguish between a simple resource page and a category for the same label is the singular or plural form of the label. For example ”Algorithm” and ”Algorithms” are two different DBpedia pages with different values for the predicate rdf:type. Generally, for DBpedia categories the predicate rdf:type has the value of skos:concept.

Granularity $GRAN(c_i)$ of concept c_i is a Boolean feature that exploits the aforementioned information to indicate if c_i is a concept or a broader category, where we assume that broader categories tend to be easier to learn because they are more common and potentially occur in different contexts, which makes it more likely that individuals encountered them before. It is computed as follows:

$$GRAN(c_i) = \begin{cases} 0, & \text{if } c_i \text{ is a class and rdf:type == skos:concept} \\ 1, & \text{otherwise} \end{cases} \tag{4.5}$$

In other words, $GRAN(c_i)$ is only 1 if c_i represents a specific concept and 0 whenever it is a more general concept, i.e., a category.

Related knowledge $RK(c_i)$ of a concept c_i is another way to capture specificity. If a concept has more related concepts, we assume it to be more generic and therefore easier to fully grasp. This information is extracted by examining for each concept the link structure of Wikipedia, which is available in DBpedia. The essence of this idea is described by the

Table 4.2: Few examples of concepts from DSA, the relevant predicate value of rdf:type and the corresponding result for *Granularity* according to Equation 4.5.

Concept_label	rdf:type	$GRAN(c_i)$
Sorting_algorithm	owl:Thing, dbo:Software	1
Sorting_algorithms	skos:Concept	0
Quicksort	dbo:Software	1
Mergesort	dbo:Software	1

out-degree, which can be obtained by:

$$RK(c_i) = \sum_j w_{ij} \tag{4.6}$$

, where w_{ij} corresponds to the weight of the edge from c_i to c_j and $i \neq j$.

Ambiguity $AMB(c_i)$ also measures a certain type of specificity of concept c_i. If c_i is more common and therefore easier, it occurs likely in multiple categories. One way to capture this assumption is by checking if COBEC had to disambiguate c_i. In this case c_i is more likely a common and familiar concept. Therefore, $AMB(c_i)$ is a Boolean feature that describes if concept c_i was disambiguated by COBEC or not. It is computed according to:

$$AMB(c_i) = \begin{cases} 1, & \text{if } c_i \text{ wasn't disambiguated} \\ 0, & \text{otherwise} \end{cases} \tag{4.7}$$

Background Knowledge-related Features

Another aspect that contributes to the difficulty of a concept is the required background knowledge. Based on the notion that a lack of background knowledge makes a concept harder, whereas satisfying all required or expected background knowledge simplifies understanding a concept, we assume that concepts with more required background knowledge are harder to master. Features from this category are extracted from either Wikipedia or DBpedia.

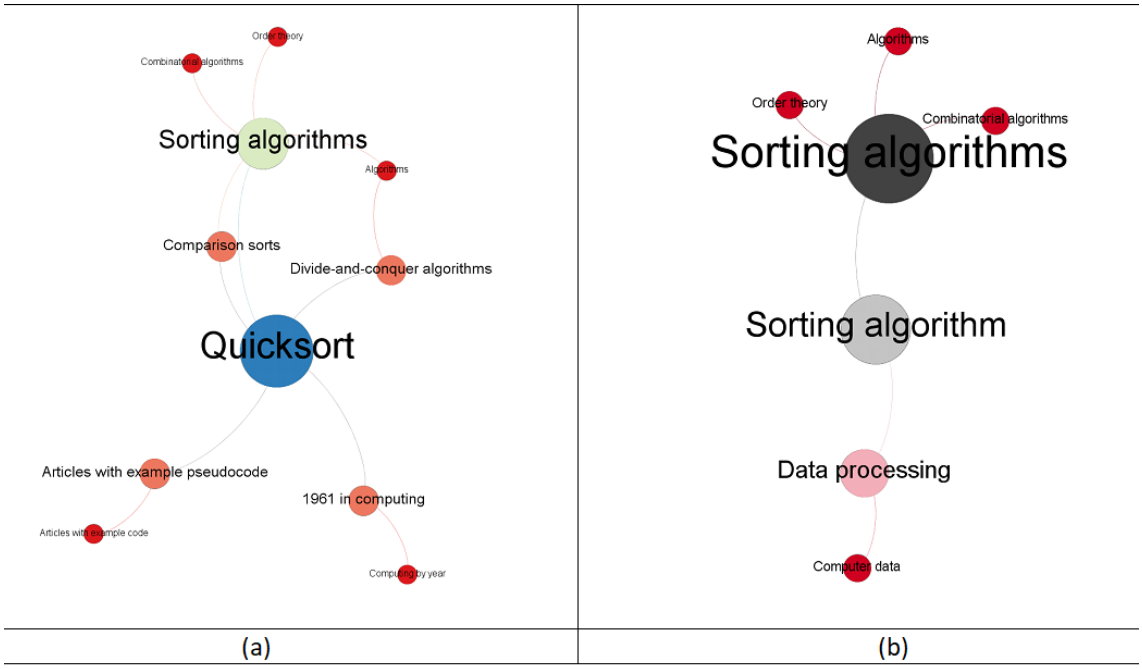

Figure 4.6: Example for extracting the feature DBpedia Hierarchy. Two subgraphs are extracted from the DBpedia hierarchy by expanding the parent category, grandparent category, and great-grandparent category of c_i=”Quicksort” in (a) and c_j=”Sorting algorithm” in (b). Thus, the graph is expanded by three hops according to the DBpedia hierarchy. The size of each node depicts its number of outgoing edges - the more outgoing edges a node has, the bigger the corresponding node is. It can be seen that one of the parents of ”Sorting algorithm”, namely ”Sorting algorithms”, is the grandparent of ”Quicksort”, which indicates that ”Quicksort” will be at a lower level than ”sorting algorithm” in the combined version of these two graphs. According to our assumption this suggests that ”Sorting algorithm” is part of the required background information for understanding ”Quicksort”.

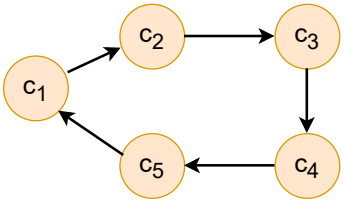

Figure 4.7: Example for the introduction of cycles when computing the feature DBpedia Hierarchy. The node labels correspond to the processing order of the nodes when computing the feature. For example, first $DBPH(c_1)$ was computed, then $DBPH(c_2)$, etc. The resulting cycle is a consequence of the processing order of the nodes and to deal with such situations consistently, we omitted all prerequisites involved in the cycle. This means for the given example that c_1 is not counted as a prerequisite for c_2, c_2 does not count as a prerequisite of c_3, c_3 does not count as a prerequisite of c_4, c_4 does not count as a prerequisite of c_5, and c_5 does not count as a prerequisite of c_1.

DBpedia Hierarchy $DBPH(c_i)$ counts the number of prerequisites for concept c_i according to the hierarchical structure of DBpedia. We assume that a more abstract concept, i.e., a concept that is at a higher level in the DBpedia hierarchy, is easier to learn than a more specific concept, which is at a lower level in the hierarchy. If the more abstract concept is also a parent (P), grandparent (GP) or a great-grandparent (GGP) of the more specific concept, the abstract concept is regarded as a prerequisite for the more specific concept. The $P/GP/GGP$ relations are defined in terms of the parent category of a concept, i.e., a given concept is expanded by three hops toward its parent categories, s.t. additional nodes from one level (=parents), two levels (=grandparents), and three levels (=great-grandparents) are included in a subgraph G, which contains those additional nodes from the three levels according to the DBpedia hierarchy and directed edges point from a more specific level to a more abstract level. The reason for not traversing the DBpedia graph beyond three hops for this feature is that the DBpedia categories quickly become too abstract. This decision is motivated by the findings in [41], in which the researchers found that relevant concepts lie only few hops apart in the DBpedia graph. $DBPH(c_i)$ for concept c_i is computed by comparing it with all other nodes $c_j \in C \setminus \{c_i\}$

using the method explained above. In other words, G_i is constructed by expanding c_i by three hops along the parent category and the same happens for constructing G_j by expanding c_j by three hops as well. Now c_j is counted as a prerequisite of c_i after merging G_i and G_j into G_k if and only if c_j is among the parents, grandparents, or great-grandparents of c_i in G_k. In other words, if and only if c_i is a child of c_k in G_k, c_j is counted as a prerequisite for c_i. To clarify this process, a sample computation of DBpedia Hierarchy is presented in Fig. 4.6 based on the graphs of the two concepts "Quicksort" and "Sorting algorithm". Note that the computation of $DBPH(c_i)$ will introduce cycles with high likelihood when building a graph that contains all $c_i \in C$ nodes and an edge from c_i to c_j if and only if c_i is more abstract than c_j according to DBpedia Hierarchy. Those cycles occur based on how c_i are processed and as a post-processing step, when computing $DBPH(c_i)$ for concept c_i, we remove all edges $c_i \rightarrow c_{i+1}$ from the resulting graph that were part of a cycle of length n, $cycle_n = (c_i \rightarrow c_{i+1}, \ldots c_{i-1+n} \rightarrow c_{i+n})$, which implies that c_i is ignored as a prerequisite for $c_{i+1}$when computing $DBPH(c_{i+1})$ assuming that the edge $c_i \rightarrow c_{i+1}$ is part of a cycle. This is illustrated with an example in Fig. 4.7.

Reference Distance (RefD) is an unsupervised measure to estimate the prerequisite relation between a pair of concepts c_i and c_j based on the Wikipedia link structure [9]. We refer to this as $RefD(c_i, c_j)$. This feature is widely available in various forms of text, such as in hyperlinks, notes, and book citations. RefD specifically assumes that the higher frequency of references is evidence for a pairwise prerequisite relation. In particular, given c_i and c_j, then c_j would be more likely considered as a prerequisite of c_i if most of the related concepts of c_i refer to c_j, while only few of the related concepts of c_j refer to c_i. $REF(c_i)$ counts how many prerequisites a concept c_i has among all concepts $c_j \in C \setminus c_i$ according to RefD. It is computed as:

$$REF(c_i) = \sum_{c_j \in C \setminus \{c_i\}} I(RefD(c_i, c_j)) \tag{4.8}$$

, where $I(c)$ is an indicator function that takes the corresponding RefD score between c_i

and c_j as input to return 1 if and only if c_j is a prerequisite of c_i according to RefD:

$$I(RefD(c_i, c_j)) = \begin{cases} 1, & \text{if } RefD(c_i, c_j) > 0 \\ 0, & \text{otherwise} \end{cases}$$

4.2.4 Computing a Difficulty Score for a Concept

We compute a difficulty score ds for concept c_i as follows:

$$ds(c_i) = MC(c_i) + GRAN(c_i) + RK(c_i) + AMB(c_i) + DBPH(c_i) + REF(c_i) \quad (4.9)$$

Since low values for all features from Section 4.2.3 are always assumed to indicate easier concepts, low difficulty scores refer to easier concepts, whereas higher scores identify harder concepts.

