KNOWLEDGE GRAPH CREATION FROM STRUCTURE

A Dissertation in

Computer Science and Engineering

by

Shuting Wang

© 2017 Shuting Wang

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

August 2017
The dissertation of Shuting Wang was reviewed and approved* by the following:

C. Lee Giles
David Reese Professor of Information Science and Technology
Professor of Computer Science and Engineering
Professor of Supply Chain and Information Systems
Dissertation Advisor
Chair of Committee

Jesse Barlow
Professor of Computer Science and Engineering
Professor of Statistics

George Kesidis
Professor of Computer Science and Engineering
Professor of Electrical Engineering

Bruce Desmarais
Professor of Political Science
Professor of the Institute for Cyber Science

Chita R. Das
Distinguished Professor of Computer Science and Engineering
Professor and Graduate Program Coordinator

*Signatures are on file in the Graduate School.
Abstract

Knowledge graphs, which organize and structure knowledge by linked concepts, have been widely used as teaching and learning tools for science education tasks. Typically, they are manually created by domain experts to serve as “ontology” or “knowledge base” for different purposes. However, with the tremendous growth of massive online educational data, automating the creation of concept maps becomes necessary. The task is challenging as it requires not only extracting domain concepts from educational content but also identifying the semantic relations among them.

This thesis proposes to perform concept map extraction from high-quality academic resources, such as textbooks and research papers, inside of which more meaningful structures and semantics can be leveraged. Moreover, we leverage other types of structure knowledge such as web knowledge bases to help create the concept map. With rich semantic information and a large-scale hyperlink network, it has been widely used in learning and education. We first present a work on concept hierarchy extraction using rich structure in textbooks. However, this work did not explicitly identify concept relations embedded in the textbooks. To resolve this, we perform joint optimization for concept extraction and prerequisite relations identification using rich book structures. In the last work, instead of only being interested in the prerequisites, we aim to extract a concept map with multiple types of relationships, such as “is-a” and “has-part”. Experimental results show that our proposed model achieves the state-of-the-art concept map extraction result on concept maps, manually created from six textbooks.

Lastly, this thesis performs some pilot studies on using the automatically extracted concept map for educational purposes. Two systems are introduced in this thesis. One is BBookx - an automatic textbook creation system[^1]. Another is an automatic assessment system using prerequisite concept maps.

[^1]: https://bbookexp.psu.edu/
# Table of Contents

<table>
<thead>
<tr>
<th>List of Figures</th>
<th>viii</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Tables</td>
<td>ix</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>xi</td>
</tr>
</tbody>
</table>

## Chapter 1

### Introduction

<table>
<thead>
<tr>
<th>1.1 Research Challenge</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1.1 Determining Domain Key Concepts</td>
<td>2</td>
</tr>
<tr>
<td>1.1.2 Extracting Concepts from Scientific Literature</td>
<td>3</td>
</tr>
<tr>
<td>1.1.3 Concept Relations for Educational Purposes</td>
<td>3</td>
</tr>
<tr>
<td>1.1.4 Data Resources</td>
<td>4</td>
</tr>
<tr>
<td>1.1.5 Scale</td>
<td>5</td>
</tr>
</tbody>
</table>

| 1.2 Research Methods | 5 |

| 1.3 Research Questions | 8 |

| 1.4 Structure of this Thesis | 10 |

## Chapter 2

### Related Work

<table>
<thead>
<tr>
<th>2.1 Automatic Knowledge Graphs Creation in e-Learning</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1.1 Knowledge Graph Creation from Structured Textual Data</td>
<td>13</td>
</tr>
<tr>
<td>2.1.2 Knowledge Graph Creation from Unstructured Data Sources</td>
<td>14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2.2 Knowledge Graph Embedding</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.1 Translation-based Embedding</td>
<td>16</td>
</tr>
<tr>
<td>2.2.2 Structured &amp; Unstructured Embedding</td>
<td>18</td>
</tr>
<tr>
<td>2.2.3 Neural Network based Models</td>
<td>18</td>
</tr>
<tr>
<td>2.2.4 Factor Models</td>
<td>19</td>
</tr>
</tbody>
</table>

| 2.3 Keyphrase Extraction using Wikipedia | 19 |
Chapter 3

Concept Map Construction from Structure Knowledge

3.1 Concept Hierarchies Extraction from Textbooks

3.1.1 Problem Definition & Approach

3.1.1.1 Concept Hierarchy Extraction from Books

3.1.1.2 Local and Global Concept Hierarchy Extraction from Books

3.1.2 Concept Hierarchy Extraction from Books

3.1.2.1 Domain Specific Concept Identification

3.1.2.2 Candidate Selection

3.1.2.3 Concept Hierarchy Generation

3.1.2.3.1 Local Features

3.1.2.3.2 Global Features

3.1.2.3.3 Redundancy Features and Consistency Features

3.1.2.3.4 Learning Order features

3.1.2.4 Concept Hierarchy Extractor Training

3.1.3 Data Sets and Evaluation Metrics

3.1.3.1 Data Preparation and Experiment Setup

3.1.3.2 Evaluation Metrics

3.1.4 Experiments and Results

3.1.4.1 Baseline Method

3.1.4.2 Book-Level Concept Hierarchy Extraction

3.1.4.3 Subchapter-Level Concept Hierarchy Extraction

3.2 Prerequisite Concept Maps Extraction from Textbooks

3.2.1 Problem Definition

3.2.2 Joint Knowledge Graph Extraction from Textbooks

3.2.2.1 Concept Map Extraction

3.2.2.1.1 Joint Modeling

3.2.2.1.2 Optimization

3.2.2.2 Representation Schemes

3.2.2.2.1 Word Based Similarity

3.2.2.2.2 Word Embedding

3.2.2.2.3 Wikipedia Anchors

3.2.2.2.4 Textbook Structure

3.2.3 Experiment Settings

3.2.3.1 Dataset

3.2.3.2 Baseline - Key Concept Extraction
### 3.2.3.1 TextRank

- Supervised Key Concept Extraction (Supervised KCE)

### 3.2.3.2 Wikify

- Baseline - Prerequisite Relationship Identification
  - Hyponym-Hypernym
  - Supervised Relationship Identification (Supervised RI)

### 3.2.3.3 Baseline - Prerequisite Relationship Identification

- Hyponym-Hypernym
  - Supervised Relationship Identification (Supervised RI)

### 3.2.3.4 Wikipedia Coverage of Concepts

### 3.2.3.5 Parameter Selection

### 3.2.3.6 Model Initialization

### 3.2.4 Experimental Results

- Effect of Textbook Information
- Joint Optimization
- Measurement Importance

### 3.2.5 Case Study

### 3.3 Multi-Relation Concept Map Extraction

- Distant Supervision Relation Extraction
- Data Preparation and Experiment Settings
  - Data Preparation

### 3.4 Experimental Results

- Key Concept Extraction
- Prerequisite Relation Identification
- Prerequisite Concept Map Delivery

### Chapter 4

**Applications with Concept Map**

- BBookx: Dynamic Book Creation
  - System Overview
- Automated Assessment using Prerequisite Concept Maps
  - Prerequisite Concept Map Construction
    - Key Concept Extraction
    - Prerequisite Relation Identification
    - Prerequisite Concept Map Delivery
  - Experiments
    - Experiments Setup

### Chapter 5

**Implications**

- Implications for Knowledge Graph Construction using Academic Resources
  - Hierarchical Structure Extraction
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1.2 Knowledge Graph with Prerequisite Relationships</td>
<td>107</td>
</tr>
<tr>
<td>5.1.3 Performance for Different Textbooks</td>
<td>109</td>
</tr>
<tr>
<td>5.2 Dataset for Knowledge Graph Extraction</td>
<td>110</td>
</tr>
<tr>
<td>5.2.1 Textbooks</td>
<td>110</td>
</tr>
<tr>
<td>5.2.2 Domain Dictionaries</td>
<td>111</td>
</tr>
<tr>
<td>Chapter 6</td>
<td></td>
</tr>
<tr>
<td>Conclusion and Future Work</td>
<td>112</td>
</tr>
<tr>
<td>Bibliography</td>
<td>118</td>
</tr>
</tbody>
</table>
List of Figures

1.1 System architecture for knowledge graph construction ........................................ 7
3.1 Example of an extracted concept hierarchy ......................................................... 29
3.2 Workflow of the CHEB system .......................................................... 36
3.3 Example of an extracted concept map in economics ............................................ 54
3.4 System workflow for concept map extraction with joint modeling ................. 62
3.5 Precision@n (n=1,3,5) for key concept extraction from six textbooks. *Local* refers to the supervised learning model using local features defined in Section 3.2.3.2.3 with same experiment settings as *Supervised KCE* ...................................................... 75
3.6 Precision@n (n=1,3,5) for key concept extraction from six textbooks with/without joint optimization ................................................................. 77
3.7 Case study for geometry concept maps ............................................................... 80
3.8 Example of an extracted concept map in economics .......................................... 86
4.2 A running example for using prerequisite concept map to help student fix learning gap ................................................................. 96
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Symbol notation</td>
<td>28</td>
</tr>
<tr>
<td>3.2</td>
<td>Physical characteristics of books</td>
<td>46</td>
</tr>
<tr>
<td>3.3</td>
<td>Book-level experimental results</td>
<td>51</td>
</tr>
<tr>
<td>3.4</td>
<td>Subchapter-level experimental results</td>
<td>52</td>
</tr>
<tr>
<td>3.5</td>
<td>Physical characteristics of books. # labeled pairs is the number of candidate concept pairs labeled as whether two concepts have prerequisite relationships. # pairs with relationships is the number of concept pairs with prerequisite relationships in all the labeled pairs.</td>
<td>69</td>
</tr>
<tr>
<td>3.6</td>
<td>Extracted Lexico-syntax patterns. NP1 represents a subsequent Noun Phrase (NP) and NP2 represents a prerequisite Noun Phrase (NP).</td>
<td>72</td>
</tr>
<tr>
<td>3.7</td>
<td>F-1 score for concept relationship prediction. * Hyponym refers to the hyponym-hypernym baseline method. Textbook/Wikipedia are supervised learning methods with features textbooks/Wikipedia features using same experiment settings as Supervised RI. * indicates when textbook features are statistically significantly better (p &lt; 0.01) than the Wikipedia features.</td>
<td>76</td>
</tr>
<tr>
<td>3.8</td>
<td>F-1 score for concept relationship prediction with/without joint optimization. * indicates when the joint optimization model is statistically significantly (p &lt; 0.01) better.</td>
<td>78</td>
</tr>
<tr>
<td>3.9</td>
<td>Precision@n (n=1,3,5) for key concept extraction from six textbooks using different feature combinations. Title/Content features include local and global features derived from title/content information as defined in Section 3.2.3.2.3.</td>
<td>78</td>
</tr>
<tr>
<td>3.10</td>
<td>F1-score@n (n=1,3,5) for the relationship prediction using different measures.</td>
<td>79</td>
</tr>
<tr>
<td>3.11</td>
<td>F-1 score for different feature combinations.</td>
<td>88</td>
</tr>
<tr>
<td>4.1</td>
<td>Accuracy for mathematics concept map extraction</td>
<td>102</td>
</tr>
</tbody>
</table>
4.2  Accuracy for big data concept map extraction  . . . . . . . . . . . 103
Acknowledgments

I would like to express my gratitude to my academic advisor Dr. C. Lee, who not only provided me with valuable suggestions for my researches, but also supported and encouraged me to overcome many difficulties through my PhD studies. I would also like to thank Dr. Bart Pursel and Dr. Becky Passonneau for their great advise through this process.

I would also like to thank my dissertation committee, Dr. Jesse Barlow, Dr. George Kesidis, and Dr. Bruce Desmarais for their interests and time in serving as committee members.

Finally, I am very grateful to my father, Chengye Wang, and mother, Caihong Hu for their continued support and suggestions throughout my PhD studies. I also want to give special thank to my grandmother, Zhimin Wang. It would be impossible for me to be here today without her love and guidance.
Chapter 1

Introduction

A knowledge graph, in which nodes represent concepts and edges reflect relationships between concepts, exhibits a natural way to organize and represent knowledge and has been widely used for teaching and learning in education, with various applications, such as documenting and exploring concept-changing, knowledge-sharing, and knowledge-acquisition. A knowledge graph is especially useful for education. It provides a roadmap of knowledge for learners who start with a “blank sheet” and want to build a structure for domain knowledge. When experts for specific learning fields are not available, a knowledge graph can provide the skeleton of domain knowledge which can be very helpful.

Previous works typically relied upon domain experts to construct knowledge graph, which is time-consuming and of high cost. Emerging new domains also have made knowledge graph construction increasingly challenging due to its ongoing developmental nature. Under this scenario, automatic knowledge graph creation
is becoming increasingly attractive. Moreover, the rapid growth of massive online education resources has made automatically constructed domain knowledge graph increasingly attractive and brings new challenges and opportunities to enhance concept extraction.

This thesis focuses on the problem of automatically constructing knowledge graphs for science education. We leverage academic resources, specifically textbooks, to help with knowledge graph extraction. Moreover, we utilize web knowledge bases such as Wikipedia and wiki data to enrich textbook contents and connect inside-book knowledge with external knowledge.

1.1 Research Challenge

Automatic knowledge graph construction, which usually contains two difficult sub-tasks: first, extracting all domain key concepts, and second, identifying the semantic relations among them, has the following challenges described in the following sections.

1.1.1 Determining Domain Key Concepts

Automatic knowledge graph systems encounter difficulties in determining concepts relevant to the problem area. To define concepts and relationships more precisely, it is important to remove noisy concepts in the extraction phase. Therefore, a challenging task in automatic knowledge graph construction is to narrow the
extracted concept to the chosen domain. There have been works using domain
dictionaries to resolve these problems \cite{16, 60}. However, most previously used
dictionaries were created by experts and not available for many domains. Therefore,
one of our research challenges is to address the problem of automatically determining
domain key concepts.

1.1.2 Extracting Concepts from Scientific Literature

Key phrase extraction is an important research topic in the natural language
processing domain and has been well studied by many researchers \cite{26, 63, 116}.
With the increasing availability of scientific literature provided by systems like
citeseerx \cite{53}, semantic scholar \footnote{http://allenai.org/semantic-scholar/}, and google scholar \footnote{https://scholar.google.com/}
key phrase extraction scientific literature \cite{41, 42} has attracted more and more attention. Scientific
literature usually contains more complicated sentence structure compared to news
articles or social network posts, which widely serve as data resources in key phrase
extraction. On the other hand, there is rich structured information embedded in
scientific literature which can be utilized in key phrase extraction.

1.1.3 Concept Relations for Educational Purposes

While concept relations have been studied in various contexts \cite{3, 22, 101}, little effort
has been made in the educational domain. Tackling the two tasks on educational
content could expect several challenges. First, educational contents from textbooks, slides, Massive Open Online Course (MOOC) resources, etc., differ from general web content or content in other domains; thus, existing concept/relation extraction techniques cannot be directly applied here while new techniques that could better use the specialty of educational content should be developed. For example, the structural information, such as a table of contents (TOC) of a textbook could be helpful in identifying concept relations. Second, other than general concept relations such as “equal”, “is-a”, “part-of”, etc., a more important relation we focus on is the prerequisite dependency among concepts that plays a significant role in supporting curricula planning for students and course designing for teachers. Therefore, it is crucial to investigate different types of relations, such as prerequisite relations or learning orders between concepts.

1.1.4 Data Resources

Data resources play an important role in knowledge graph construction. Metadata of high-quality data can improve the quality extracted knowledge graphs and vice versa. For instance, the domain dictionary extracted from high-quality data can help determine and position domain concepts precisely while the domain dictionary extracted from low-quality data might introduce a lot of noise. Besides the quality of the data resources, another challenge about metadata is how to integrate different resources. For instance, different knowledge bases, such as Wikipedia and academic
articles, provide different contexts for each concept. How to absorb and integrate knowledge from different resources would be problematic.

1.1.5 Scale

Scale refers to the size of the concepts which are considered in knowledge graph construction. It is a key issue in domain concepts determination and relations inference. The proposed knowledge graph extraction framework should be able to operate at the scale of concepts within common domain subjects, such as mathematics and physics.

1.2 Research Methods

Previous work on knowledge graph construction used relational databases [35], domain ontologies [33,40,45], and university course descriptions [119], which often lack coverage and consistency for concepts for a specific domain. For example, course descriptions only cover a small fraction of the domain concepts and might have inconsistent usage of a particular concept.

We propose to study the extraction task using high-quality structure knowledge, such as academic papers and textbooks. Academic articles present comprehensive and well-organized knowledge of a specific domain and have been used as the major educational resource in schools and universities. In addition, the structures and semantics inside an academic article could benefit the extraction
algorithms. For instance, textbooks provide organized units of knowledge and a balanced and chronological presentation of information. As such, they are a high-quality information resource for knowledge graph extraction.

There are a few previous studies focusing on using textbooks and research papers [47, 104]. However, most have only made use of the textual information within the academic articles, not leveraging the rich structure of the academic articles. Moreover, without connecting the inside-the-article knowledge to external knowledge, these methods cannot guarantee the quality of the results in terms of interpretability and reusability.

Instead of solely using academic articles as resources, we propose a method for extracting knowledge graphs from academic articles with the help of web knowledge. Our work fits into the growing amount of web knowledge which offers significant opportunities to enhance learning for students by encouraging knowledge-sharing and supporting dynamic interactions among learners [21, 106]. The web offers students an alternative learning resource. A growing amount of web knowledge offers significant opportunities to enhance the learning of students by encouraging knowledge-sharing and supporting dynamic ways of interactions of people [21, 106]. Compared to traditional education sources, web knowledge enables geographically dispersed people and of different cultures to communicate, collaborate, and share experiences. Moreover, web knowledge is more up-to-date and easier to access. Wikipedia is one example of a free-access web knowledge base. By allowing users to
create and make comments on pages, Wikipedia encourages users to collaborate and is becoming increasingly important to facilitate learning among students. Therefore, we leverage Wikipedia, a free-access web knowledge base that contains more than 4 million concepts, to assist in knowledge graph extraction.

![Key Concept Extractor](Key Concept Extractor.png) ![Key concepts](Key concepts.png) ![Document](Document.png) ![Concept Relationship Identifier](Concept Relationship Identifier.png) ![Domain Concept Map](Domain Concept Map.png)  

**Figure 1.1:** System architecture for knowledge graph construction.

Figure 1.1 visually depicts the idea of using academic articles to construct domain knowledge graphs with the help of Wikipedia. In the Figure, a piece of a document and some web knowledge base is provided for the knowledge graph construction system. The system first performs domain key concepts extraction and then derives concept relations.
1.3 Research Questions

Based on the discussion above, the following are proposed as research questions to be investigated as part of this research.

- **Different levels of abstractions and details**

  Education materials for the same subject usually deliver concepts of different difficulty levels to different users depending on their expertise or desired level of complexity. For instance, instructors who teach mathematics will teach basic math concepts, such as “Addition” and “Subtraction” to 2th-grade students, while advanced concepts, such as “Derivatives” and “Gaussian Distribution” to 12th-grade students. Similarly, one concept may be interpreted at different levels of abstraction and detail. For instance, when a student is studying the concept “Addition”, he may first work with “Integer Addition”, then extend to “Rational Number Addition”, and then “Complex Number Addition”. In this case, it is crucial for us to consider the level of granularity. Therefore, attractive research questions to ask are: 1) How could we obtain an ordered concept list which represents the concept difficulty level to help select the desired level of detail when performing knowledge graph construction? 2) How could we decide the difficulty level of different content associated with a concept and construct knowledge graphs which range from easy to difficult?

- **Integrate Academic Article Structures into knowledge graph Ex-**
traction  Academic articles are usually well organized and written for knowledge transmission. The rich structure contained in academic articles provides valuable information for concept relations inference. This research seeks to investigate how article structure information can be integrated into knowledge graph construction. For instance, the order between two concepts within a textbook could be integrated as features for concept relations identification. Alternatively, a co-training approach could be used where one identifier is trained on the article’s contents and the other on the article’s structure.

- **Integrate Information from web Knowledge Base into knowledge graph Extraction**  The web offers students an alternative learning resource. A growing amount of web knowledge offers significant opportunities to enhance learning of students by encouraging knowledge-sharing and supporting dynamic ways of interactions of people [21,106]. While the rich information contained in the web knowledge base can help knowledge graph extraction, one question is how to integrate information from the web knowledge base, which is a challenging problem. Most of the web knowledge bases are crowdsourcing-based information resources and usually contain noisy information. It is a challenging task to perform information cleaning to better utilize web knowledge bases.

- **knowledge graph Evaluation**  An important problem emphasized in automatic knowledge graph construction is the need for an objective evaluation
method that can be used for the comparison of knowledge graph construction results. Previous works evaluate their knowledge graphs by comparing the machine-extracted knowledge graphs with the human-extracted maps since humans know how to capture important concepts and produce a well-formed knowledge graph. The problem with human evaluators, however, is that different people look at the source content in their own way, and the evaluation is mostly subjective. The quality of the evaluation is difficult to control. To reduce this problem, previous works introduce inter-humans between several experts for knowledge graph evaluation [19,56]. Although the map agreed by several skillful humans can be used as a gold standard for evaluation, this method is time-consuming and of large human cost.

1.4 Structure of this Thesis

In proposing the work described above, the rest of this thesis is structured as follows. Chapter 2 discusses related work focusing on automated knowledge graph construction since it is the focus of the proposed work. Chapter 3 describes the work performed thus far on automated knowledge graph construction. A generic system for knowledge graph construction is described, as well as an algorithm for improving performance in knowledge graph construction and then applications in knowledge graph construction. We mainly address three types of knowledge graphs: 1) a knowledge graph organized using a textbook structure, so-called
concept hierarchies; 2) a knowledge graph with prerequisite relations; and 3) a knowledge graph with multiple relations such as “is-a” and “has-part”. We also discuss two systems which use/potentially use knowledge graphs for educational purposes in Chapter 4. Chapter 5 then discusses implications from the results presented in previous chapters for both knowledge graph extraction and its education applications. Lastly, Chapter 6 discusses the contributions and implications of our paper and suggests an agenda for future research.
Chapter 2

Related Work

This thesis is related to several lines of work: First, our work is related to automatic knowledge graph extraction in e-learning domain; Another line of related work is knowledge graph embedding which aims to represent knowledge graph using continuous vector space; It is also related to key phrase extraction and concept relatedness which can be used in identify entities and relations in a knowledge graph.