4.2.5 Constructing a Concept Map with Difficulty Scores

Given the difficulty scores of all concepts, i.e., $ds(c_i)$ for $c_i \in C$, based on Equation 4.9, they can be mapped to a continuous color map. For example, lighter colors may correspond to easier concepts, whereas darker colors indicate harder concepts. An edge from concept c_i to concept c_j in this concept map represents a coverage relation, in which c_i corresponds to a related and more abstract concept than c_j. This relation is determined by DBpedia Hierarchy when computing $DBPH(c_j)$ for all nodes c_i that are considered as prerequisites of c_j, an edge from c_i to c_j is added to the concept map. In other words, $DBPH(c_j)$ does not return the number of prerequisites as described in Section 4.2.3, but instead returns a list of prerequisites of c_j. For the sake of simplicity, we refer to such edges in the concept map as c_i covering c_j according to the coverage relation.

Chapter 5

Experiments and Results

In this chapter we evaluate our methods proposed for the two subtasks of concept extraction and concept map construction based on difficulty scores. Section 5.1 explains the detailed evaluation of COBEC while Section 5.2 focuses on CODIF.

5.1 Evaluation of COBEC

In our evaluation we focus on analyzing COBEC in terms of four research questions (RQ). First, we quantify the performance of our method in Section 5.1.4 (RQ1). In Section 5.1.5 we analyze separately the underlying main assumption of COBEC's second phase, namely if the identified central node according to Section 4.1.7 is similar to the underlying topic (RQ2). We then qualitatively investigate COBEC in terms of concepts with ties in Section 5.1.6 (RQ3). Last but not least, in Section 5.1.7 we investigate the robustness of our method when varying the parameter β, which determines how many of the candidate concepts at the top ranks to consider as concepts at the end of the second phase (RQ4). This is important for real-world applications of COBEC in order to set default values for the parameters in the absence of labeled datasets.

5.1.1 Datasets

In addition to evaluating our method on four common datasets used for evaluating keyphrase extraction algorithms, SemEval[60], Wiki20[61], Theses100[1], and Nguyen2007[62]; we also test it on three newly created datasets - DM, OS, and DB. The latter three datasets are tailored specifically to our scenario of extracting concepts from a set of unstructured textual learning materials, which are books in our case. All datasets are summarized in Table 5.1.

SemEval is the largest dataset in our experiments. It consists of 244 scientific papers extracted from the ACM digital library of four different research areas about computer science (CS). Each paper length varies from six to eight pages. The gold concepts are both author-assigned and reader-assigned. Among these gold labels 19% do not appear in the text, therefore the maximum achievable recall is 81%, thus with the maximum F1-score on this dataset would be 89% assuming 100% precision.

In this dataset each paper would have its own underlying topic, which would be the problem the respective paper addresses. Therefore, the underlying topics are not explicitly given in this dataset as they could be described with synonyms.

Nguyen2007 comprises 209 scientific papers to which student volunteers assigned keywords without seeing the author-assigned keywords to avoid introducing any bias. Similarly to SemEval, the 209 underlying topics are ambiguous and correspond to the problems addressed in the papers.

Theses100 contains 100 full master and PhD theses from the University of Waikato covering a diverse range of domains such as chemistry, psychology, and computer science. Each thesis represents an underlying topic, which is not explicitly given.

Wiki20 consists of 20 research article related to CS. Fifteen teams assigned keywords to each papers using Wikipedia article titles as the candidate vocabulary which implies that there are gold labels which do not occur in any of the papers. A unique feature of the dataset is that it provides the gold labels as well as the Wikipedia labels. For example, if the gold label is ”Process”, the corresponding Wikipedia label

[1] https://github.com/zelandiya/keyword-extraction-datasets

Table 5.1: Key characteristics of our datasets used for evaluation: dataset name (Name), number of input documents (#Docs), total number of words (#Words), number of gold concepts (#Cons), document length (DocLen), average number of gold concepts per input document (#Cons/Doc), topics covered by the dataset (Topic), number of gold concepts described by four or more words (#Cons > 3).

Name	#Docs	#Words/Doc	#Cons	DocLen	#Cons/Doc	Topic	#Cons > 3
SemEval	244	8332	4002	6-8 pages	16.70	Computer Science	373
Nguyen2007	209	5202	2369	6-8 pages	11.33	Computer Science	213
Theses100	100	4729	767	8-10 pages	7.67	Miscellaneous	49
Wiki20	20	6117	730	8 pages	36.50	Technical Reports	40
DM	20	15530	477	29 pages	23.85	Data Mining	4
DB	27	10994	395	26 pages	14.63	Databases	0
OS	1	4834	43	19 Pages	43	Operating Systems	0

will be ”Process_(computing)” because ”Process” is ambiguous and is disambiguated on Wikipedia. Since our approach is Wikipedia-based, we can leverage those Wikipedia labels for evaluation purposes as well: either COBEC ranks the gold label or the corresponding Wikipedia label among the most relevant candidate concepts - in either case they are counted as true positives since both labels refer to the same concept. However, since the baseline methods are independent of Wikipedia, we report for COBEC two results: one that utilizes only gold labels as true positives (**GOLD**) like all baseline methods and one utilizing Wikipedia and gold labels (**GOLD+WIKI**). An additional advantage of the latter set of labels is that it can be also used as an indicator for the effectiveness of our word sense disambiguation technique (Section 4.1.5): an improved performance of COBEC with GOLD+WIKI labels over our method with GOLD labels would suggest that our method manages to map some extracted ambiguous concepts to the correct Wikipedia articles.

Like in SemEval, each research article has its own underlying topic, namely the problem it addresses. Therefore, the underlying topics are not given explicitly.

DM is a dataset we created to test our method further on academic texts in the form of books. The dataset comprises all 20 chapters of a data mining book[63] where we treat each chapter as a separate input document. Moreover, we ignored any available metadata such as the table of contents. Instead we only used the raw text from the book pages. We recruited three experts to let them discuss which concepts are taught in the different chapters. All 477 extracted gold concepts in the dataset result from unanimous decisions among these experts. Each chapter has its own underlying topic, but in contrast to Wiki20 and SemEval, they are unambiguous and given explicitly as the topics correspond to the chapter titles.

DB is the second dataset that we created. It was extracted from a book about database management systems[64] and covers all 27 chapters, where each chapter represents a separate document. Adopting the same annotation strategy as described in DM leads to the extraction of 395 gold concepts. The chapter titles correspond to the unambiguous underlying topics like in DM.

OS is the third new dataset that we created and is a toy dataset, but it is a toy dataset. It is based on the "Process"[2] chapter of an online book about "Operating Systems" [3]. We applied the same annotation procedure as described in DM. This resulted in the extraction of 43 gold concepts from that chapter.

One of the authors has been teaching this course for a long time, thus we mainly include it for illustrative purposes. Although we also report the performance of COBEC on this dataset, we do not draw any conclusions from it due to its small size.

5.1.2 Baseline Methods and Evaluation Metrics

We employ popular state of the art methods from Section 2.1 as baselines in our experiments. Specifically, we choose TF-IDF[15], TextRank (TextR)[16], KEA[29], TopicRank (TopicR)[17], MutltipartiteRank (MultiPR)[18] and YAKE![20]. [65] implemented these methods [4] and also included MultiPR and YAKE! after their publications. All of these methods are unsupervised except KEA which is supervised. We apply them with their default values in our experiments. Moreover, for comparison we also include a commercially available tool, MonkeyLearn (MonkL) [5].

Since we cast the task of detecting concepts as a ranking problem, we follow the traditional approach of measuring the performance based on the concepts at the k top ranks. Since the baselines do not permit ties, this approach is equivalent to measuring the performance based on the k top-ranked concepts. However, as pointed out in Section 4.1.5, COBEC does not split ranks in case of ties because the tied candidate concepts are semantically similar and refer to the same gold label. This implies that COBEC potentially utilizes more than k candidates at its k top ranks. This setup favors our method over the baselines, as by chance multiple true positives could theoretically be tied at the same rank.

Therefore we introduce COBEC-T for a fair comparison, which splits ties in COBEC randomly. However, we always report results for COBEC and COBEC-T to illustrate the

[2] https://cnx.org/contents/epUq7msG@2.1:vLiqr17-@1/Process

[3] http://cnx.org/content/col10785/1.2/

[4] https://github.com/boudinfl/pke

[5] https://monkeylearn.com/

real capabilities of COBEC since concepts end on a tied rank if they are either closely related or even synonyms. Thus, a human would benefit from seeing a list with candidate concepts, among which some might have tied ranks, and in practice COBEC would be used instead of COBEC-T. Nevertheless, we only compare COBEC-T with baselines and only report results for COBEC. For measuring the performance of COBEC and the baseline methods in all our experiments, we evaluate each document of a dataset separately and then report macro-averaged results over all documents in line with previous works[20]. As the performance metric we adopt macro-averaged F1-scores at rank k. In addition, we also report macro-averaged precision at rank k and recall at rank k to give more insights about the capabilities of the methods. In one experiment (see Section 5.1.6), we also adopt R-precision which considers only the k top-ranked concepts in a document with k being set to the number of gold concepts in the respective document. Therefore, it indicates how many of the gold concepts were among the top-ranked candidate concepts of an input document. Due to the minimal differences between micro- and macro-averaged performance metrics across our experiments, we only report the latter ones.

Across all experiments we compute performance metrics based on exact matches, i.e., only detected concepts that match exactly a gold concept label are counted as true positives, while those detected ones that are semantically similar or partial matches are omitted.

5.1.3 Dataset Preprocessing and Experimental Settings

In terms of preprocessing the datasets we assume that concepts may be described as n-grams of length $n = \{1, 2, 3\}$, which implies that gold concepts of four words or more in our test datasets can neither be captured with our method nor the baselines. This decision is in line with previous works [20] which also discarded longer n-grams. In addition, as shown in Table 5.1, longer n-grams occur rarely in our datasets. The frequency threshold for unigrams is set to 15 and to 2 for bigrams and trigrams, respectively, meaning that extracted candidate concepts with fewer occurrences are discarded. In addition to excluding

common English stopwords, we also exclude typical words from academia as defined in the academic word list[66] [6] as all our datasets are from the educational domain.

Across all experiments we initialize our method as follows. In COBEC we set $\alpha = 5$, i.e., only the five top-ranked candidates from the first phase are passed into the second phase for constructing the subgraph from DBpedia. As stated earlier, choosing a small value for α is important to avoid propagating noisy candidate concepts into the second phase. Since the different algorithms used in this experiment return a different number of concepts, we consider only the concepts at the 15 top ranks per method. For our method this translates to setting $\beta = 15$. Considering the 15 top-ranked candidate concepts is chosen similarly to [20] where the 10 top-ranked ones were taken into account.

5.1.4 RQ1: Comparison with Baseline Methods

In this experiment we assess the viability of COBEC and COBEC-T and evaluate how well they fare against the seven supervised and unsupervised baselines introduced in Section 5.1.2. The settings described in Section 5.1.3 are used for the experiments.

The macro-averaged F1-scores of COBEC and COBEC-T with the seven baselines are depicted in Table 5.2. The results for precision are in Table 5.3, and the results for recall are in Table 5.4. We report results for COBEC in Table 5.2a and for COBEC-T in Table 5.2b separately for the first and second phase to measure the benefits of the second phase on the performance.

First phase of COBEC-T COBEC-T's first phase outperforms all baselines on three datasets in terms of F1-score, namely DM, Wiki20, and DB, while not being worse than at most 2% on the other datasets than the best method. Overall, this indicates that the method is promising.