2.1 Automatic Knowledge Graphs Creation in e-Learning

A number of work have focused on the automatic construction of knowledge graphs from different types resources.
2.1.1 Knowledge Graph Creation from Structured Textual Data

Sources

Ontology, which can formalized as triple subject-predict-object\(^{62}\), are widely used in knowledge graph generation\(^{33,10,45}\). The basic approach of using ontology in knowledge graph construction is through a direct mapping of ontology instances and associations into concepts and relationships in knowledge graphs.\(^{40}\) proposed a method to translate ontology of the English vocabulary into a knowledge graph by first identifying instances in the ontology and then translating the synonyms, intersections, and unions among instances into relationships between concepts. A similar algorithm is proposed in\(^{33}\) where the first phase of the algorithm identified instances of classes and the second phase extracted synonyms, intersections, and unions among classes and translated these associations into relationships between concepts.\(^{45}\) creates a visualization for knowledge graph generation from an ontology by starting with a user-specified concept and extracting other concepts from ontology according to their relationships with the starting concept.\(^{10}\) constructed knowledge graphs using WordNet\(^{1}\), to disambiguate the sense of a word from a knowledge graph, with the map itself as its context.

Ontology is also used to enrich knowledge graph construction or handle errors in knowledge graphs.\(^{43}\) provides an interactive system for knowledge graph construction where mines the web to suggest concepts that could enhance the

\(^{1}\)https://wordnet.princeton.edu/
map and handle errors such as misspelled concepts or synonym through the use of WordNet.

Learners’ testing records have been used to automatically construct knowledge graphs by mining grade fuzzy association rules in the testing records of learners and inference prerequisite relationships between learning concepts \([104]\), to discover concept-effect relationships for diagnosing students’ learning problem \([37]\) and to diagnose the learning barriers and misconception of learners through knowledge graphs constructed from testee’s test portfolio \([50]\). \([2,13]\) applies fuzzy rules and fuzzy reasoning techniques and evaluate the relevance degrees between concepts for knowledge graph construction using learners’ testing records. \([48]\) formulates knowledge graphs from online discussion boards using fuzzy ontology.

### 2.1.2 Knowledge Graph Creation from Unstructured Data Sources

Unstructured texts are another important sources in knowledge graph generation. There are several groups of applicable techniques proposed in previous work.

Statistical methods to analyze concept importance within a domain and relations among concepts. Term co-occurrences are used in selecting domain key concepts in knowledge graph construction. \([9]\) uses co-occurrences between terms to suggest a list of suggested concepts that are relevant to the map under construction. \([14]\) uses a list of important terms selected by the researcher as seed concept to look for terms with high co-occurrences with these concepts to extend the extracted concept
list. This technique is also applied to reveal concept relations \[28,105\]. Term Frequency-Inverse Document Frequency (tf-idf) is also used to measure concept correlation in knowledge graph construction \[48,49,89,124\].

These methods tend to be efficient and do not depend on a specific language or domain. However, they are usually imprecise because they did not consider the semantics relatedness between terms.

Text mining has been applied to automatically construct knowledge graphs from academic articles. Clustering techniques are also applied in knowledge graph extraction. A hybrid algorithm which uses grammar analysis and fuzzy clustering techniques is proposed in \[86\]. \[12\] investigated a two-phase model to construct domain key concept which first extracted domain key concepts and then identified relationships among concepts in e-learning domain. Latent semantic indexing and K-means are used in \[96\] discovering topic similarity between concepts and serve as guidance in topic ontology building.

Dictionaries with domain concepts and concept relations such as synonyms and metonymy are also used to extract concepts and relations more precisely within in a domain. Dictionaries can also help minimize the number of candidate terms in a construction phase. In the bio-medical domain, researchers \[16,60\] used the Medical Subject Headings (MeSH) \(^2\) and Unified Medical Language System (UMLS) \(^3\) ontology as a dictionary to help knowledge graph construction.

\(^2\)https://www.nlm.nih.gov/mesh/
\(^3\)https://www.nlm.nih.gov/research/umls/
Recently, increasing number of works are using large-scale of academic articles such as textbooks and papers to construct a knowledge graph. \cite{59} studied learning-dependency between knowledge units from documents that cover four domains using text association mining. \cite{47} performs a framework for the semiautomatic generation of the domain module which is a structure for knowledge representation similar to knowledge graph from electronic textbooks using natural language processing techniques and heuristic reasoning. \cite{119} proposed a learning-to-rank approach to explore prerequisite relationships among courses and constructed a concept graph based on the relationships. \cite{44} generates a reading tree for papers by measuring the generality score of a paper and the overlap between two papers. Index terms in textbooks provide lists of important concepts within a domain and are used knowledge graph construction \cite{80,105}.

2.2 Knowledge Graph Embedding

Knowledge graph embedding aims to project entities and relations into continuous vector space. Several lines of related work in knowledge graph embedding are listed below.

2.2.1 Translation-based Embedding

The basic translation principle of this line of work is $h + r \approx t$, where $h$ and $t$ are entity embedding vectors projected in the relation-specific space. The score
function is defined as:

\[
    f_r(h, t) = \|h_r + r - t_r\|_2^2
\]  

(2.1)

Different lines of previous work assign different assumptions for the space that entity embedding vector project into. \cite{6} assumes that same entities in different relations share the same entity embedding vectors and sets the entities in the original entity space: \( h_r = h \) and \( t_r = t \). However, this assumption does not always hold to one-to-many relation. For instance, entity “Laptop” has both “Screen” and “Keyboard”. However, based on the assumption in \cite{6}, “Screen” and “Keyboard” are going to have the same representation vector, which does not make sense. To address one-to-many and many-to-one relations, later work proposed to project entity embedding vector into different space \cite{58,111}. TransH \cite{111} models a relation as a hyperplane and model the entity embedding as: \( h_r h - w_r^T h w_r \), \( h_r h - w_r^T h w_r \). TransR \cite{58} projects the entity embeddings using the relation specific matrix: \( h_r = M_r h, t_r = M_r t \). A clustering-based method, CTransR, is also proposed in \cite{58}. The basic idea of CTransR is to cluster the entity pairs for a relation into different groups, and the pairs in the same group share the same relation vector. Another issue of training knowledge graph embeddings is that not all the training samples leverage the relations among entities. To address this issue, TransM \cite{24} utilizes the structure of the knowledge graph via pre-calculating the distinct weight for each training triple. In \cite{118}, transG address the issue of multiple relation semantics that a relation may have multiple meanings revealed by the entity pairs associated
with the corresponding triples. Except for considering relations between different
etransformations and corresponding relations, there are many work to enhance knowledge
graph embedding by utilizing other information. [34] proposed Semantically
Smooth Embedding which aims to enforce the embedding space to be semantically
smooth, i.e., entities which are semantically close will lie close to each other in
the embedding space. [108] incorporate the constraints translated from rules and
formulates inference as an integer linear programming problem. [57] also utilizes
the path information in the knowledge graph to enhance embeddings.

2.2.2 Structured & Unstructured Embedding

[7] transforms entity embeddings with the left and right hand relation matrices for
the relation $r_k$ and define the similarity function for a given entity as $S_k(E_i, E_j) =
\|R_{lhs}^{r_k}E_i - R_{rhs}^{r_k}E_j\|$. However, this model can not capture the relationship between
etities. Later, Semantic Matching Energy [5,7] was proposed to handle correlations
between entities by performing matrix product on left entity and left hand-side
relation matrix, right entity and right hand-side relation matrix, and finally merging
these two functions together.

2.2.3 Neural Network based Models

[99] proposed Single Layer Model which applies a neural network to knowledge
graph embedding which connects the entity vectors implicitly through the nonlin-
earity of a single layer neural network. In the same work, they proposed *Neural Tensor Network*, which replaces a standard linear neural network layer with a bilinear tensor layer and directly relates the two entity vectors. In this work, each entity is represented as an average of their constituting word vectors.

### 2.2.4 Factor Models

[102] proposed to capture the second-order correlation between entities by a quadratic form and defined the score function as $f_r(h, t) = h^T W_r t$. [74] and [75] investigate RESCAL, a tensor factorization for relation learning for knowledge base embedding.

### 2.3 Keyphrase Extraction using Wikipedia

Keyphrase extraction typically proceeds in two steps: 1) identifying a list of candidate keyphrases, and 2) re-rank the candidates to determine which keyphrases are correct and this field has been studied by many researchers. In this thesis, we focus on work that uses Wikipedia as a resource for keyphrase extraction since this line of work is closely related to our problem.

[8] first explores Wikipedia as a resource for detecting key phrases in open domain text. They trained a disambiguation SVM kernel which compared the lexical context around the ambiguous named entity to the content of the candidate Wikipedia page to perform disambiguation on named entities. Mihalcea and Csomai
performs automatic keyword extraction and word sense disambiguation with Wikipedia by training a Naive Bayes classifier and using the hyperlink information in Wikipedia as ground truth. proposed Wikipedia-based key phrases as a candidate’s document frequency multiplied by the ratio of the number of Wikipedia articles where the candidate appears as a link to the number of articles where it appears. The assumption of this feature that a candidate is likely to be a keyphrase if it occurs frequently as a link in Wikipedia. Semantic relatedness between Wikipedia candidates are also considered to obtain a coherent disambiguation on named entities. Essentially, besides the content similarity between the entity and Wikipedia candidates, Wikipedia articles selected from the same article should be semantically close to each other. These works also stressed the semantic relatedness between the Wikipedia candidates and optimized the disambiguation results.

2.4 Concept Relatedness

There have been many studies investigating different techniques for computing Wikipedia concept relatedness. Broadly speaking, Wikipedia-based approach to concept relatedness can be split between those using lexical contents in the articles and those using links between articles.

derives a model called *WikiRelate!* which searches for Wikipedia article that contain two given Wikipedia concepts in their titles and computes the semantic relatedness using various distance measures between titles of those retrieved pages.
represents each word as a weighted vector of Wikipedia concepts, and computes semantic relatedness between two concepts by comparing the two concept vectors. 

derives measures using information contents and text overlap.