[6] `http://www.essp-ny.org/images/stories/CCSS_academic_vocabulary_by_number.pdf`

Second phase of COBEC-T COBEC-T's second phase exhibits superior performance on five datasets in terms of F1-score, which are DM, Wiki20, OS, DB, and Nguyen2007, while not performing more than 1.5% worse on the other two datasets than the best methods. Similar observations hold true for recall and precision. Regardless of the performance metric, COBEC-T outperforms all baselines consistently on the datasets extracted from learning materials, which are OS, DM, and DB.

First phase vs. second phase of COBEC-T The second phase consistently exhibits better performance in terms of F1-score than using only COBEC-T's first phase, where the performance improvement ranges from 0.6%-7%. COBEC-T's performance also improves consistently in terms of recall and precision, with recall SemEval being the only exception in that recall and precision decrease in the second phase compared to the first one. For COBEC-T precision also decreases in the second phase on Nguyen2007. Overall these results indicate that enriching the context information with more abstract categories from DBpedia is a feasible strategy to steer the context towards the underlying topic. However, this hypothesis will be assessed in more depth in Section 5.1.5.

Word sense disambiguation in COBEC-T As a byproduct of our experiment, we computed for Wiki20 the performance of COBEC-T using two types of gold concepts as described in Section 5.1.1: GOLD and GOLD+WIKI. Including the WIKI labels lets us check if COBEC-T could disambiguate ambiguous candidate concepts properly. The improvement of 4.8% observed with the latter set of labels over the regular gold concepts in the first phase and the improvement by 4.4% in the second phase suggest that our word sense disambiguation method is able to map most of the ambiguous candidate concepts to correct DBpedia entries.

COBEC the same patterns observed for COBEC-T also hold for COBEC, but it consistently outperforms COBEC-T due to concepts ending up at the same tied rank.

Overall, COBEC-T, and similarly COBEC, perform best on the datasets extracted from books - DM, OS, and DB - regardless of the performance metric. There are two

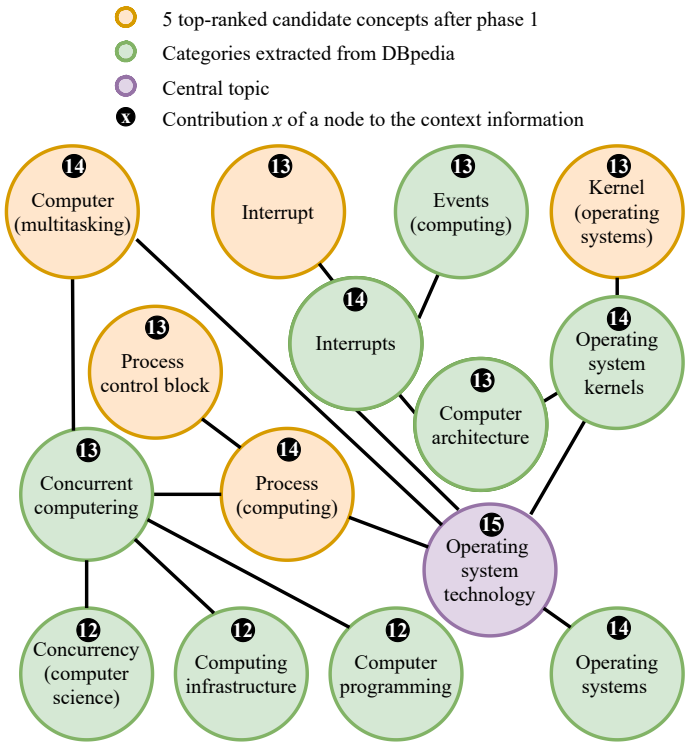

Figure 5.1: Resulting graph $\tilde{G}$ for dataset OS in the second phase of COBEC. All nodes are added to the context information and contribute with their weights according to Eq. 4.4 (with $N = 15$), depicted in black circles, to the context.

possible explanations for this result. On the one hand, SemEval, Wiki20, Nguyen2007, Theses100 deal with state of the art research articles, thus COBEC is less likely to find DBpedia entries for candidate concepts, whereas DM, OS, and DB are extracted from books which explain common and established concepts for which it is more likely to find corresponding DBpedia entries. On the other hand, we speculate that another possible explanation for this result could be that research articles are interdisciplinary, thus the candidate concepts revolve around multiple underlying topics which would violate the implicit assumptions made in COBEC and COBEC-T. But this hypothesis needs further exploration in the future.

To give a better intuition about the actual categories that are added to the context information in the second phase of COBEC-T, we illustrate in Fig. 5.1 the resulting graph $\tilde{G}$ (according to Section 4.1.6, Section 4.1.7, and Section 4.1.8) on OS. We use $\alpha = 5$

as in all other experiments. Thus, the 5 top-ranked concepts after the first phase, which are indicated by orange are added to the graph. Then they are expanded by DBpedia to retrieve and add their categories. Similarly, expanding these categories in DBpedia adds all their parent categories. These two expansion steps add all nodes in the graph that are not orange. After computing their centrality scores, "Operating system technology" is identified as the most central node, which corresponds to the central topic. Since $\tilde{G}$ contains 15 nodes, N in Eq. 4.4 is set to $N = 15$. Starting from this node the resulting weights of the nodes are assigned and represented by black circles in Fig. 5.1.

5.1.5 RQ2: Relation between the Central Node and Underlying Topic

Although we reported in Section 5.1.4 that the second phase of COBEC-T leads to performance gains, in this experiment we investigate if this improvement can be explained by the underlying topic being similar to the central node or if the observed improvements are more likely due to chance. We analyze this question with the following experimental design using the settings described in Section 5.1.3. Given a dataset, we run COBEC-T with both phases and as part of that process, we obtain the DBpedia graph according to Section 4.1.6 and the central node according to Section 4.1.7. Here we do not distinguish between COBEC-T and COBEC as the same arguments hold for both methods. However, we utilized COBEC-T to generate the results reported in Table 5.5.

Our main hypothesis in the second phase is that identifying the central node in the extracted DBpedia graph and assigning it the highest weight before adding it to the context information is responsible for the enhanced performance of COBEC. To investigate this hypothesis, we test if the central node is more related to the underlying topic than the other nodes in the extracted DBpedia graph. If this is the case, we found evidence that the central node, which has the biggest impact on re-ranking candidate concepts according to the context information at the end of the second phase, steers the re-ranking process in the right direction for more accurate concept extraction. This experiment assumes that the underlying topics are explicitly known, otherwise they could be ambiguous as explained in Section 5.1.1. Only OS, DM, and DB meet this criterion, hence the other datasets are

excluded from this experiment.

We measure the coverage between the central node and the underlying topic by the cosine similarity between their vector embeddings, which are known to preserve semantic relations [67]. More specifically, in our experiment we compute two similarities for each document in a dataset: 1) the similarity between the embedding of the central node and the embedding of the underlying topic (SINGLE-SIM), and 2) the average similarity between the underlying topic with all nodes in the extracted DBpedia graph according to Section 4.1.7 (AVG-SIM). However, in AVG-SIM the central node is excluded because it is already considered in SINGLE-SIM. If SINGLE-SIM is significantly larger than AVG-SIM, it indicates that the central node is more similar to the underlying topic and thus that it steers to re-ranking procedure towards a more accurate ranking which enhances COBEC's performance. To ensure that SINGLE-SIM is larger than AVG-SIM not due to chance, we measure the significance (significance level p=0.05) of the results by the non-parametric one-sided Wilcoxon signed-rank test as the distributions of the similarity scores are unknown and we only evaluate one direction, i.e., if SINGLE-SIM is larger than AVG-SIM. We report SINGLE-SIM and AVG-SIM macro-averaged over all documents in a dataset. In our experiments we use fastText embeddings [68, 69] which have the advantage to provide embeddings for out-of-vocabulary (OOV) words, i.e. words that did not occur in the training data, by representing words as subwords. Thus, if there is no embedding for a certain OOV word and also none for any of its subwords, fastText combines the embeddings of the unigrams that constitute that particular OOV word. For example, if "hello" is such an OOV word, its embedding is a combination of the embeddings of the unigrams "h", "e", "l", "l", and "o". We rely on pre-trained fastText embeddings that were trained on the English Wikipedia [7], where all embeddings are 300 dimensional vectors. We note that OS must be discarded from the datasets in this experiment as it contains only a single document, which renders performing a statistical significance test meaningless.

The results of this experiment are shown in Table 5.5. While SINGLE-SIM is significantly larger than AVG-SIM on DB, this is not the case for DM. Nevertheless, the

[7] https://fasttext.cc/docs/en/pretrained-vectors.html

macro-averaged SINGLE-SIM score is consistently larger than AVG-SIM score in both datasets. However, for DM it can not be ruled out by the significance test that we obtained these scores by chance. One possible explanation for the non-significant result is that in DM there are four chapters for which AVG-SIM is at least twice as large as SINGLE-SIM. Upon manual inspection it turns out that the central nodes are semantically different from the underlying topic, although they describe the contents of the chapters accurately. For example, in the chapter "Cluster analysis: advanced concepts", different methods using trees as data structures and hierarchical methods are described in detail. Accordingly, COBEC identifies "Trees" as the central node, while other nodes in the DBpedia graph, such as "Cluster analysis", are naturally more similar to the chapter title. Therefore, the similarity of the remaining DBpedia graph without the central node has a high chance to exhibit a higher similarity. This example highlights the shortcoming of our experiment and explains why the statistical significance might not be the best decision criterion in this experiment. In light of this discussion we interpret our results in Table 5.5 as a partial confirmation of our initial hypothesis that the central node is at least partially responsible for enhancing COBEC's performance in the second phase

5.1.6 RQ3: Qualitative Analysis of Ties in COBEC

Ties in COBEC favor its performance when considering the $k = 15$ top-ranked concepts in our experiment because unlike other methods, COBEC does not resolve ties as they indicate that different candidate concepts refer to the same concept as explained in Section 4.1.5.

To analyze if the concepts with tied ranks are semantically similar or even synonyms, we use the settings described in Section 5.1.3 for COBEC and utilize both of its phases. For the sake of brevity we examine only the 10 (instead of 15) top-ranked concepts for COBEC, TextRank, and MonkeyLearn on OS for a qualitative assessment of the ties in Table 5.6. Overall, COBEC includes 18 instead of 10 candidates with seven ties. Upon manual

inspection one can see that those ties all represent semantically similar concepts as op-

posed to TextR and MonkL. For example, ”PCB” is the abbreviation of ”Process_control_block” and ”Stack_pointer” is closely related to ”Call_stack”, whereas both baselines tend to rank more abstract candidates towards the top. At the end of the first phase seven ties occur in COBEC and after re-disambiguation and re-ranking at the end of the second phase the same number of ties exists among the 10 top-ranked candidate concepts. Normally the number of ties is stable across both phases. The only exception we observed is that the number of concepts involved in ties increases slightly, although the number of ties remains the same. This may happen if candidates representing the same gold label were incorrectly disambiguated in the first phase, but after updating the context information, they are now correctly disambiguated, thus previously incorrectly matched candidates merge with existing ties.

5.1.7 RQ4: Robustness of COBEC

In practice, α and β are the two parameters to tune in COBEC apart from setting the frequency thresholds of unigrams, bigrams, and trigrams for preprocessing the input documents. The range of value for α is limited because the top-ranked concepts after the first phase are assumed to be accurately describing the underlying topic T that all input documents are related to. Otherwise the context information is updated with unrelated noisy n-grams. Therefore, we do not vary α in this experiment, only β, which controls how many of the candidate concepts at the top ranks to consider as concepts. Note that we now use COBEC instead of COBEC-T because as stated before, we use COBEC in practice instead of COBEC-T and in this experiment we propose sensible default parameters for COBEC in the absence of gold labels for parameter tuning. In a realistic setting β will not be set to an absolute number which depends on a dataset, but it is rather set in relative terms w.r.t. the dataset size. This way the same β leads to a different number of detected concepts based on the dataset: if more candidate concepts exist, more actual concepts are extracted and vice versa. Therefore, we set $\beta = P_c$ allowing us to vary c. Here P_c denotes that only those candidate concepts are considered as concepts if they are at least in the c-th percentile of the re-ranked candidate concepts. Apart from setting

$\beta = P_c$, we use the settings described in Section 5.1.3. In addition to the different percentiles, we also report R-precision, for the same reason we use percentiles: the number of gold concepts could vary across documents and datasets. which considers only the k top-ranked concepts on a dataset with k being equal to the number of gold concepts in the respective dataset. Therefore, it indicates how many of the gold concepts were among the top-ranked candidate concepts.

The results of this experiment are depicted in Table 5.7. Despite varying the percentile over a range of values, the F1-scores remain at a similar level. The only notable exception is observed in DM with a decrease in F1-score by 5.4% from P_{75} to P_{25}, whereas the F1-scores vary at most by 2.7% otherwise. While there is room for fine-tuning c on a dataset, for real-world scenarios, without any available gold labels, our experiments suggest that setting $c = 50$ as the default value for β leads to stable results which are also close to the R-precision values across all datasets.

5.1.8 Discussion and Conclusion(COBEC)

In this work we proposed an unsupervised method to extract concepts from unstructured textual learning materials for a course. Assuming that all documents are related to a single central topic, namely the course topic itself, enables our approach to infer context information explicitly for disambiguating and ranking extracted candidate concepts unlike existing methods. Our method comprises two phases, where the first one focuses on identifying suitable candidate concepts that describe the central topic, while the second one aims at enriching the available context information with the help of the graph structure extracted from DBpedia to re-disambiguate and re-rank all extracted concepts more accurately. In the experiments conducted on seven datasets our novel method outperforms all seven popular baselines on five datasets and is competitive on the other two demonstrating its efficacy. Further experiments revealed its robustness when varying its parameters. Our three newly created datasets for the topics "Data Mining", "Operating Systems", and "Database Management Systems" along with the code are available[8].

[8] `https://github.com/gulsaima/COBEC`

One limitation of our proposed method is that it fails if there is no DBpedia entry for a candidate concept. This is more likely to occur for the most recent discoveries in science as it takes a while until they become accepted and popular enough to justify a DBpedia entry. To mitigate this problem, we plan to combine our method with fuzzy matches. More precisely, we will combine our method with the idea that a language model, such as BERT[70], already represents a knowledge base[71]. When trained on the latest research articles, such a language model would provide indicators as to whether a candidate concept, that does not exist in DBpedia, should be retained or discarded. Furthermore, in the future our goal is to reduce the processing time required for word sense disambiguation. While it may take several minutes in the current implementation to disambiguate a candidate concept, there is room for improvement. By discarding candidates based on their categories or parent categories using the DBpedia property ”rdf:type” one can potentially speed up the processing time. Once our novel method is mature enough, we will integrate it as a plugin into Moodle.

Another plan for the future is to investigate why COBEC performs particularly well on the datasets extracted from books compared to regular research articles. Our idea is to use other types of learning materials as input documents, especially lecture slides and subtitles of lectures, to see if COBEC outperforms other methods on such datasets as clearly as it did on books.

5.2 Evaluation of CODIF

To assess how feasible our proposed method is for constructing automatically a concept map with concepts as nodes, coverage edges interconnecting the nodes, and node color encoding concept difficulty, we conduct four experiments that assess different aspects of the methodology. Each experiment focuses on one specific research question (RQ):

1. RQ1 - How does CODIF perform and which feature categories are most important? (Section 5.2.2)

2. RQ2 - Can existing prerequisite datasets be labeled automatically? (Section 5.2.3)

3. RQ3- How similar is our coverage relationship to the prerequisite relationship? (Section 5.2.4)

4. RQ4 - How do the resulting concept maps look like? (Section 5.2.5)

In RQ1 we focus on quantifying the quality of the computed difficulty scores for concepts. RQ2 explores a strategy for creating more labeled datasets for the task of predicting conceptual text complexity from existing datasets for prerequisite detection, whereas RQ3 investigates the quality of the edges to be added in the concept map. Last but not least, RQ4 analyzes the overall quality of the resulting concept maps while also examining the hypothesis that prerequisites of a concept are easier than the concept itself, which is an essential assumption for our proposed methodology to automatically construct a concept map from unstructured textual learning materials.

5.2.1 Datasets

Since the task of predicting conceptual text difficulty was introduced only recently [11], not many datasets exist. To the best of our knowledge only two datasets exist, one of them using articles from the simple and normal Wikipedia [72]. However, as pointed out in [73], half of the sentences in simple Wikipedia are not simplifications of the original sentences. The other dataset, which is known as Newsela [73], contains 1130 newspaper articles and human editors created four simplified versions of each article, and each version is simpler than the previous one. Hence these simplified versions are suitable for children and second language learners with varying language proficiency. But none of the datasets covers the educational domain. Moreover, Newsela only provides labels w.r.t. the same article, which allows only comparing simplified versions of the same article with each other, but not across different articles. Therefore, we created suitable datasets based on Metacademy[9], where experts provide prerequisite relationships between concept pairs for a number of domains. We specifically focused on the domains "Data Structures & Algorithms" (DSA) and "Machine Learning" (ML), since are common undergraduate

[9] `https://metacademy.org/`

courses in computer science. DSA comprises 29 concepts in total, whereas ML contains 140 concepts. We asked two experts to label both datasets in terms of the concepts being either ”easy” or ”hard”. Both were instructed to rate the difficulty of a concept only based on its Wikipedia article. Experts resolved disagreeing labels by discussions. Their inter-annotator agreement according to Cohen's kappa was $\kappa = 0.86$ on ML and $\kappa = 0.81$ on DSA, which are both considered almost perfect agreement [74]. The resulting label distributions are shown in Table 5.8. Since the relative amount of hard concepts is higher in ML than in DSA, ML seems slightly more difficult, which seems intuitive, given that more background in mathematics is required to comprehend all ML concepts. However, both datasets are imbalanced as easy concepts are more prevalent than hard ones.

5.2.2 RQ1: Performance of CODIF and Feature Category Importance

To measure the performance of CODIF, we apply it to ML and DSA and compute for each concept $c_i \in C$ its difficulty score $ds(c_i)$ according to Equation 4.9. To quantify the performance of CODIF, we solve a binary classification task with the labels ”easy” and ”hard” that are available as ground truth in DSA and ML. Therefore, we discretize the continuous scores of CODIF by running k-means with $k = 2$ to cluster the resulting difficulty scores of all concepts. All concepts in the cluster with lower average difficulty score are assigned the label ”easy”. Similarly, all concepts in the cluster with higher average difficulty score receive the label ”hard”. This way we can measure CODIF's performance in terms of accuracy, F1-score, precision, and recall. Due to the datasets being imbalanced, reporting only accuracy is insufficient. Since CODIF is the first unsupervised method, we report the results of a majority label classifier as baseline method which assigns the most common label, which is ”easy” in case of both datasets according to Table 5.8, to any given concept.

Table 5.9 shows the results for CODIF and the baseline method, including the use of different combinations of feature categories for CODIF. What stands out is that the best subset of feature categories comprises only Content and Background Information, but not

Specificity. The resulting best performance for this subset is 0.87 in terms of F1-score on DSA and 0.8 on ML, whereas the performance drops in terms of F1-score, when using all feature categories, by 0.1 on DSA and by 0.2 on ML respectively, indicating that Specificity has problems to capture concept difficulty especially on ML. This is further supported by looking at performances of any other subset of feature categories involving Specificity - each time that subset exhibits a substantially worse performance on ML than on DSA. One reason for the better performance of CODIF using any subset involving Specificity on DSA could be that it is an artifact due to the smaller dataset size compared to ML. Another reason could be directly presented by the performance of the baseline method. The baseline method performs better on DSA than on ML as expected because the label distribution is more skewed toward easy concepts on DSA. The performance on DSA is competitive with CODIF, using only Content and Background Information, in terms of F1-scores. Therefore, an alternative explanation for Specificity worsening the performance of CODIF could be that Specificity tends to perform better for easy concepts which are more prevalent in DSA. Nevertheless, more datasets are needed to identify the exact cause. But CODIF, using only features from the Content and Background Information category, outperforms all other variants across all metrics. Only in terms of recall on ML the category Content is more accurate. But overall, the results indicate that Specificity should not be considered. In terms of the most important feature category, Background Information wins because it performs well consistently across all datasets and metrics with little variance unlike Content, which struggles with the distinction of easy and hard concepts on ML, which becomes clear when looking at its low precision of 0.63.

5.2.3 RQ2: Automatically Labeling Prerequisite Datasets

Due to the limited number of datasets, we also explore the idea of automatically labeling existing prerequisite datasets in terms of difficulty. To that end we hypothesize that *"the number of prerequisites of a concept indicates how challenging it is for an individual to understand that concept. Thus, the more prerequisites a concept has, the harder it is"* (H1). This is motivated by the idea that having to understand a large number of prereq-

uisites first, before applying this knowledge to actual concept to be learned, is a difficult task in its own right. In contrast, learning a concept without having any prerequisites appears to be easier.

The binary difficulty labels for the concepts of ML and DSA are automatically derived according to the following procedure. Given the set of prerequisite datasets $D = \{\text{DSA, ML}\}$, each $d \in D$ contains a set of concepts C. For each concept $c_i \in C$, its true number of prerequisites is counted in d. Let that count be cnt_i. After processing all concepts in d, all counts cnt_i are available. Applying k-means with $k = 2$ to these counts yields two clusters c_e and c_h. All concepts belonging to c_e, the cluster containing the lower counts on average, are assigned the "easy" label, while the concepts from cluster c_h, which contains the concepts with higher counts on average, are assigned the label "hard". Comparing these automatically assigned labels with those determined by the experts in Section 5.2.1 allows us to examine our hypothesis H1 from above based on the resulting confusion tables for ML (Table 5.10a) and DSA (Table 5.10b). In addition to reporting accuracy, F1-score, precision, and recall like in Section 5.2.2, we also measure the correlation coefficient to see if there is a linear relationship between the two variables, the automatically assigned labels and the ground truth labels from Section 5.2.1.

The resulting performances are reported in Table 5.11. While the number of prerequisites works surprisingly well on ML with an F1-score of 0.84, it yields less reliable predictions on DSA with only 0.68. In terms of correlation coefficient, there is medium correlation ($r = 0.5$) between the automatically assigned labels and the ground truth labels on ML, whereas the correlation is very weak on DSA. This can be explained by the fact that most concepts were considered easy by experts, although they have many prerequisites. This leads to the question if either DSA or ML is an artifact or whether there is simply no inherent relationship between the prerequisite detection task and predicting conceptual text complexity, which can only be answered with more datasets.