The above work use a single-hop reasoning, i.e., perform reasoning on individual relations to predict relation. However, this line of work ignore the transitivity between relationships. For instance, we can infer a relation “PlaysInLeague(Tom Brady, NFL)” from the facts “PlaysForTeam(Tom Brady, New England Patriots)” and “PartOf(New England Patriots, NFL)”. Therefore, another line of work performs relation reasoning over multi-hops. introduced the Path Ranking Algorithm (PRA), which improves efficiency and robustness by replacing exhaustive search with random walks, and using unique paths as features in a per-target-relation binary classifier. modifies the Path Ranking Algorithm, which learns to rank a given node relative to the query node, by adding weights obtained from the statistics from a population of training queries rather than enumerated completely. improves Path Ranking Algorithm by adding addition of edges labeled with latent features mined from a large dependency parsed corpus of 500 million Web documents. Later, improves Path Ranking Algorithm by considering semantically similar to the edge type in the path. incorporates constraints on the allowed entity types for the candidate entities, to multi-hop relation extraction by including entity type information. Another problem of Path Ranking Algorithm is that it treats a relational path as an atomic features. To address this issue,
presents an approach that reasons about conjunctions of multi-hop relations non-atomically.

first investigate relatedness between two Wikipedia articles by comparing their numbers of in-links based on Normalized Google Distance, combined with several heuristics and collocation strength. In the latter work , they incorporate machine learning techniques to calculate the concept semantic relatedness. Random walk based methods are applied to derive concept relatedness. obtained concept relatedness using random walk on Wikipedia hyperlink network. applied random walk on a network derived from the Freebase Wikipedia Extraction dataset, which is richer than Wikipedia hyperlink network.

Other information like Wikipedia category and infoboxes are also used in deriving concept relatedness. investigated a Maximum Entropy model with features derived from reference links in Wikipedia articles to measure concept relatedness. In latter work , they considered different relations including category inheritance, page overlap in two categories and instance-of relations between Wikipedia page and category in the relatedness measurement. A Wikipedia path based measurement is derived in to define the semantic distance between two Wikipedia concepts. utilizes information embedded in Wikipedia infoboxes to derive concept hypernym-hyponym relations.

Except for deriving similarity between two Wikipedia concepts, there are also studies focusing on deriving another type of relations between Wikipedia articles:
prerequisite relations or learning dependencies. [103] proposes various features including PageRank score and random walk with restart score to predict prerequisite structure in Wikipedia. [55] derives a metric reference distance which considers the difference between reference links in two Wikipedia articles for prerequisite relations inference.

Graphical models are widely used in relation extraction. [11] proposes a generative probabilistic model to discover in-domain relation using constraints via posterior regularization. [120] proposes a LDA-like model which captures the selectional preferences of relations. [90] addresses noisy patterns in distant supervision using an undirected graphical model. [121] performs relation extraction across multiple documents and enforces selectional preference.

Joint inference which performs entity recognition and relation extraction is also widely used. [91] use the ILP framework to enforce manually-specified constraints between the tasks. [121] enforces selectional preference via Wikipedia. Name Entity Recognition (NER) and semantic role labeling are also used to extract relations between mentions [81][88]. Instead of only considering entity recognition and relation extraction, [97] proposes to perform joint inference of entities, relations and coreference.

Different from above work, our problem focuses on scholarly entities rather than real-world entities. Thus, assumptions that utilizes entity types such as selectional preference may not work for our case.
In Chapter 1 the use of structure knowledge, specifically, academic articles, in concept map construction was motivated. The motivation focused on utilizing the rich structure and contents contained in academic articles to construct concept map. This section describes the work done thus far involving automatic concept map construction from textbooks, which is widely used in science education. Since rich structure of academic article is a motivator for concept map construction, Section 3.1 describes an investigation into leverage table of contents of textbooks to construct concept map. The study focuses extracting key concept extraction from each book chapter formalize domain concept hierarchies utilizing local relatedness between candidate concepts and book chapter and global coherence embedded in extracted concept set. Table of contents of textbooks is effective for key concept extraction.
from domain concept map, however, does not identify concept relations explicitly. Therefore, we present another completed work which identifies concept prerequisite relations using textbooks in Section 3.2. This work proposes a two phase model which first extracts domain concepts and then identifies concept relations, and fully leverages the mutual dependency between two phases to improve the construction performance.

### 3.1 Concept Hierarchies Extraction from Textbooks

This section is based on the following work:


Early work on concept hierarchy construction relied heavily on human expertise. However, with more open educational resources, many available online, we feel automatic concept hierarchy extraction from these resources can be very useful for knowledge extraction and creation.

Textbooks provide organized units of knowledge and a balanced and chronological presentation of information. As such they are a high-quality information resource for concept hierarchy extraction. However, most previous work on concept hierarchy extraction from textbooks has only made use of the textual information within the textbooks, not leveraging the rich structure of textbooks or connect the inside-the-
book knowledge to external knowledge resources [47, 79].

We propose a method for extracting concept hierarchies from digital books using Web knowledge. Our work fits into the growing amount of Web knowledge which offers significant opportunities to enhance learning for students by encouraging knowledge sharing and supporting dynamic interactions among learners [21, 106].

Specifically, we leverage Wikipedia, a free-access Web knowledge base that contains more than 4 millions concepts, to assist in concept hierarchy extraction. For brevity we abbreviate Concept Hierarchy Extraction from Books as (CHEB).

In the CHEB task, we are given a digital textbook with its lexical content and table of contents (TOC) with the goal to extract and output a concept hierarchy for that book. To do this we extract a set of related important Wikipedia concepts for each book chapter and organize them as a concept hierarchy using the book’s TOC.

To extract the concept hierarchy, we utilize a Learning-to-Rank approach which considers both local relatedness and global coherence. We propose local features to extract related concepts for each chapter separately, utilizing measures such as textual similarity between a book chapter and candidate concepts. We also expect the extracted concept hierarchy to be globally coherent, i.e. the concept in a given chapter should also be related to other concepts in current/different subchapter(s).

In presenting these contributions, the rest of this section is laid out as follows. We first define the Concept Hierarchy Extraction from Books (CHEB)
approach and introduce its work flow in Section 3.2.1. Local and global features are introduced in Section 3.2.2. In Section 3.1.3, we discuss the data preparation and evaluation metrics. In Section 3.2.4 analyzes the experimental results for three well used textbooks and presents an example of the generated concept hierarchy.

3.1.1 Problem Definition & Approach

We first formalize our Concept Hierarchy Extraction from Books (CHEB) approach and then briefly introduce our local and global CHEB features which consider both the relatedness between extracted concepts and books and the global coherence among the extracted concepts.

3.1.1.1 Concept Hierarchy Extraction from Books

Essentially, CHEB utilizes the TOC of the book to construct a concept hierarchy by extracting related concepts in each chapter. Instead of performing keyword extraction on the book’s contents [79], we use Web knowledge to improve the concept extraction and enrich the book content. We use Wikipedia to identify important concepts in the book. For simplicity, we consider each Wikipedia title as a concept.

The input to a CHEB framework is a book $B$ with a list of titles $TB = \{tb_1, tb_2, ..., tb_N\}$ and contents $CB = \{cb_1, cb_2, ..., cb_N\}$. $tb_i$ and $cb_i$ are the title and the content for the $i^{th}$ subchapter in the TOC respectively, and $N$ is the total number
of subchapters in the book. Here we use the term “subchapter” to refer to all the
headings in the TOC and ignore the level of the headings. For instance, both 1.1
and 1.1.1 are subchapters. As for the term “chapter”, we use it to refer to a set of
subchapters whose first level chapter numbers are the same. For instance, chapter
1 may include subchapter 1, subchapter 1.1 and subchapter 1.2.

Given a book $B$ and a set of Wikipedia titles $W = \{w_1, w_2, ..., w_{|W|}\}$, our goal
is to produce a concept hierarchy which lists a set of important Wikipedia concepts
for each subchapter. We represent the output hierarchy as $\Gamma = \{cs_1, cs_2, ..., cs_N\}$
where $cs_i = \{w_1, w_2, ..., w_K\}$ is a K-tuple and $w_j \in cs_i$ is an important concept for
subchapter $j$. $CHEB$ constructs a concept hierarchy for the book by organizing
the concepts extracted from each subchapter using the book’s TOC. Figure 3.1
gives an example of the input and output of $CHEB$. The left side is the TOC of a
macroeconomics book and the right side is the concept hierarchy extracted from
the book.
3.1.1.2 Local and Global Concept Hierarchy Extraction from Books

Since the concept hierarchy uses the inherent structure of the book, our goal is to devise an algorithm that extracts a set of concepts which are related to the book chapter and also forms a “coherent” knowledge hierarchy which is consistent with the book structure. A necessary attribute of the concept hierarchy is local relatedness, i.e., the extracted concepts for a specific subchapter need to be related to the subchapter in some way. For instance, they share similar keywords or key phrases.

The local CHEB approach extracts important concepts for each subchapter independently. Specifically, given a subchapter $i$, its title $tb_i$ and content $cb_i$, let $\Phi(cs_{ij}|tb_i, cb_i)$ be the score function such that concept $cs_{ij}$ is the $j^{th}$ related concept...
in this subchapter. The local approach solves the following optimization problem:

\[ \Gamma^*_{\text{local}} = \arg \max_\Gamma \left[ \sum_{i=1}^{N} \sum_{p \in cs_i} \Phi(cs_{ip}|tb_i, cb_i) \right] \]  

(3.1)

Besides having local relatedness, we also expect that the concept hierarchy is globally coherent. Whether to put a concept in a specific subchapter is not only decided by the relatedness between the concept and the subchapter, but also by the coherence between this concept and the concepts in the same/different subchapter(s). For instance, given a book about macroeconomics, if we already rank “Gross Domestic Product” as an important concept for subchapter 1.1, we may want to lower the rank of this concept in subchapter 1.2. Therefore, we expect that the extracted concept hierarchy not only considers the local relatedness, but also preserves the global coherence. In genera for global coherence, CHEB is expected to extract a concept hierarchy with the following attributes: less redundancy in the sense chapters do not always talk about all of the same concepts; consistency with other concepts in the same chapter; and consistent learning order in that concepts follow each other as with prerequisites as discussed in Section 1.

Based on above three assumptions, global optimization for concept hierarchy
occurs when the solve the following equation:

\[ \Gamma^* = \arg \max_{\Gamma} \sum_{i} \sum_{p \in cs_i} [\Phi(cs_ip|tb_i, cb_i) - \Psi(\Gamma) + \Theta(\Gamma) + \gamma(\Gamma)] \]  

(3.2)

where \( \Phi(\cdot) \) is the local optimization function and \( \Psi(\cdot), \Theta(\cdot) \) and \( \gamma(\cdot) \) are three functions corresponding to the features proposed above.

To be specific, \( \phi(cs_ip|tb_i, cb_i) \) captures the relatedness between each candidate concept and the book chapter. \( \Psi(\cdot) \) captures the redundancy of concept hierarchy by calculating the total pairwise information overlap between concepts in different subchapters, which should be minimized. \( \Theta(\cdot) \) corresponds to the consistency feature and captures the pairwise relatedness between concepts within the same subchapter. The global consistency feature proposed above requires this function to be maximized. \( \gamma(\cdot) \) ensures that the hierarchy orders the concepts following pairwise learning order on the book level. For any concept in the hierarchy, introducing its prerequisite concept after it or its subsequent concept before it should be avoided.

Eq. 3.2 is NP-hard and approximations are needed to solve this an an optimization problem. The common approach is to estimate the pairwise relation \( \Psi(\cdot), \Theta(\cdot), \) and \( \gamma(\cdot) \) and generate approximated concept hierarchy contexts \( \Gamma_1, \Gamma_2, \) and \( \Gamma_3 \) for \( \Psi(\cdot), \Theta(\cdot), \) and \( \gamma(\cdot) \) respectively. In this work, Wikipedia content and link information are utilized to estimate the relatedness and the learning order between concepts \( w_i \) and \( w_j \), which brings two benefits: 1) a good estimation of pairwise concept relation and relatedness due to the rich semantics residing in Wikipedia content.
and links, and 2) an easy way for computing the features because Wikipedia has a
unified template for most concepts and links.

Given the estimated relation between concepts, we then solve Eq. 3.3 in an
approximate form:

\[
\Gamma^* \approx \arg \max_{\Gamma} \sum_{i=1}^{N} \sum_{p \in \mathcal{C}_i} [\Phi(c_{sp}, t_{bi}, c_{bi}) - \sum_{c_{sjq} \in \Gamma_1} \Psi(c_{sp}, c_{sjq}) \\
+ \sum_{c_{sjq} \in \Gamma_2} \Theta(c_{sp}, c_{sjq}) + \sum_{c_{sjq} \in \Gamma_3} \gamma(c_{sp}, c_{sjq})]
\] (3.3)

As we discussed above, function \(\Psi(\cdot)\) captures the redundancy in the concept
hierarchy and therefore, given a concept \(w_j\) that serves as a candidate concept in
the \(i^{th}\) subchapter, the concept hierarchy context considered for this concept (\(\Gamma_1\) in
Eq. 3.3) should be those concepts in different chapters. Notice that we are using
chapters but not subchapters here. The reason is that some books present relatively
different concepts in different subchapters while some do not. To generalize our
solution, we consider concepts in different chapters when we deal with the issue of
concept redundancy.

Similarly, we can simplify \(\Gamma_2\) and \(\Gamma_3\) for \(\Theta\) and \(\gamma\) respectively. Basically, for each
candidate concept \(w_j\) in the \(i^{th}\) subchapter, \(\Gamma_2\) only includes \(w_k\) from subchapter \(i\)
since we focus on the consistency of concepts within the same subchapter; as for
\(\Gamma_3\), which considers the learning order, we include concepts from all subchapters
except for those from the current subchapter. Therefore, we can rewrite Eq. 3.3 as
the following form:

$$
\Gamma^* \approx \arg \max_{\Gamma} \sum_{i=1}^{N} \sum_{p \in cs_i} \left[ \Phi(cs_{ip}|th_i, cb_i) \right] - \sum_{CN(j) \neq CN(i)}^{N} \sum_{q=1}^{|cs_j|} \Psi(cs_{ip}, cs_{jq}) + \sum_{q=1}^{|cs_j|} \Theta(cs_{ip}, cs_{iq}) + \sum_{i \neq j}^{N} \sum_{q=1}^{|cs_j|} \gamma(cs_{ip}, cs_{jq})
$$

(3.4)

where $\lambda(\cdot)$ is defined as a function which returns the chapter number of given a subchapter. For instance, $\lambda(1.1.1)$ and $\lambda(1.1)$ return both 1 which is the chapter number of subchapter 1.1.1 and 1.1. Therefore, $\sum_{CN(j) \neq CN(i)}^{N} \sum_{q=1}^{|cs_j|} \Psi(cs_{ip}, cs_{jq})$ is the total information overlap between candidate $cs_{ip}$ and all candidates in different chapters. We want to minimize this overlap to reduce redundancy in the concept hierarchy. 

$\sum_{q=1}^{|cs_j|} \Theta(cs_{ip}, cs_{iq})$ corresponds to the second global feature of the book such that concepts within one subchapter should be consistent. $\sum_{i \neq j}^{N} \sum_{q=1}^{|cs_j|} \gamma(cs_{ip}, cs_{jq})$ is used to capture the consistency between the learning order of the candidate concepts and the order of subchapters in the book. It can be expanded as $L(cs_{ip}, cs_{jq}) \times I(i, j)$. $L$ is a pre-extracted matrix of size $|W| \times |W|$ where $|W|$ is the size of domain specific dictionary. $L(cs_{ip}, cs_{jq})$ denotes the prerequisite relationship between concepts $cs_{ip}$ and $cs_{jq}$. $I(i, j)$ represents the order of subchapter $i$ and subchapter $j$. $L$ an $I$ are formally defined as:
Given a concept \( i \), we want its prerequisite concepts to appear before it and its subsequent concepts to appear after it in the extracted concept hierarchy.

Eq. (3.4) can be solved by finding each \( cs_{ip} \) for the \( i^{th} \) subchapter independently, and still enforce some degree of global coherence by adding function \( \Phi \), \( \Theta \) and \( \gamma \) in the optimization function.

### 3.1.2 Concept Hierarchy Extraction from Books

In this section we present our method, CHEB, for solving the optimization problem defined in Eq. (3.4). CHEB combines a local model and a global model which capture three characteristics of an our concept hierarchy: less redundancy, content
consistency and a appropriate learning order. Each function in the equation can be represented as a weighted sum of local and global features which capture chapter-concept or concept-concept pairwise relatedness. For instance, the local relatedness function $\Phi$ is defined as:

$$\Phi(w|tb, cb) = \sum_{i} \omega_i \phi_i(w|tb, cb)$$

where $\phi_i(w|tb, cb)$ is the $i^{th}$ local feature that captures the relatedness between the candidate concept $w$ and the book chapter given its title $tb$ and content $cb$. Details of the local features utilized in CHEB will be introduced in following sections. The coefficient $\omega_i$ is learned using a Support Vector Machine over training data from the constructed data set, described in Section 3.1.3.1.

Similarly, the redundancy function $\Psi$ and consistency function $\Theta$ are defined as the weighted sums of the features which capture the relatedness between two candidates from different chapters and within the same subchapter respectively. The learning order function $\gamma$ defines that whether two are appropriately ordered in the concept hierarchy based on the pre-estimated learning order relationship extracted from Wikipedia.

In general, CHEB is a three-stage method as shown in Figure 3.2. It first extracts a domain-specific dictionary for a given book topic and then performs candidate selection for each chapter. Finally, by re-ranking the candidates based on the local and global features, it generates the concept hierarchy which arrives at coherent
sets of important concepts for a given book.

In the following sections, we will describe three modules of CHEB as suggested in Fig 3.2. We first present a domain-specific concept dictionary construction method using Wikipedia and then introduce our candidate selection method based on title and content similarity. Finally, we discuss our concept hierarchy extraction method and present the details of the proposed local and global features.
3.1.2.1 Domain Specific Concept Identification

The first step of our method is to build a domain-specific dictionary which contains all the possible concepts related to the topic of a book. Specifically, we depth-first search crawl Wikipedia starting from the Wikipedia page of the topic. For instance, Wikipedia page “Macroeconomics” is set as the starting page to perform crawling for a macroeconomics book. For every page visited by our crawler, we extract all the Wikipedia pages that are linked to by anchor texts in the current page and add their titles into the concept dictionary \[87\]. Thus the dictionary is supposed to consist of a set of Wikipedia titles related to a given domain.

During the crawling process, there would be Wikipedia concepts which have low relatedness to the domain being accessed. For instance, “Salt Lake City” would be crawled since it is linked by concept “Packet Switching”. However, this concept is not related to the “Computer Network” domain. Therefore, we perform a filter on the extracted dictionary which removes the unrelated concepts using Wikipedia category information. A category is considered to be a “weak category” if the number of Wikipedia pages in the dictionary which belong to this category is below some threshold. Notice that a concept may belong to multiple categories. If half of its categories are weak categories, the concept will be removed from the dictionary.
3.1.2.2 Candidate Selection

The next step of our method is to select all related candidate concepts for each book chapter and construct a candidate concept hierarchy for the book. It is intuitive that a Wikipedia concept is related to a book chapter if their titles or contents are similar. Therefore, we first define titleMatch as a function measuring the relatedness between a candidate concept and a chapter. Given the book chapter title $tb$ and a Wikipedia candidate title $tw$, if the Wikipedia title is in the book chapter title, $titleMatch(tb, tw) = 1$; Otherwise, $titleMatch(tb, tw) = 0$. For example, for title “Inflation and Interest Rates”, Wikipedia candidates “Inflation” and “Interest Rates” are found and their titleMatch score over the book chapter is 1.

The next measure designed for candidate selection is cosineSim which measures the cosine similarity between the content of Wikipedia candidate and that of the book. Given a chapter, we first match concepts from the dictionary in the chapter content and obtain a list of Wikipedia concepts which appears in the chapter. Then all the anchor texts in these Wikipedia pages and all the concepts in the dictionary are used as a vector space to calculate the normalized term frequency-inverse document frequency (tf-idf) vector for the book subchapter and each Wikipedia candidate. The $cosineSim$ score between $c$ and each candidate is then calculated as the cosine similarity between word vectors.

Our candidate set consists of the top $N$ candidates based on cosineSim score and those candidates whose titleMatch equals 1, i.e., the candidates whose title
appears in the chapter title. These two simple but powerful features are able to capture most of the related and important concepts for each book chapter. They are also used in the relatedness feature set, which will be introduced in the following section.

3.1.2.3 Concept Hierarchy Generation

In this section, we present the details of the local and global features proposed.

3.1.2.3.1 Local Features  In addition to the two features used in the candidate selection, we also make use of the Jaccard distance between the chapter title $tb_i$ and the Wikipedia candidate title $w_i$ as a feature.

\[
\text{Jaccard}(tb_i, w_i) = 1 - \frac{|tb_i \cap w_i|}{|tb_i \cup w_i|}
\]

3.1.2.3.2 Global Features  Global features contain three subsets which correspond to the three characteristics of a concept hierarchy: less redundancy, content consistency and an appropriate learning order.

3.1.2.3.3 Redundancy Features and Consistency Features  In order to resolve the redundancy issue in the concept hierarchy, we reduce the information overlap between the concepts in different chapters, which can be approximated by calculating the pairwise relatedness between the candidate being considered and candidates in different chapters. Similarly, whether a Wikipedia candidate
is “consistent” in this chapter can be approximated by calculating the pairwise relatedness between the candidate being considered and the concepts in the same chapter.