5.2.4 RQ3: Similarity between Coverage and Prerequisite Relationship

To test how accurately the coverage relation based on DBpedia Hierarchy (cf. Section 4.2.3) captures prerequisites, we compare it with prerequisite relations discovered by RefD. Both methods are unsupervised and RefD was devised for detecting prerequisites, whereas DBpedia Hierarchy estimates prerequisites based on the DBpedia hierarchy. Since one requirement for our method to construct concept maps is that it scales, only unsupervised methods are considered. For prerequisite detection RefD is the only available method. On each dataset we compute coverage and prerequisite relations for all pairwise combinations of concepts, resulting in the confusion matrices shown in Table 5.12. While RefD has problems to distinguish false positives from true negatives in Tables 5.12b and 5.12a, DBpedia Hierarchy struggles with separating true positives from false negatives as shown in Tables 5.13b and 5.13a. This leads to the performances reported in Table 5.14, where the coverage relation is more accurate with a higher recall, while RefD is more precise. Overall, both relations perform similarly poorly in terms of F1-scores on ML, while on DSA the coverage relation achieves a twice as high F1-score with 0.3 compared to RefD, but given the size of DSA this might be an artifact. Overall, the results indicate that the coverage relation performs at least as well as RefD in terms of detecting prerequisite relations, thus making it a viable choice for connecting nodes in the concept map.

5.2.5 RQ4: Concept Maps for ML and DSA

Here we perform a qualitative analysis of the generated concept maps in terms of how well they serve their purpose of guiding individuals correctly. To that end, we employed CODIF, using only the feature categories Background Knowledge and Content according to Section 5.2.2 to generate concept maps for DSA and ML, where nodes correspond to concepts and edges are inserted based on the coverage relation. Note that the continuous difficulty scores instead of the discrete difficulty labels are used for encoding concept

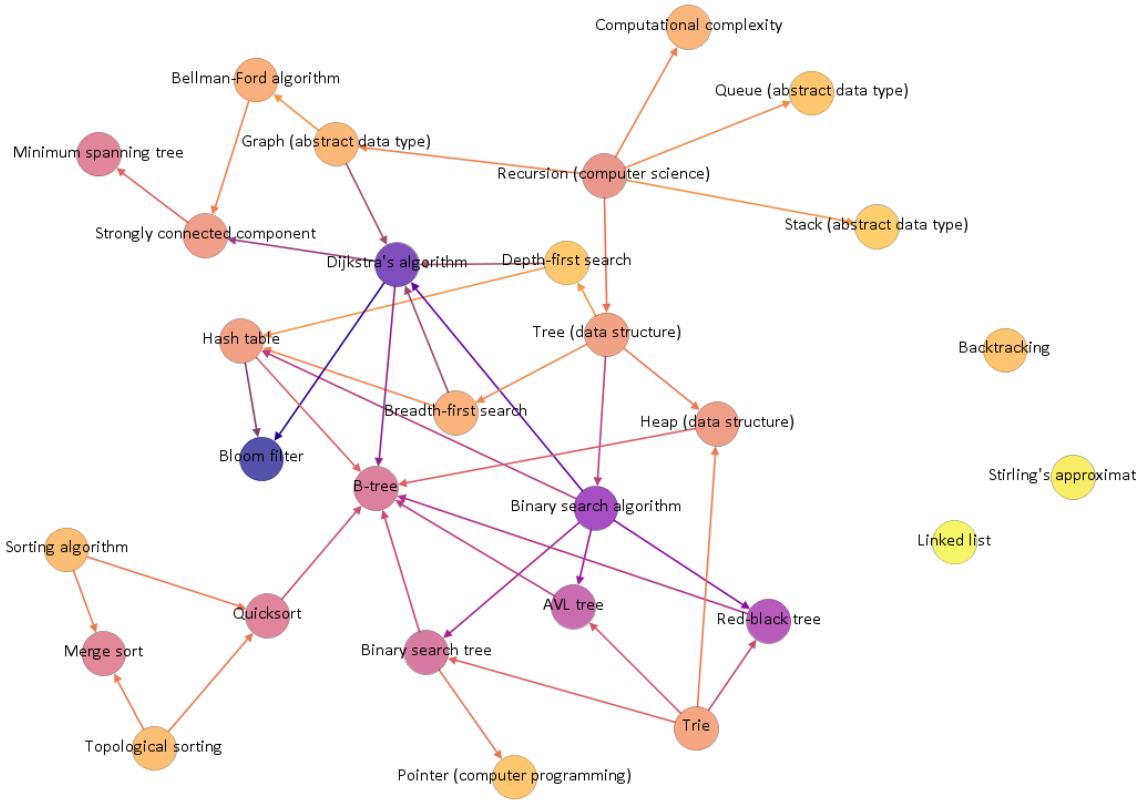

Figure 5.2: Resulting concept map for DSA according to our proposed methodology, where nodes represent concepts from DSA, edges represent the coverage relation according to Section 4.2.5. Similarly, the node color encodes concept difficulty, where lighter colors correspond to simpler concepts and darker colors to harder concepts.

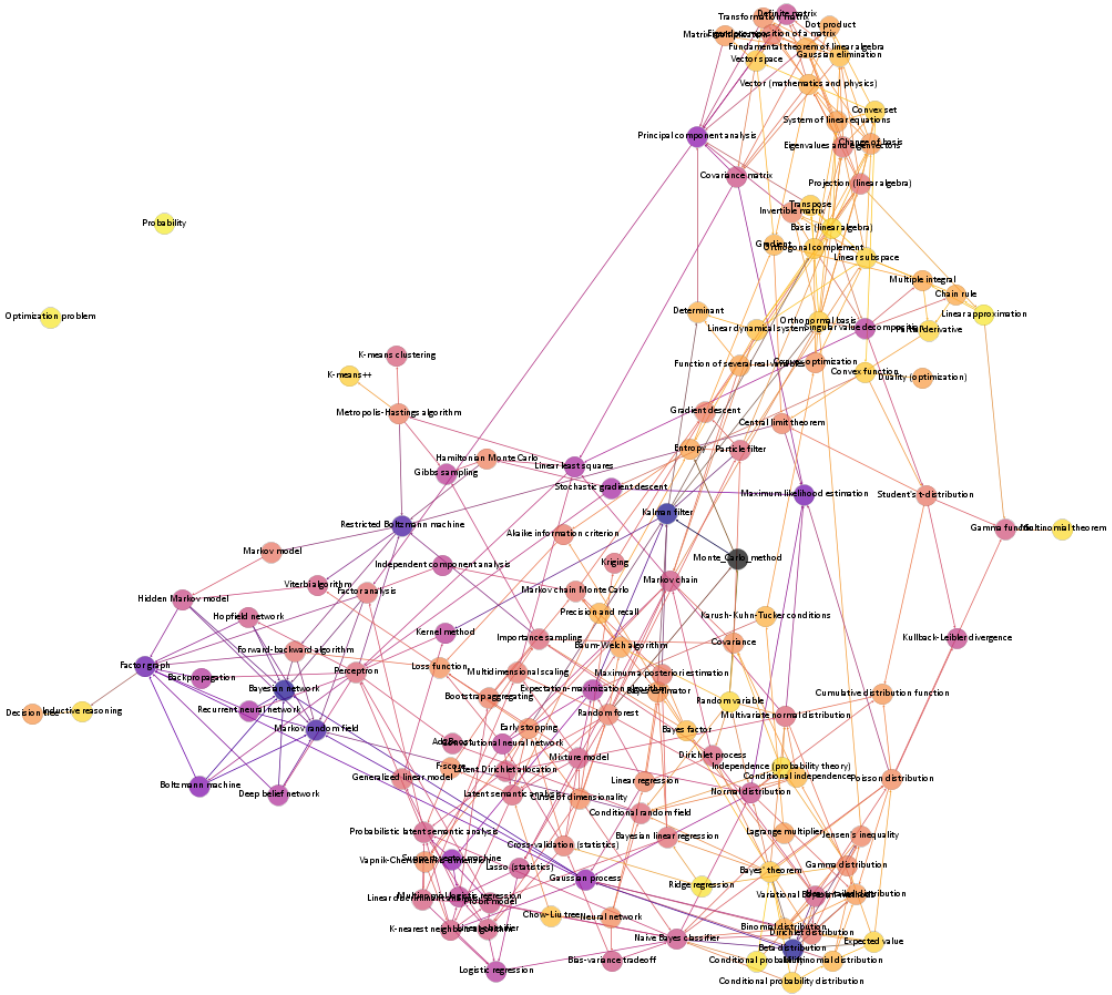

Figure 5.3: Resulting concept map for ML according to our proposed methodology, where nodes represent concepts from ML, edges represent the coverage relation according to Section 4.2.5. Similarly, the node color encodes concept difficulty, where lighter colors correspond to simpler concepts and darker colors to harder concepts.

difficulty as node color. We only clustered those scores in Section 5.2.2 to evaluate our method based on ground truth discrete difficulty labels of concepts. When constructing the concept map, not all edges need to be shown. For instance, if there is an edge from concept c_i to c_j, an edge from c_j to c_k, and an edge from c_i to c_k, the last edge is redundant because there is already a path connecting c_i with c_k, thus the edge from c_i to c_k does not add any new information, it only clutters the graph. The edge from c_i to c_k is known as a transitive edge. Therefore, after constructing the concept map, we remove transitive edges by computing its transitive reduction [75]. The statistics of both resulting concept maps are shown in Table 5.15. Most notably the percentage of transitive edges in a concept map increases with the number of concepts, thus it becomes important to declutter the resulting map by removing such transitive edges. The concept maps are also suitable to evaluate our hypothesis that *"prerequisites of a given concept tend not to be harder than the concept itself"*(H2). To that end we consider the true prerequisites from ML and DSA, respectively, and count for how many of those H2 holds.

The concept map for DSA is depicted in Fig. 5.2. It is directly clear from the node coloring that the majority of concepts is easy. Only "Bloom filter" and "Dijkstra's algorithm" stand out as difficult concepts. With background knowledge about DSA it is immediately clear that "Binary search tree" being a prerequisite of "Pointer (computer programming)" is incorrect, which is more likely to be noticed by an individual with no background knowledge because a more difficult concept, "Binary search tree", is a prerequisite for a simpler one. But applying this heuristic in case of "Recursion (computer science)" being a prerequisite of "Computational complexity" would be incorrect because the identified prerequisite relation is correct, although the prerequisite is harder than the concept itself. When checking all correct prerequisites to analyze H2, it turns out that 39/52 (75%) of the prerequisites are not harder than the given concept in DSA. Therefore, our heuristic seems applicable. Although we depict the resulting concept map of ML in Fig. 5.3, it is only included for the sake of completeness, as too many concepts with too many edges exist, which makes the concept map hard to read without zooming into certain areas, which is impossible to do in this thesis. However, we note that "Beta

distribution" is the hardest concept, followed by "Kalman filter" and "Bayesian network". While there are also examples where our heuristic expressed in H2 holds and fails, when looking at all true prerequisites in ML, 208/306 (68%) of the prerequisites are not harder than the given concept. This indicates again that our heuristic appears to be useful.