Therefore, for both redundancy and consistency features, it is necessary to capture the relatedness between two Wikipedia candidates. Given two candidates $w_i$ and $w_j$, three relatedness measures are utilized:

- **cosineSim**, which considers the cosine similarity between contents of $w_i$ and $w_j$.
- **Jaccard**, which considers the Jaccard distance between titles of $w_i$ and $w_j$.
- **semSim**, which computes the semantic similarity of a pair of articles from the links they make [115]. Let $L_i$ be the set of Wikipedia concepts which link to $w_i$ and $W_{all}$ be the total number of concepts in Wikipedia, $semSim$ is defined as

$$semSim(w_i, w_j) = 1 - \frac{\max(\log |L_i|, \log |L_j|) - \log |L_i \cap L_j|}{W_{all} - \min(\log |L_i|, \log |L_j|)}$$

The redundancy that a candidate can possibly bring into the concept hierarchy is captured by the following features:

$$cosSimRed(cs_{ip}) = \sum_{\lambda(j) \neq \lambda(i)} \sum_{q=1}^{\min(|cs_j|, K)} \text{cosineSim}(cs_{ip}, cs_{jq})$$

$$JaccardRed(cs_{ip}) = \sum_{\lambda(j) \neq \lambda(i)} \sum_{q=1}^{\min(|cs_j|, K)} \text{Jaccard}(cs_{ip}, cs_{jq})$$

40
\[ \text{semSimRed}(cs_{ip}) = \sum_{\lambda(j) \neq \lambda(i)}^{N} \sum_{q=1}^{\min(|cs_{j}|, K)} \text{semSim}(cs_{ip}, cs_{jq}) \]

where \( K \) is a pre-specified parameter and \( \min(|cs_{j}|, K) \) is number of candidates to be considered in subchapter \( j \) when computing the redundancy. We want to minimize the information overlap for candidates in different concepts. However, if a candidate is not an important concept for a subchapter, it makes no sense to minimize the information overlap between this concept and other candidates in different chapters. Therefore, when calculating redundancy features, we only want to consider those concepts with higher probability of being important candidate concepts. Empirically, we find that the local features are very powerful and the candidates ranked by \( \text{titleMatch} \) and \( \text{cosineSim} \) have relatively high ranking precisions. Therefore, we assume that top-\( K \) candidates have higher probability of being important and only consider these \( K \) concepts.

Consistency features are defined by the following measures:

\[ \text{cosSimCons}(cs_{ip}) = \sum_{q=1}^{\min(|cs_{j}|, K)} \text{cosineSim}(cs_{ip}, cs_{iq}) \]

\[ \text{JaccardCons}(cs_{ip}) = \sum_{q=1}^{\min(|cs_{j}|, K)} \text{Jaccard}(cs_{ip}, cs_{iq}) \]
semSimCons(cs_{ip}) = \sum_{q=1}^{\min(|cs_j|, K)} semSim(cs_{ip}, cs_{iq})

Similarly, we only consider top-$K$ candidates in a subchapter when calculating consistency features.

3.1.2.3.4 Learning Order features  In order to represent the learning order between two Wikipedia candidates $w_i$ and $w_j$, we define $L$ as a $|W| \times |W|$ matrix where $L(w_i, w_j)$ is the learning order relationship between $w_i$ and $w_j$ as suggested in Section 3.1.1.2. At issue is how do we know the prerequisite relationship between two concepts?

Since Wikipedia pages have a relatively uniform format, we try to extract the learning order based on two heuristics. The first sentence of most of the Wikipedia pages, if not all, gives a succinct and general definition for the concept. And the first heuristic used is: given two Wikipedia concepts $w_i$ and $w_j$ and their first sentences $s_i$ and $s_j$, $w_i$ is the prerequisite of $w_j$ if $w_i$ appears in $s_j$. For example, the first sentence of the Wikipedia concept “Hyperinflation is In economics, hyperinflation occurs when a country experiences very high and usually accelerating rates of inflation, rapidly eroding the real value of the local currency, and causing the population to minimize their holdings of the local money.” We thus consider concept inflation a prerequisite of hyperinflation.

Also, most Wikipedia pages have a TOC which links to related concepts. The
second heuristic used is based on the TOC: given two Wikipedia concepts $w_i$ and $w_j$ and their TOC $toc_i$ and $toc_j$, $w_i$ is the prerequisite of $w_j$ if $w_j$ appears in $toc_i$.

For example, the TOC of Wikipedia concept *Money* contains *Money supply* and we thus treat concept *Money* as a prerequisite of *Money supply*. However, this heuristic may have some problems. One is that two Wikipedia concepts can appear in each other’s TOC, such as *Inflation* and *Monetary policy*. It is difficult to figure out which concept we should learn first. For these cases, these two concepts are considered to have no learning order. Since the TOC based rule is not as strong as the definition based rule, it is considered as a complementary of the definition rule, i.e., if the definitions already suggest some learning orders, we will not consider the TOC.

After quantifying the learning order between two concepts, the next step is to capture the global coherence of the concept hierarchy. Given a concept $cs_{ip}$ in subchapter $i$, we hope that all $cs_{ip}$’s prerequisites introduced in the book appear in subchapters before $i$ and all $cs_{ip}$’s subsequent concepts introduced in the book appear in subchapters after $i$. In order to achieve this goal, we define feature $preCorr$ and $subCorr$ to capture the global learning order of the concept hierarchy given the candidate $cs_{ip}$ in the $i^{th}$ subchapter:

$$preCorr(cs_{ip}) = \frac{\sum_{j<i} \sum_{q=1}^{\min(|cs_j|,K)} L(cs_{ip},cs_{jq})=-1}{\sum_{j \neq i}^{N \min(|cs_j|,K)} \sum_{q=1}^{\min(|cs_j|,K)} L(cs_{ip},cs_{jq})=-1}$$
\[
\text{subCorr}(cs_{ip}) = \frac{\sum_{j>i}^{N} \sum_{q=1}^{\min(|cs_j|,K)} 1_{L(cs_{ip},cs_{jq})=1}}{\sum_{j \neq i}^{N} \sum_{q=1}^{\min(|cs_j|,K)} 1_{L(cs_{ip},cs_{jq})=1}}
\]

Similarly, we consider only top-\(K\) candidates in a subchapter when calculating the learning order features. Eq. 3.2.3.2.3 and Eq. 3.1.2.3.4 compute the percentage of concepts that are appropriately ordered based on the prerequisite relationships for \(cs_{ip}\)'s and capture the consistent learning order of a useful concept hierarchy.

### 3.1.2.4 Concept Hierarchy Extractor Training

After generating the features for concept hierarchy extraction, we learn the coefficients for the extractor using SVM\textsuperscript{rank} \[39\] on a data set with manually labelled rankings of Wikipedia candidates for each chapter in three classic textbooks. We use different combinations of features to train our extractor in order to study the importance of different features.

### 3.1.3 Data Sets and Evaluation Metrics

In this section, we first discuss the data preparation for testing \(CHEB\) approach and then introduce the evaluation metrics.

#### 3.1.3.1 Data Preparation and Experiment Setup

We evaluate \(CHEB\) on three high quality textbooks: “Computer networking: a top-down approach featuring the Internet” (hereafter, the computer network
We apply CHEB on these three books to see how it performs on textbooks in different domains.

The general procedure to build test bed for CHEB includes four steps: 1) remove the subchapters with less than 100 words or no important concepts; 2) extract domain specific dictionary for each book; 3) select the top-30 candidates for each subchapter; and 4) manually label the candidates as “important” or “unimportant”.

**Book Subchapter Preprocessing** Besides subchapters with less than 100 words are removed, the “Introduction” and “Conclusion” subchapters which summarize the concepts in other subchapters are also removed.

**Domain Specific Dictionary Construction** To construct our data sets for CHEB, we first perform domain specific dictionary construction as described in Section 3.1.2.1 for each book. Here we use “Computer Network” as the root Wikipedia page for the computer network book, “Macroeconomics” for the macroeconomics book, and “Precalculus” for the precalculus book. A filter is then applied on the dictionary as described in Section 3.1.2.1. If a Wikipedia category contains less than 15 pages in the dictionary, it is considered as a weak category. The number of Wikipedia titles in the dictionary for the three books are: 29689 for the Computer

---


**Candidate Selection** The top-30 Wikipedia candidates are selected using the two features described in Section 3.1.2.2 (*titleMatch* feature and *cosineSim* feature).

**Data Labeling** Based on the extracted candidates, we manually label each Wikipedia candidate as “important” or “unimportant”. For each book, three graduate students with corresponding background knowledge are recruited to label the data. The correlation between the annotators is quite high. For instance, for the computer network book, the three annotators achieve a 79% correlation. This high agreement shows that our manually constructed data set is reliable. Moreover, we use a majority vote to solve the cases where there is not a unanimous agreement.

The books also have different structures including the depth of the TOC, the number of subchapters and the average number of concepts in each subchapter. Table 3.5 provides some statistics for the book structures.

<table>
<thead>
<tr>
<th></th>
<th>Network</th>
<th>Macroeconomics</th>
<th>Precalculus</th>
</tr>
</thead>
<tbody>
<tr>
<td>toc depth</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td># subchapters</td>
<td>50</td>
<td>21</td>
<td>17</td>
</tr>
<tr>
<td>avg # important concepts</td>
<td>3.6</td>
<td>4.5</td>
<td>4.3</td>
</tr>
<tr>
<td>avg # candidate concepts</td>
<td>69.933</td>
<td>80.1071</td>
<td>69.2903</td>
</tr>
<tr>
<td>avg length of title</td>
<td>3.34</td>
<td>6.19</td>
<td>2.53</td>
</tr>
</tbody>
</table>

Table 3.2: Physical characteristics of books
3.1.3.2 Evaluation Metrics

To evaluate the performance of our extractor, we use the metrics \textit{precision@n} and \textit{Mean-Average-Precision(MAP)}. \textit{Precision@n} measures the fraction of the important concepts in \textit{top-n} ranking results. As shown in Table 3.5, most of the book subchapters have less than five important concepts. Therefore, for \textit{Precision@n}, we set \( n = 1, 3, 5 \). We also use Mean Average Precision@10 \textit{MAP@10} to demonstrate an average precision over \textit{top-10} ranking results.

3.1.4 Experiments and Results

We conduct experiments to extract concept hierarchies from books in different domains. Specifically, we test whether the proposed local and global features are effective for identifying important concepts in each subchapter.

We conduct two sets of experiments by comparing our method with baselines. The book-level experiment uses two books as training data and the other book as testing data, and the subchapter-level experiment conducts experiments on three books separately by using part of the subchapters of a book as training data and the remaining chapters as testing data. We finally given a case study on a concept hierarchy extracted from the computer network book.
3.1.4.1 Baseline Method

*SimSeerX* [114] is a similar document search engine and we use the keyphrase method implemented as the baseline model. It receives a whole document as a query, performs automatic information extraction on the document, and then uses several similarity functions to identify and rank similar documents in an indexed collection. *SimSeerX* has been designed in order to work with multiple document collections and offer multiple similarity functions. It currently supports similarity functions based on keyphrases [63], sequences of terms, and overall word similarity in documents. *SimSeerX* provides a generic architecture for similarity search and has been used with several document collections, such as the *CiteSeerX* collection, Wikipedia dataset and a plagiarism dataset in which it was the best plagiarism detector. In this study, we use the keyphrase similarity function in *SimSeerX* as a baseline.

When using keyphrase similarity in *SimSeerX*, two documents are considered potentially similar if they share at least one automatically extracted keyphrase or if the keyphrase exists in the text. For each document indexed by *SimSeerX*, keyphrases are automatically extracted using the *Maui tool* [63]. *Maui* begins by identifying candidate keyphrases in the text based on n-grams of words, and then features [63] for each word are inputs to a machine learning model and with the output the probability that the candidate keyphrase is a keyphrase. *SimSeerX* indexes the top-10 keyphrases identified by *Maui*. At query time, the top-10 queries
are extracted from the query document using the same procedure and indexed documents with at least one matching keyphrase are retrieved. This candidate set of results is then ranked based on the full text cosine similarity of each document with the query document. For what we believe to be a fair comparison, we remove the concepts which are not in the domain specific dictionary from the ranking results of the baseline method.