5.2.6 Discussion and Conclusion(CODIF)

We proposed an unsupervised method for the task of predicting conceptual text complexity. Our goal was to estimate the difficulty of concepts from the educational domain to construct a concept map automatically based on the concepts extracted from unstructured textual learning materials. The nodes in this concept map encode their difficulty based on node color, while we use coverage relationships among concepts to interconnect them. We argued that this relation combined with the difficulty score equips individuals with the ability to learn concepts in a specific sequence that is reminiscent of the prerequisite order because we assumed that prerequisites for a given concept are not harder than the concept itself. Thus, even in case of incorrect relations, individuals will notice and ignore such edges because hard concepts should not be related to easier concepts. Our unsupervised method for estimating the difficulty of a given concept extracts features based on the concept's position in the hierarchical structure of DBpedia, the given textual description of the concept as well as background information needed for understanding that specific concept. Aggregating those features into a score yields the approximate concept difficulty. When analyzing in Section 5.2.2 with regard to RQ1 how well CODIF performs, it turns out that CODIF outperforms a majority label baseline and features from the category Specificity worsen CODIF's performance as it seems that the information these features capture overlaps with features from Background Information and Content, but the latter two are more accurate at distinguishing easy from difficult concepts. Hence, CODIF only uses the features from those two categories. Instead of using a specific feature category to label existing prerequisite datasets automatically, in Section 5.2.3 we use only the number of prerequisites as a feature to predict the difficulty label as this information is already present in such datasets, which is why this feature can be extracted easily without needing

any preprocessing or expensive feature extraction procedure.

The experiment conducted in Section 5.2.4 stresses the difficulty of predicting prerequisite relations as neither the well-established RefD metric nor our coverage relation based on the DBpedia Hierarchy feature detect prerequisite relations reliably (see Table 5.14). Nevertheless, based on the semantic authority of both types of relations, the DBpedia Hierarchy feature is preferable, because it captures more prerequisites than RefD due to a higher recall, which is the more important metric compared to precision as missing prerequisite edges are worse for individuals than including redundant edges. As a consequence of this experiment, the concept maps for ML and DSA in Section 5.2.5 contain more edges than needed. But the maps provide evidence in favor of our assumption that prerequisites of a concept tend to be at most as difficult as the concept itself. This, in turn, suggests that our general idea for constructing a concept map based on the coverage relation with node colors denoting concept difficulty is a viable approach, while the insertion of edges poses the biggest challenge due to the poor performance of detecting prerequisites with either the prerequisite or the coverage relation.

Although simple in nature, our method performs well in the evaluation on our two newly created and released benchmark datasets. Moreover, the resulting concept maps supported our hypothesis that prerequisites of a concept tend to be not harder than the concept itself. Therefore, the idea of constructing concept maps with our methodology is viable, but inserting reliable relationships among the concepts poses the biggest challenge. As a byproduct of the labeling process for the two benchmark datasets we also experimented with a heuristic to automatically assign difficulty labels to concepts based on their number of prerequisites. An advantage of this heuristic is that no expensive feature extraction procedure is required. However, this information is only available for prerequisite datasets, and in our experiments the heuristic produced difficulty labels of mixed quality and it is unclear how reliable it is in general. This does not rule out the possibility that there might be an inherent link between prerequisite detection and conceptual text complexity, but it indicates that other aspects, such as how a concept is described, will probably also have to be included.

One drawback of our method is that it relies on the existence of a concept in Wikipedia or DBpedia. But given that Wikipedia and DBpedia are constantly getting updated, this limitation might vanish over time. Another avenue for addressing this problem is to combine our method with fuzzy matches. More precisely, we will combine our method with the idea that a language model, such as BERT[70], already represents a knowledge base[71]. Currently our method takes into account features from DBpedia, the concept description, and background information. However, when the experts labeled our two datasets, they mentioned that some Wikipedia articles were considered easy or hard based on the visualizations - either images, animations, or videos. While our current method already yields good results, integrating features from those visualizations into our approach would potentially improve the performance. Student assessment data would constitute another promising factor to integrate into our method, similar to [38] and [39], who rely only on this assessment data. However, this student assessment data would only be available if our method were to be integrated into a fully fledged e-learning platform. In such an e-learning platform additional features could be considered that take into account an individual's cognitive capacity, analytical reasoning skills [76] as they all affect how concept difficulty is perceived subjectively. This additional information could be used to provide a sequence of learning materials tailored to an individual [77]. Annotating more datasets for the task of predicting conceptual text complexity has a high priority in the future. This would allow re-evaluating our proposed heuristic to label prerequisite datasets automatically with difficulty labels based on the idea that harder concepts have more prerequisites. Our obtained mixed results on two datasets do not allow to draw any conclusions about the usefulness of our heuristic to mitigate the scarceness of suitable benchmark datasets yet. A closely related research avenue concerns evaluating the quality of the automatically derived difficulty labels with our simple heuristic when assigning more nuanced difficulty labels and not only the binary case. Our proposed heuristic is flexible enough to accommodate more granular difficulty labels. Our plan is to analyze this in the future not only on our two released datasets about conceptual text complexity, but more prerequisite datasets to see if our simple heuristic is robust or just an artifact. Related to this

avenue is also testing our methods on courses from other domains than computer science, because some features, most notably Mathematical Complexity, might be inappropriate for concept descriptions that do not contain any mathematical equations. Last but not least, we also plan to investigate the role of language for predicting concept difficulty as common readability scores failed to correlate with concept difficulty in our preliminary experiments. One reason might be their simplistic nature, thus a more sophisticated approach like [78] might be able to distinguish difficult linguistic concept descriptions more accurately.

Table 5.2: Macro-averaged F1-scores of COBEC and COBEC-T versus seven baseline methods on seven different datasets. Note that the scores of our method with * are obtained with the GOLD+WIKI labels, whereas the others utilize only GOLD labels (see Section 5.1.1 for an explanation). The best performance on each dataset is marked in bold.

(a) First (COBEC(1)) and second phase of COBEC (COBEC(2)).

Dataset	COBEC(1)	COBEC(2)	YAKE!	MultiPR	TopicR	TextR	KEA	TF-IDF	MonkL
SemEval	0.124	**0.132**	0.112	0.079	0.070	0.028	0.096	0.093	0.103
DM	0.288	**0.307**	0.101	0.068	0.043	0.010	0.079	0.080	0.082
Wiki20	0.174 0.216*	**0.197** 0.241*	0.094	0.071	0.063	0.003	0.092	0.084	0.074
OS	0.448	**0.510**	0.186	0.325	0.255	0.023	0.172	0.068	0.092
DB	0.263	**0.321**	0.182	0.126	0.113	0.001	0.119	0.045	0.207
Theses100	0.0906	**0.123**	0.090	0.100	0.088	0.004	**0.120**	0.073	0.064
Nguyen2007	0.209	**0.227**	0.212	0.173	0.136	0.036	0.176	0.170	0.161

(b) First (COBEC-T(1)) and second phase of COBEC-T (COBEC-T(2)).

Dataset	COBEC-T(1)	COBEC-T(2)	YAKE!	MultiPR	TopicR	TextR	KEA	TF-IDF	MonkL
SemEval	0.094	0.100	**0.112**	0.079	0.070	0.027	0.096	0.093	0.103
DM	0.242	**0.254**	0.101	0.068	0.043	0.010	0.077	0.080	0.082
Wiki20	0.166 0.214*	**0.184** 0.230*	0.094	0.071	0.063	0.003	0.092	0.084	0.074
OS	0.310	**0.380**	0.186	0.325	0.255	0.023	0.172	0.068	0.092
DB	0.244	**0.293**	0.182	0.126	0.113	0.001	0.119	0.045	0.207
Theses100	0.078	0.093	0.090	0.100	0.088	0.004	**0.120**	0.073	0.064
Nguyen2007	0.196	**0.212**	**0.212**	0.173	0.136	0.036	0.176	0.170	0.161

Table 5.3: Macro-averaged precision scores for COBEC and COBEC-T versus seven baseline methods. The best performance on each dataset is marked in bold.

Dataset	COBEC(1/2)	COBEC-T(1/2)	YAKE!	MultiPR	TopicR	TextR	KEA	TF-IDF	MonkL
SemEval	(0.094/0.164)	(0.102/ 0.078)	0.103	0.059	0.051	0.010	**0.160**	0.091	0.103
DM	(0.373/0.393)	(0.320/**0.333**)	0.126	0.066	0.056	0.005	0.05	0.046	0.087
Wiki20	(0.273/0.306)	(0.266/**0.300**)	0.160	0.120	0.106	0.01	0.166	0.081	0.074
OS	(0.866/0.943)	(0.600/**0.733**)	0.288	0.466	0.400	0.021	0.333	0.133	0.063
DB	(0.265/0.314)	(0.248/**0.285**)	0.177	0.125	0.113	0.007	0.116	0.041	0.204
Theses100	(0.087/0.133)	(0.083/**0.120**)	0.066	0.074	0.065	0.003	0.088	0.054	0.061
Nguyen2007	(0.264/0.234)	(0.243/0.213)	**0.266**	0.209	0.166	0.045	0.221	0.190	0.207

Table 5.4: Macro-averaged recall scores for COBEC and COBEC-T versus seven baseline methods. The best performance on each dataset is marked in bold.

Dataset	COBEC(1/2)	COBEC-T(1/2)	YAKE!	MultiPR	TopicR	TextR	KEA	TF-IDF	MonkL
SemEval	(0.164/0.127)	(0.140/0.098)	0.011	0.059	0.052	0.011	0.073	0.099	**0.103**
DM	(0.232/0.249)	(0.198/**0.210**)	0.085	0.041	0.035	0.006	0.032	0.030	0.097
Wiki20	(0.125/0.138)	(0.122/0.135)	0.067	0.051	0.045	0.004	0.071	0.071	**0.151**
OS	(0.302/0.348)	(0.209/**0.255**)	0.112	0.162	0.139	0.029	0.116	0.146	0.140
DB	(0.272/0.342)	(0.253/**0.313**)	0.194	0.133	0.118	0.007	0.128	0.053	0.219
Theses100	(0.097/0.192)	(0.082/0.166)	0.141	0.150	0.131	0.005	**0.180**	0.054	0.062
Nguyen2007	(0.204/0.257)	(0.191/**0.238**)	0.206	0.174	0.135	0.036	0.176	0.182	0.155

Table 5.5: Macro-averaged similarities between the central node and the underlying topic (SINGLE-SIM), average similarity between the underlying topic and all nodes in the DBPedia graph (AVG-SIM) and the respective p-value of a one-sided Wilcoxon signed-rank test, where significant differences (p=0.05) are highlighted with *.

Dataset	SINGLE-SIM	AVG-SIM	(test statistic, p-value)
DM	0.522	0.493	(W=132, p=0.16)
DB	0.557	0.486	(W=272, p=0.02)*

Table 5.6: Comparison of the 10 top-ranked concepts extracted from the dataset OS using our method, TextRank (TextR) and MonkeyLearn (MonkL).

Rank	COBEC	TextR	MonkL
1	Stack_pointer, Activation_record, Call_stack	Process systems	Process
2	Process	User process systems	Multiple cpu machine
3	Kernel, Operating_system_kernel	Process systems	Collection of instructions
4	Interrupt, Software_interrupt	Various process states	Program
5	PCB , Process_control_block	Several processes	Time
6	Call, System_call	Independent processes	Modern computer system
7	Input, User_input	Parallel processes	Multithreading
8	Context_switch	Active processes	Several process
9	Interrupt_handler, Handler	Distinguish process	Computer architecture
10	Thread	Independent process	Computer program

Table 5.7: Macro-averaged F1-scores of COBEC per dataset when varying β, where we set $\beta = P_c$ and P_c denoting that only those candidate concepts correspond to actual concepts if they are at least in the c-th percentile of the top-ranked candidate concepts after the second phase.