3.1.4.2 Book-Level Concept Hierarchy Extraction

In this section, a book-level concept hierarchy extraction is first performed by using two of the books as training data and the third one as testing data. Table 3.4 shows the ranking precisions on computer network book, macroeconomics book and precalculus book respectively. As shown, we test different combinations of features, with the local features derived from different aspects of relatedness between book subchapter and Wikipedia candidates, and global features which consider the global coherence of the book structure. The results show that incorporating our proposed local and global features into the extractor does achieve significantly higher precision than the baseline model. Recall that different books vary significantly in terms of their structure and the number of important concepts in each subchapter, but our results appear robust across all of them.

From the experimental results, we see that local features, namely titleMatch, cosineSim and Jaccard, are effective in the concept hierarchy extraction. However,
the *titleMatch* feature is not very robust because its usefulness depends on type of book title. A title that is an analogy or has too little information can make this not very useful. For instance, the title of subchapter 1.1.1 of the computer network book is *A Nuts-and-Bolts Description*, making it difficult for the *titleMatch* feature to obtain meaningful information. Moreover, information contained in the title is usually limited and thus the *titleMatch* feature has a very low recall which leads to the low *MAP@10* score. As shown in the results, the *titleMatch* feature has the lowest *MAP@10* score among all the features on three books. The feature achieves a best *MAP@10* score of 0.12 on the macroeconomics book, which has the largest average number of words in the title as shown in Table 3.5.

Incorporating global features does achieve better results for the computer network book and the macroeconomics book, but not so on the precalculus book. A potential reason for this is that the precalculus book is an entry-level book and splits each concept into more than one subchapter in order to present more details. For instance, Chapters 2,3,4,5, and 6 all discuss functions. Therefore, it is hard for CHEB to capture the book structure and thus the global features.

### 3.1.4.3 Subchapter-Level Concept Hierarchy Extraction

From the numbers in Table 3.5 we see the book structures are quite different. In order to capture their different structures, we also conduct experiments on each book separately.
Table 3.3: Book-level experimental results

Parts of the subchapters are used as training data to train the extractor and the remaining subchapters are used as testing data using 5-fold cross validation. From the results, we observe that our model outperforms the baseline method for the overall performance on three books and the results obtained using different feature sets are consistent with the findings in the overall experimental results (See Table 3.4).

Although the gains in the global features are marginal, global features are
especially helpful in predicting the top-1 important concept. As we can observe, adding global features improves precision@1 from 0.79 to 0.84 for the computer network book, 0.8 to 0.83 for the macroeconomics book, and 0.80 to 0.83 for the precalculus book.

<table>
<thead>
<tr>
<th></th>
<th>P@1</th>
<th>P@3</th>
<th>P@5</th>
<th>MAP@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline Method</td>
<td>0.29</td>
<td>0.23</td>
<td>0.18</td>
<td>0.17</td>
</tr>
<tr>
<td>TitleMatch Feature</td>
<td>0.42</td>
<td>0.19</td>
<td>0.16</td>
<td>0.21</td>
</tr>
<tr>
<td>CosineSim Feature</td>
<td>0.74</td>
<td>0.48</td>
<td>0.4</td>
<td>0.35</td>
</tr>
<tr>
<td>Local Features</td>
<td>0.79</td>
<td>0.52</td>
<td>0.43</td>
<td>0.34</td>
</tr>
<tr>
<td>Global Features</td>
<td>0.38</td>
<td>0.34</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>Local+Global Features</td>
<td>0.84</td>
<td>0.52</td>
<td>0.42</td>
<td>0.37</td>
</tr>
</tbody>
</table>

(a) Subchapter-level experimental results on network book

<table>
<thead>
<tr>
<th></th>
<th>P@1</th>
<th>P@3</th>
<th>P@5</th>
<th>MAP@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline Method</td>
<td>0.4</td>
<td>0.33</td>
<td>0.24</td>
<td>0.23</td>
</tr>
<tr>
<td>TitleMatch Feature</td>
<td>0.5</td>
<td>0.37</td>
<td>0.24</td>
<td>0.27</td>
</tr>
<tr>
<td>CosineSim Feature</td>
<td>0.57</td>
<td>0.61</td>
<td>0.46</td>
<td>0.33</td>
</tr>
<tr>
<td>Local Features</td>
<td>0.8</td>
<td>0.52</td>
<td>0.44</td>
<td>0.32</td>
</tr>
<tr>
<td>Global Features</td>
<td>0.5</td>
<td>0.35</td>
<td>0.32</td>
<td>0.29</td>
</tr>
<tr>
<td>Local+Global Features</td>
<td>0.83</td>
<td>0.54</td>
<td>0.42</td>
<td>0.34</td>
</tr>
</tbody>
</table>

(b) Subchapter-level experimental results on macroeconomics book

<table>
<thead>
<tr>
<th></th>
<th>P@1</th>
<th>P@3</th>
<th>P@5</th>
<th>MAP@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline Method</td>
<td>0.36</td>
<td>0.13</td>
<td>0.08</td>
<td>0.08</td>
</tr>
<tr>
<td>TitleMatch Feature</td>
<td>0.41</td>
<td>0.15</td>
<td>0.09</td>
<td>0.07</td>
</tr>
<tr>
<td>CosineSim Feature</td>
<td>0.6</td>
<td>0.51</td>
<td>0.41</td>
<td>0.32</td>
</tr>
<tr>
<td>Local Features</td>
<td>0.8</td>
<td>0.49</td>
<td>0.4</td>
<td>0.34</td>
</tr>
<tr>
<td>Global Features</td>
<td>0.43</td>
<td>0.3</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>Local+Global Features</td>
<td>0.83</td>
<td>0.46</td>
<td>0.39</td>
<td>0.34</td>
</tr>
</tbody>
</table>

(c) Subchapter-level experimental results on precalculus book

Table 3.4: Subchapter-level experimental results
3.2 Prerequisite Concept Maps Extraction from Textbooks


A knowledge graph organizes knowledge by linking entities with their relationships and is applicable to many NLP tasks such as question answering [54] and knowledge acquisition [31]. While recent work has addressed reasoning in knowledge graphs (DBpedia [1] and YAGO [101]), and for real world facts [85], there has been little effort in organizing knowledge for educational purposes. Applications of such knowledge structures in education have been widely used in teaching and learning assessment [61].

There are many interesting challenges in extracting knowledge graphs for education. In some cases, nodes in an educational knowledge graph can be scientific and mathematical concepts, such as “Lasso” and “Regularization”, instead of typical entities such as individuals, locations, or organizations. As such, instead of using general concept relationships such as “is-a” and “part-of”, we focus on the prerequisite dependencies among concepts. A prerequisite dependency requires that learning one concept is necessary before learning the next. For instance, we
need to have basic knowledge of “Regularization” in order to learn “Lasso” (or L1-regularized regression).

We present a method for constructing a specific type of knowledge graph, a concept map, which is widely used in the learning sciences \[113\]. In such a directed graph, each node is a scientific concept and directed links between these nodes imply their prerequisite dependencies. Figure [3.3] shows an example of an extracted concept map in the economics area where each node is an economical concept such as “Gross domestic product” and “Consumer price index” and links indicate prerequisite dependencies relating these concepts (from prerequisites to subsequent concepts).

![Chapter 1: Macroeconomics](Image)

![Chapter 2: The Data of Macroeconomics](Image)

**Chapter 1: Macroeconomics**

**Chapter 2: The Data of Macroeconomics**

2.1 Measuring the value of economic activity: gross domestic product
2.2 Measuring the cost of living: the consumer price index
2.3 Measuring joblessness: the unemployment rate
...

Figure 3.3: Example of an extracted concept map in economics.

Traditional approaches to knowledge graph extraction generally consist of two separate steps: 1) Extracting key concepts and, 2) Identifying relationships between key concepts. While these two common information extraction tasks have been well
studied [3,22,101], solving these two tasks independently for educational content poses problems. We argue that these two problems are actually strongly coupled, meaning that the results of one affects the results of the other. Thus, solving these sub-problems independently might lead to sub-optimal performance. For example, in educational resources, a concept is often presented by first introducing its prerequisites. Thus the order in which two concepts appear in a document source can help identify their prerequisite relation. If a concept in this ordered chain is not correctly extracted, its prerequisite relation to other concepts will be lost. Furthermore, if this concept is the prerequisite to many others, we may no longer identify an important key concept.

Leveraging information from existing educational resources, we propose a concept map extraction model that jointly optimizes these two sub-problems. It then utilizes identified prerequisites to refine the extracted key concepts and vice versa. This model produces a related key concept set with prerequisite relations among those concepts.

There are many education resources from which one could build concept maps. For this work, we focus on textbooks since they often provide a comprehensive list of domain concepts and are often used as major educational resources in schools, colleges and universities. Educational resources such as textbooks and slides can provide implicit knowledge structures for knowledge graph extraction. For example, structural information such as table of contents (TOC) of a textbook can be very
useful in identifying concept relationships. We feel this method could be easily
generalized to other education resources with structured information such as slides
and courses. We then augment “inside-the-book” knowledge with web content (for
now, Wikipedia), thus enriching the content of a specific book with complementary
information. As described in Section 3.2.3.4 we will empirically verify that using
such complementary resources can give quality concept information at the secondary
school and undergraduate level. In summary, our contributions are:

- The first attempt, to the best of our knowledge, to use textbooks to extract
  concept maps with explicit prerequisite relationships among the concepts.
- A set of principled methods that utilize both Web knowledge (Wikipedia) and
  the rich structure in textbooks to identify prerequisite relationships among domain
  concepts.
- An optimization framework that jointly solves the two sub-problems of concept
  map extraction and linkage.
- The generation of datasets from books in six different educational domains to
  show how our methods work.

### 3.2.1 Problem Definition

We define the input and output of CMEB task as:

**Input:** The input to a CMEB framework consists of two components:

- A book $B$ with a list of titles $TB = \{tb_1, tb_2, ..., tb_K\}$, where $K$ is the number
of subchapters in the book, and contents $CB = \{cb_1, cb_2, ..., cb_k\}$. $th_i$ and $cb_i$ are the title and the content for the $i^{th}$ subchapter in the TOC respectively, and $K$ is the total number of subchapters in the book.

- A set of Wikipedia concepts $W = \{w_1, w_2, ..., w_{|W|}\}$ with a list of titles $TW = \{tw_1, tw_2, ..., tw_K\}$ and contents $CW = \{cw_1, cw_2, .., cw_k\}$.