Dataset	P_{75}	P_{50}	P_{25}	R-precision
SemEval	0.115	0.129	0.129	0.128
Wiki20	0.174	0.201	0.185	0.203
DM	0.279	0.262	0.225	0.279
OS	0.400	0.422	0.421	0.418
Theses100	0.113	0.121	0.134	0.129
Nguyen2007	0.170	0.220	0.207	0.217
DB	0.264	0.304	0.304	0.299

Table 5.8: Descriptive statistics of our newly created datasets ML and DSA.

Dataset	Concepts	Easy	Hard
DSA	29	22 (76%)	7 (24%)
ML	140	87 (62%)	53 (38%)

Table 5.9: Performance of CODIF when using only specific feature categories for predicting concept difficulty. The best performance per dataset is highlighted in bold font.

Feature Category	Dataset	Accuracy	F1-score	Precision	Recall
Majority Label (Baseline)	DSA	0.76	0.86	0.76	1.0
	ML	0.62	0.77	0.62	1.0
All	DSA	0.66	0.77	0.77	0.77
	ML	0.59	0.60	0.76	0.48
Content	DSA	0.69	0.81	0.76	0.86
	ML	0.62	0.77	0.63	**0.99**
Specificity	DSA	0.69	0.80	0.79	0.81
	ML	0.46	0.36	0.68	0.24
Background Information	DSA	0.73	0.82	0.82	0.82
	ML	0.73	0.78	0.78	0.78
Content + Background Information	DSA	**0.79**	**0.87**	**0.83**	**0.90**
	ML	**0.75**	**0.80**	**0.81**	0.78
Content + Specificity	DSA	0.66	0.76	0.80	0.73
	ML	0.46	0.36	0.68	0.24
Background Information + Specificity	DSA	0.66	0.78	0.75	0.82
	ML	0.57	0.57	0.76	0.45

Table 5.10: Confusion tables per dataset comparing the automatically derived label with the true label.

(a) ML.

Ground Truth \ Automatic Label	Easy	Hard
Easy	81	27
Hard	6	26

(b) DSA.

Ground Truth \ Automatic Label	Easy	Hard
Easy	13	9
Hard	4	3

Table 5.11: Accuracy, F1-score, precision, recall when predicting the difficulty labels based on the number of prerequisites of each concept according to Hypothesis H1.

Dataset	Accuracy	F1-score	Precision	Recall
DSA	0.59	0.68	0.81	0.59
ML	0.77	0.84	0.76	0.93

Table 5.12: Confusion matrix per dataset comparing the overlap between prerequisites according to DBpedia Hierarchy and the true prerequisite edges.

(a) ML.

Prerequisite \ DBpedia Hierarchy	Yes	No
Yes	51	255
No	1168	8257

(b) DSA.

Ground Truth \ DBpedia Hierarchy	Easy	Hard
Easy	16	36
Hard	38	330

Table 5.13: Confusion matrix per dataset comparing the overlap between prerequisite edges according to RefD and the true prerequisite edges.

(a) ML.

Prerequisite \ RefD	Yes	No
Yes	181	125
No	5018	4475

(b) DSA.

Ground Truth \ RefD	Easy	Hard
Easy	23	29
Hard	209	159

Table 5.14: Accuracy, F1-score, precision, recall for predicting prerequisite edges based on DBpedia Hierarchy and RefD.

Method	Dataset	Accuracy	F1-score	Precision	Recall
DBpedia Hierarchy	ML	0.85	0.067	0.17	0.04
	DSA	0.82	0.30	0.31	0.30
RefD	ML	0.48	0.066	0.59	0.03
	DSA	0.43	0.16	0.44	0.10

Table 5.15: Summary statistics of the concept maps: number of nodes (#Nodes), number of edges before transitive reduction (#Edges (before)), number of edges after transitive reduction (#Edges (after), percentage of transitive edges that were removed from the concept map (% Transitive edges).

Dataset	#Nodes	#Edges (before)	#Edges (after)	% Transitive edges
DSA	29	54	41	24%
ML	140	1219	398	67%

Chapter 6

Conclusion and Future Work

This chapter first sums up the thesis and then discusses implications of our findings. Afterwards, future research directions that go beyond the topics discussed in Chapters 4 and 5 are outlined.

6.1 Summary

While having a concept map visualizing coverage relations among concepts, whose node colors encode concept difficulty, offers many benefits to learners, to date there is no method available that constructs such a concept map in an automatic fashion. Hence, we set out in this thesis to develop methods that work with any unstructured textual learning materials to construct such a map automatically. To accomplish this goal, we divided this task into two separate subtasks, namely extracting concepts from unstructured textual learning materials and then constructing the respective concept map, which involves computing a difficulty score per concept as well as determining coverage edges among the concepts. The difficulty score provides additional information to individuals, which helps them discern potentially unreliable from reliable coverage edges based on the assumption that prerequisites of a concept might not be harder than the actual concept itself. Using coverage instead of prerequisite relations creates a concept map with less strict recommendations for individuals for studying concepts in a certain sequence. which, in turn,

allows inaccuracies due to the additional visual cues provided by coloring nodes according to their difficulty scores as prerequisites tend to not be more difficult than the given concept to learn. Therefore, this thesis proposed to estimate prerequisite relations with a combination of node color and coverage edges.

Proposing a novel unsupervised method for concept extraction was motivated by the fact that existing methods failed to consider context information. This idea proved to be helpful for identifying the most relevant concepts in learning materials, given the context, because a valid concept may be relevant in one context and completely irrelevant in another one. More specifically, the context information was exploited for disambiguating concepts and for determining if the disambiguated sense of the concept was relevant in the given context or not. As a result, the conducted experiments confirmed that our method outperformed state-of-the-art methods.

For the second subtask we proposed an unsupervised method for estimating concept difficulty that extracts different features about a concept. Given that the task of predicting conceptual text complexity was introduced only recently, we focused on non-lexical features that affect how difficult a concept is perceived to be. Therefore, our extracted features were based on the hierarchical structure of DBpedia, the given textual description of this concept, and background information required for understanding that specific concept. Aggregating those features resulted in their difficulty scores. Edges among concepts were added in the concept map based on the closeness of concepts according to the hierarchical structure of DBpedia. Although simple in nature, our method for concept difficulty prediction performed surprisingly well in the evaluation, while the quality of the inserted edges leaves room for improvement in the future.

6.2 General Conclusion

When evaluating our unsupervised method for the first subtask, we found that it outperformed existing state-of-the-art methods on four existing benchmark datasets and also three newly created datasets that were particularly created for the educational domain.

Those new datasets cover the topics ”Data Mining”, ”Database Management Systems”, and a book chapter from operating systems about ”Process”. Our method is flexible as it expects unstructured textual descriptions for concepts, while making two assumptions about the concepts to be extracted. For one, they must have existing DBpedia entries. Secondly, relevant concepts are related to a broader, overarching topic. While this might sound restrictive, there is no limitation on how abstract the overarching topic should be. For example, ”Data Mining” or ”Database Management Systems” are suitable choices.

As part of the evaluation process of our method for computing a difficulty score per concept in the second subtask, we created two new datasets covering the educational domain, which are ”Machine Learning” and ”Data Structures and Algorithms”, because no datasets existed for this domain before. This is due to the fact that the task of predicting the difficulty of entire documents, which represent textual descriptions of concepts in our case, has only been introduced recently [11]. Thus, with the release of these two new datasets, we hope to stimulate more research related to the task of conceptual text complexity. Related to that goal we also explored the possibility of automatically assigning difficulty labels to benchmark datasets that were created for prerequisite detection. The heuristic for assigning those labels automatically was based on the idea that harder concepts have more prerequisites, which is only available as reliable information in datasets that were specifically labeled for the task of prerequisite detection. Due to the limited number of datasets for evaluation, we cannot draw any conclusions about the reliability of our heuristic yet as we observed mixed results on our two datasets. Therefore, more datasets are needed to draw conclusions about a potential connection between the task of prerequisite detection and predicting conceptual text complexity. However, the key assumption of our proposed concept map, namely that a concept’s prerequisites tend to be not more difficult than the concept itself, turned out to hold in most cases. Thus, our idea of replacing prerequisite relations by a combination of coverage edges and difficulty scores, expressed by node color, in a concept map seems promising. The main limitation of our proposed methods is that only concepts with a DBpedia or Wikipedia entry are processed. But due to both of these resources constantly getting updated with new entries,

this appears to be a minor drawback. One way to mitigate this problem is to explore the idea that a language model, such as BERT[70], already represents a knowledge base[71], which might be sufficient for our purposes.

6.3 Future Work

While this thesis proposes methods for constructing a concept map in which node color represents a concept's difficulty and directed edges indicate a coverage relation, the resulting concept map could be enhanced if the visualization were available in an e-learning platform. Apart from having additional data such as student assessments available for estimating concept difficulty more accurately, it would be possible to personalize the concept map. For example, given the performance of other learners on the platform, their performance could be mapped to the nodes and contrasted with the performance of the logged in individual visually in the map. However, that would require additional information about which concepts are covered by each assessment question because normally each question covers more than a single concept. Quantifying the contribution of each concept to each question poses another research problem that requires to be addressed before it can be visualized accordingly. Another important future research direction is devising a novel unsupervised method for extracting coverage or prerequisite edges because our experiments showed the difficulty of this particular task.

We can extract pre-requisite relation by utilizing the "Background feature" of CODIF. We implemented RefD and used DBpedia hierarchy feature to find the number of pre-requisite for a concept. It is pertinent to mention that we used some heuristics to compute DBPH feature by using DBpedia ontology instead of directly using the ontological structure. The combination of these two features may result in more accurate pre-requisite relations.

Bibliography

[1] M. R. Genesereth and N. J. Nilsson, *Logical foundations of artificial intelligence*. Morgan Kaufmann, 2012.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *The semantic web*, pp. 722–735, Springer, 2007.

[3] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge acquisition*, vol. 5, no. 2, pp. 199–220, 1993.

[4] J. D. Novak and A. J. Cañas, "The theory underlying concept maps and how to construct them," *Florida Institute for Human and Machine Cognition*, vol. 1, pp. 2006–2001, 2006.

[5] C. C. Yang, H. Chen, and K. Hong, "Visualization of large category map for internet browsing," *Decision support systems*, vol. 35, no. 1, pp. 89–102, 2003.

[6] N. Chen, Kinshuk, C. Wei, and H. Chen, "Mining e-learning domain concept map from academic articles," *Computers & Education*, vol. 50, no. 3, pp. 1009–1021, 2008.

[7] S. Wang, A. Ororbia, Z. Wu, K. Williams, C. Liang, B. Pursel, and C. L. Giles, "Using prerequisites to extract concept maps from textbooks," in *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, pp. 317–326, 2016.

[8] P. P. Talukdar and W. W. Cohen, "Crowdsourced comprehension: Predicting prerequisite structure in wikipedia," in *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP, BEA@NAACL-HLT 2012, June 7, 2012, Montréal, Canada*, pp. 307–315, 2012.

[9] C. Liang, Z. Wu, W. Huang, and C. L. Giles, "Measuring prerequisite relations among concepts," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, pp. 1668–1674, 2015.

[10] S. Roy, M. Madhyastha, S. Lawrence, and V. Rajan, "Inferring concept prerequisite relations from online educational resources," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 9589–9594, Jul. 2019.

[11] S. Štajner and I. Hulpuș, "Automatic assessment of conceptual text complexity using knowledge graphs," in *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 318–330, 2018.

[12] K. Barker and N. Cornacchia, "Using noun phrase heads to extract document keyphrases," in *Advances in Artificial Intelligence, 13th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, AI 2000, Montréal, Quebec, Canada, May 14-17, 2000, Proceedings*, pp. 40–52, 2000.

[13] J. Villalon and R. A. Calvo, "Concept extraction from student essays, towards concept map mining," in *2009 Ninth IEEE International Conference on Advanced Learning Technologies*, pp. 221–225, IEEE, 2009.

[14] T. Tomokiyo and M. Hurst, "A language model approach to keyphrase extraction," in *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment*, pp. 33–40, 2003.

[15] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, 1972.

[16] R. Mihalcea and P. Tarau, "Textrank: Bringing order into text," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004, 25-26 July 2004, Barcelona, Spain*, pp. 404–411, 2004.

[17] A. Bougouin, F. Boudin, and B. Daille, "Topicrank: Graph-based topic ranking for keyphrase extraction," in *International joint conference on natural language processing (IJCNLP)*, pp. 543–551, 2013.

[18] F. Boudin, "Unsupervised keyphrase extraction with multipartite graphs," *arXiv preprint arXiv:1803.08721*, 2018.

[19] X. Wan and J. Xiao, "Single document keyphrase extraction using neighborhood knowledge," in *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*, pp. 855–860, 2008.

[20] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, "Yake! keyword extraction from single documents using multiple local features," *Information Sciences*, vol. 509, pp. 257–289, 2020.

[21] A. G. Parameswaran, H. Garcia-Molina, and A. Rajaraman, "Towards the web of concepts: Extracting concepts from large datasets," *PVLDB*, vol. 3, no. 1, pp. 566–577, 2010.

[22] A. Krishnan, A. Sankar, S. Zhi, and J. Han, "Unsupervised concept categorization and extraction from scientific document titles," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 1339–1348, 2017.

[23] R. Mihalcea and A. Csomai, "Wikify! linking documents to encyclopedic knowledge," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pp. 233–242, 2007.

[24] T. P. Tanon, G. Weikum, and F. Suchanek, "Yago 4: A reason-able knowledge base," in *European Semantic Web Conference*, pp. 583–596, Springer, 2020.

[25] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning, "Domain-specific keyphrase extraction," in *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 99, Stockholm, Sweden, July 31 - August 6, 1999. 2 Volumes, 1450 pages*, pp. 668–673, 1999.

[26] A. Hulth, "Improved automatic keyword extraction given more linguistic knowledge," in *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pp. 216–223, 2003.

[27] P. D. Turney, "Learning algorithms for keyphrase extraction," *Inf. Retr.*, vol. 2, no. 4, pp. 303–336, 2000.

[28] P. D. Turney, "Coherent keyphrase extraction via web mining," in *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*, pp. 434–442, 2003.

[29] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, "Kea: Practical automated keyphrase extraction," in *Design and Usability of Digital Libraries: Case Studies in the Asia Pacific*, pp. 129–152, IGI global, 2005.

[30] Q. Zhang, Y. Wang, Y. Gong, and X. Huang, "Keyphrase extraction using deep recurrent neural networks on Twitter," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, (Austin, Texas), pp. 836–845, Association for Computational Linguistics, Nov. 2016.

[31] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.

[32] Y. Zhang, M. Tuo, Q. Yin, L. Qi, X. Wang, and T. Liu, "Keywords extraction with deep neural network model," *Neurocomputing*, vol. 383, pp. 113–121, 2020.

[33] X. Yan, D. Song, and X. Li, "Concept-based document readability in domain specific

information retrieval," in *Proceedings of the 15th ACM international conference on Information and knowledge management*, pp. 540–549, 2006.

[34] D. Seyler, M. Yahya, and K. Berberich, "Generating quiz questions from knowledge graphs," in *Proceedings of the 24th International Conference on World Wide Web*, pp. 113–114, 2015.

[35] S. Štajner and I. Hulpuș, "When shallow is good enough: Automatic assessment of conceptual text complexity using shallow semantic features," in *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 1414–1422, 2020.

[36] A.-M. Vercoustre, J. Pehcevski, and V. Naumovski, "Topic difficulty prediction in entity ranking," in *International Workshop of the Initiative for the Evaluation of XML Retrieval*, pp. 280–291, Springer, 2008.

[37] F. Gasparetti, C. De Medio, C. Limongelli, F. Sciarrone, and M. Temperini, "Prerequisites between learning objects: Automatic extraction based on a machine learning approach," *Telematics and Informatics*, vol. 35, no. 3, pp. 595–610, 2018.

[38] R. Scheines, E. Silver, and I. M. Goldin, "Discovering prerequisite relationships among knowledge components.," in *EDM*, pp. 355–356, 2014.

[39] A. Vuong, T. Nixon, and B. Towle, "A method for finding prerequisites within a curriculum.," in *EDM*, pp. 211–216, 2011.

[40] M. Morsey, J. Lehmann, S. Auer, and A.-C. N. Ngomo, "Dbpedia sparql benchmark–performance assessment with real queries on real data," in *International semantic web conference*, pp. 454–469, Springer, 2011.

[41] I. Hulpus, C. Hayes, M. Karnstedt, and D. Greene, "Unsupervised graph-based topic labelling using dbpedia," in *Proceedings of the sixth ACM international conference on Web search and data mining*, pp. 465–474, 2013.

[42] M. Young, D. Lambert, C. Roberts, and M. Roberts, *Knowledge and the future school: Curriculum and social justice*. Bloomsbury Publishing, 2014.

[43] M. Newman, *Networks: An Introduction*. Oxford University Press, 2010.

[44] A. Miaschi, C. Alzetta, F. A. Cardillo, and F. Dell'Orletta, "Linguistically-driven strategy for concept prerequisites learning on italian," in *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 285–295, 2019.

[45] R. Manrique, B. Pereira, and O. Mariño, "Exploring knowledge graphs for the identification of concept prerequisites," *Smart Learning Environments*, vol. 6, no. 1, pp. 1–18, 2019.

[46] L. Pan, C. Li, J. Li, and J. Tang, "Prerequisite relation learning for concepts in moocs," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1447–1456, 2017.

[47] G. Adorni, C. Alzetta, F. Koceva, S. Passalacqua, and I. Torre, "Towards the identification of propaedeutic relations in textbooks," in *International Conference on Artificial Intelligence in Education*, pp. 1–13, Springer, 2019.

[48] M. C. Aytekin, S. Rabiger, and Y. Saygın, "Discovering the prerequisite relationships among instructional videos from subtitles," in *Proceedings of the 13th International Conference on Educational Data Mining*, pp. 569–573, EDM, 2020.

[49] S. Gul, S. Räbiger, and Y. Saygın, "Context-based extraction of concepts from unstructured textual documents," *Information Sciences*, 2021.

[50] G. L. Murphy, "Comprehending complex concepts," *Cognitive science*, vol. 12, no. 4, pp. 529–562, 1988.

[51] J. Mothe and L. Tanguy, "Linguistic features to predict query difficulty," in *ACM Conference on research and Development in Information Retrieval, SIGIR, Predicting query difficulty-methods and applications workshop*, pp. 7–10, 2005.

[52] E. von Glasersfeld, "The problem of syntactic complexity in reading and readability," *Journal of Reading Behavior*, vol. 3, no. 2, pp. 1–14, 1970.

[53] R. Szostak, "Complex concepts into basic concepts," *Journal of the American Society for Information Science and Technology*, vol. 62, no. 11, pp. 2247–2265, 2011.

[54] J. K. Nelson, M. A. Hjalmarson, K. E. Wage, and J. R. Buck, "Students' interpretation of the importance and difficulty of concepts in signals and systems," in *2010 IEEE Frontiers in Education Conference (FIE)*, pp. T3G–1–T3G–6, 2010.

[55] J. Sweller and P. Chandler, "Why some material is difficult to learn," *Cognition and instruction*, vol. 12, no. 3, pp. 185–233, 1994.

[56] R. M. Gagné, "1: Learning and instructional sequence," *Review of research in education*, vol. 1, no. 1, pp. 3–33, 1973.

[57] J. Sweller, "Cognitive load theory, learning difficulty, and instructional design," *Learning and instruction*, vol. 4, no. 4, pp. 295–312, 1994.

[58] E. Dale and J. S. Chall, "A formula for predicting readability: Instructions," *Educational research bulletin*, pp. 37–54, 1948.

[59] J. P. Kincaid, R. P. Fishburne Jr, R. L. Rogers, and B. S. Chissom, "Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel," tech. rep., Naval Technical Training Command Millington TN Research Branch, 1975.

[60] S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin, "Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles," in *Proceedings of the 5th International Workshop on Semantic Evaluation*, pp. 21–26, 2010.

[61] O. Medelyan, I. H. Witten, and D. Milne, "Topic indexing with wikipedia," in *Proceedings of the AAAI WikiAI workshop*, vol. 1, pp. 19–24, 2008.

[62] T. D. Nguyen and M.-Y. Kan, "Keyphrase extraction in scientific publications," in *International conference on Asian digital libraries*, pp. 317–326, Springer, 2007.

[63] C. C. Aggarwal, *Data mining: the textbook*. Springer, 2015.

[64] R. Ramakrishnan and J. Gehrke, *Database management systems*, vol. 3. McGraw-Hill New York, 2003.

[65] F. Boudin, "pke: an open source python-based keyphrase extraction toolkit," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, (Osaka, Japan), pp. 69–73, December 2016.

[66] A. Coxhead, "A new academic word list," *TESOL quarterly*, vol. 34, no. 2, pp. 213–238, 2000.

[67] E. Vylomova, L. Rimell, T. Cohn, and T. Baldwin, "Take and took, gaggle and goose, book and read: Evaluating the utility of vector differences for lexical relation learning," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Berlin, Germany), pp. 1671–1682, Association for Computational Linguistics, Aug. 2016.

[68] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[69] A. Joulin, É. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 427–431, 2017.

[70] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[71] C. Wang, X. Liu, and D. Song, "Language models are open knowledge graphs," *arXiv preprint arXiv:2010.11967*, 2020.

[72] W. Coster and D. Kauchak, "Simple english wikipedia: a new text simplification

task," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 665–669, 2011.

[73] W. Xu, C. Callison-Burch, and C. Napoles, "Problems in current text simplification research: New data can help," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 283–297, 2015.

[74] A. J. Viera, J. M. Garrett, *et al.*, "Understanding interobserver agreement: the kappa statistic," *Fam med*, vol. 37, no. 5, pp. 360–363, 2005.

[75] A. V. Aho, M. R. Garey, and J. D. Ullman, "The transitive reduction of a directed graph," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 131–137, 1972.

[76] W. Schneider, "The development of metacognitive knowledge in children and adolescents: Major trends and implications for education," *Mind, Brain, and Education*, vol. 2, no. 3, pp. 114–121, 2008.

[77] W. Schneider, J. Körkel, and F. E. Weinert, "Domain-specific knowledge and memory performance: A comparison of high-and low-aptitude children.," *Journal of educational psychology*, vol. 81, no. 3, p. 306, 1989.

[78] Y. Fujinuma and M. Hagiwara, "Semi-supervised joint estimation of word and document readability," in *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pp. 150–155, 2021.