**Output:** CMEB aims to extract a directed graph $G = (V, E)$ from $B$ where $v_i \in V$ is Wikipedia concept which serves as a key concept in at least one subchapter in $B$, and $E_{ij}$ is the prerequisite relationship between concept $i$ and $j$. A key concept is a concept that relates to the main point of a subchapter, with its definition, explanation, or examples usually being presented in the subchapter.

$$E(w_i, w_j) = \begin{cases} 
1 & \text{if } w_i \text{ is a prerequisite of } w_j \\
-1 & \text{if } w_j \text{ is a prerequisite of } w_i \\
0 & \text{There is no prerequisite} \\
\text{relationship between } w_i \text{ and } w_j 
\end{cases}$$  \hspace{1cm} (3.7)

A concept may appear as a key concept in multiple subchapters in a textbook. In this work, we use majority voting to decide the prerequisite relationship when the same pair of concepts have different relationship directions in different subchapters.
3.2.2 Joint Knowledge Graph Extraction from Textbooks

Here, we introduce our notation and describe how we jointly extract key concepts as well as the prerequisite relations. We define \( c \in C \) as a concept where \( C \) is a set of Wikipedia concepts, \( s \in S \) for a subchapter in the textbook. The term “subchapter” refers to all the headings in the TOC. For instance, both 1.1 and 1.1.1 are subchapters. A key concept in a book chapter is a concept which is not only mentioned but also discussed and studied in the subchapter. The input to our extractor consists of a digital book \( B \) with a list of titles, chapter number and contents for all its chapters. Each chapter contains one or more key concepts. The output is a concept map \( G \), which is represented as a set of triples in the form \( \{ (c_1, c_2, r) | c_1, c_2 \in C, r \in R \} \), where \( R = \{0, 1\} \) is the prerequisite relationship, \( r \) takes value 0 when \( c_1 \) and \( c_2 \) have no prerequisite relation; and takes value 1 when \( c_1 \) is \( c_2 \)’s prerequisite. We use \( CS = \{ cs_{ip} \in \{0, 1\} | 1 \leq i \leq |C|, 1 \leq p \leq |S| \} \), to indicate concept appearance in subchapter, where \( cs_{ip} \) takes value 1 when the \( i^{th} \) concept is a key concept in the \( p^{th} \) subchapter; otherwise takes 0. Our goal is to optimize \( CS \) and \( R \) in order to obtain a global concept map.

3.2.2.1 Concept Map Extraction

Key Concept Extraction Intuitively, if concept \( c \) is a key concept in subchapter \( s \), it should have these few properties: 1). Local Relatedness: Key concept \( c \) should be strongly related to subchapter \( s \). For instance, the concept and book
chapter share similar topics; 2). Global Coherence: We argue that extracted key concepts should be coherent in the following way: Less redundancy: Chapters do not always discuss all of the same concepts. Information overlap between concepts in different chapters should be minimized. For instance, given a geometry textbook, if subchapter 2.1 covers “Triangle” in detail, subchapter 3.1 should not cover this concept in detail again.

Note that here we mention both concept-concept relatedness and concept-chapter relatedness. We denote all relatedness as one symmetric similarity function \( f(\cdot, \cdot) \), which can take both the concept and chapter as arguments. We will discuss the definition of \( f(\cdot, \cdot) \) later in this section. Given \( f(\cdot, \cdot) \), the following objective function is proposed to derive the concept-subchapter matrix \( CS \) from the aforementioned properties:

\[
P_1(CS) = \alpha_1 \sum_{i=1}^{\mid C \mid} \sum_{p=1}^{\mid S \mid} cs_{ip} f(c_i, s_p) + \alpha_2 \sum_{i,j=1}^{\mid C \mid} \sum_{p \neq q} cs_{ip} cs_{jq} f(c_i, c_j),
\]

(3.8)

(3.9)

where \( I(\cdot) \in \{1, -1\} \) is an indicator function and returns 1 if the statement holds and returns \(-1\) otherwise. \( \alpha_s \) are the term weights.

The first term corresponds to the local relatedness attributes and captures the relatedness between candidates and book chapter. This term should be maximized
to select candidates similar to the book chapter. The second term is used to reduce redundancy in the concept map. For this term, we calculate the pairwise similarity between selected concepts in different chapters as the redundancy in the extracted concept map and this value should be minimized.

**Prerequisite Relationship** We consider a pair of concepts to have a prerequisite relationship if they are: 3). **Topically Related**: If two concepts cover different topics, it is unlikely that they have prerequisite relationships. 4). **Complexity Level Difference**: Not all pairs of concepts with similar topics have prerequisite relationships. For example, “isosceles triangle” and “right angled triangle” cover similar topics but do not have learning dependencies. Thus, given two concepts, it is necessary to identify whether one concept is basic while another one is advanced.

We denote complexity level of a concept as $l(\cdot)$ and later discuss the definition of $l(\cdot)$. Given $l(\cdot)$, we define the following optimization function for $R$:

$$P_2(R) = \alpha_3 \sum_{i,j=1,i\neq j}^{|C|} r_{ij} f(c_i, c_j) + \alpha_4 \sum_{i,j=1}^{|C|} r_{ij}(l(c_i) - l(c_j)).$$ \hspace{1cm} (3.10)

The first term corresponds to the *Topically Related* attributes and should be maximized. The second term is used to measure the *Complexity Level Difference* between two concepts and we want this value to be maximized.
3.2.2.1 Joint Modeling  To reinforce the mutual benefit between two sub-problems, we propose 5). Order coherence: Concepts should not be discussed without introducing their prerequisites, i.e., given a concept, prerequisite concepts should be introduced before this concept and subsequent concepts should be introduced after the concept. The following function is proposed to derive this mutual benefit property:

\[ P_3(CS, R) = \alpha_5 \sum_{i,j=1}^{\left|\mathcal{C}\right|} \sum_{p,q=1}^{\left|\mathcal{S}\right|} I(p < q) c_{ij} p c_{ij} q r_{ij}. \] (3.11)

In summary, the global objective function \( \Lambda(CS, R) = P_1(CS) + P_2(R) + P_3(CS, R) + \beta_1 \|CS\| + \beta_2 \|R\| \) consists of \( P_1 \) for key concept extraction, \( P_2 \) for prerequisite relationship extraction, \( P_3 \) for mutual benefit modeling and \( L_1 \) regularization terms to control model complexity and is maximized. Figure 3.4 shows the system workflow for concept map extraction with joint modeling.

3.2.2.1.2 Optimization  We maximize \( \Lambda \) to obtain the optimal concept map by adopting the Metropolis-Hasting algorithm to optimize \( CS \) and \( R \) respectively. \( \forall cs \in CS \), we calculate the value of \( \Lambda \) using current value of \( cs \) and the flipped value of \( cs \) (denoted as \( cs' \)). We follow the following update rule to update \( CS \).
Figure 3.4: System workflow for concept map extraction with joint modeling.

\[
\sigma_{CS}(cs, cs') = \begin{cases} 
1, & \text{if } \Lambda(R^{(n)}, CS^{(n)}, cs') \leq \Lambda(R^{(n)}, CS^{(n)}, cs) \\
\exp(-\beta(\Lambda(R^{(n)}, CS^{(n)}, cs') - \Lambda(R^{(n)}, CS^{(n)}, cs))), & \text{otherwise.}
\end{cases}
\]

Similarly, \( \forall r \in R \), we perform updates according to the following update rule:

\[
\sigma_{R}(r, r') = \begin{cases} 
1, & \text{if } \Lambda(R^{(n)}, CS^{(n)}, r') \leq \Lambda(R^{(n)}, CS^{(n)}, r) \\
\exp(-\beta(\Lambda(R^{(n)}, CS^{(n)}, r') - \Lambda(R^{(n)}, CS^{(n)}, r))), & \text{otherwise.}
\end{cases}
\]
3.2.2.2 Representation Schemes

We explore different schema for book chapter/concept content representation and then derive measures for concept/book chapter similarity $f(\cdot, \cdot)$ and the concept complexity level $l(\cdot)$. If multiple measures are derived for the same attribute, we adopt an equal weighted sum of different measures as the value of this attribute.

3.2.2.2.1 Word Based Similarity  
We represent each chapter using words appearing in the chapter and each concept using a bag-of-word representation from the word content in their Wikipedia pages. Standard text preprocessing/weight procedures, including case-folding, stop-word removal and term frequency-inverse document frequency(tf-idf) are applied. Based on this representation, we define the concept-chapter similarity function $f(\cdot, \cdot)$ (applied in Equation 1) as a combination of the following measures:

- **Title match**: This feature measures the relatedness between the concept tile and the chapter/concept title. Given a book chapter/concept title $tb$ and a Wikipedia candidate title $tw$, if $tw$ is in $tb$ or $tw$ is $tb$, $Titlematch(tb, tw) = 1$; Otherwise, $Titlematch(tb, tw) = 0$.

- **Content cosine similarity**: This feature measures the cosine similarity between the word TF-IDF vectors of chapter/concept contents.

- **Title Jaccard similarity**: This feature computes the Jaccard similarity between the chapter/concept title.
• **Sustained periods in subchapters:** A sustained period of a concept in a subchapter is the period from its first appearance to its last appearance. When the sustained period of a candidate concept in a subchapter is longer, it is more likely that this concept is important in this chapter.

We introduce one additional measure for concept-concept similarity. This concept-concept measure together with the other four aforementioned measures are used for concept-concept similarity and applied in Equation 2, and the first term in Equation 3.10.

• **Concept co-occurrences:** by counting the co-occurrences of two concepts lies within a sentence from either a book chapters or a Wikipedia page.

We also derive the following measures for a concept’s complexity level based on its Wikipedia anchors. These measures are used in the second term in Equation 3.10.

• **Supportive relationship in concept definition:** A is likely to be B’s prerequisite if A is used in B’s definition. Here, we use the first sentence in the concept’s Wikipedia page as its definition. \(\text{Supportive}(A, B) = 1\) if A appears in B’s definition. For instance, “Logarithm” is used to define “Natural logarithm” whose definition is “The natural logarithm of a number is its logarithm to the base e... ” and \(\text{Supportive}(\text{logarithm}, \text{natural logarithm}) = 1\).

### 3.2.2.2 Word Embedding

This method maps concepts from the vocabulary to vectors of real numbers in a low-dimensional space \(\text{[67]}\). We use word2vec which discovers lower dimensional vectors with two-layer neural networks using the
contexts and syntactic of concepts. The concept similarity is defined as the cosine similarity of two concepts’ embeddings, which is used in Equation 2 and the first term in Equation 3.10.

3.2.2.2.3 Wikipedia Anchors Besides the content information, millions of cross-page links in Wikipedia are also useful in detecting concept relatedness and concept complexity levels. Given two concepts, we calculate the following measures as their similarity and use these measures in Equation 2, and the first term in Equation 3.10.

- **Wikipedia link based Jaccard similarity:** Given two concept, this feature computes the Jaccard similarity of the in-links/out-links of their Wikipedia page.

- **Wikipedia link based semantic similarity:** This feature computes the semantic relatedness of two concepts based on their Wikipedia links [115].

\[
1 - \frac{\max(\log |Q_i|, \log |Q_j|) - \log |Q_i \cap Q_j|)}{\log W_{all} - \min(\log |Q_i|, \log |Q_j|)}
\]

where \(Q_i\) is the set of Wikipedia concepts which link to \(w_i\) and \(W_{all}\) be the total number of concepts in Wikipedia.

We also derive the following measures for a Wikipedia concept’s complexity level and use these three measures in the second term in Equation 3.10.

- **Number of in-links/out-links:** This feature returns the number of in-links/out-links in the concept Wikipedia page.
- Relational strength in textbook/Wikipedia: Relational strength $RS(w_i, w_j)$ measures the semantic relatedness between two concepts using concept co-occurrence and distance in the same sentence [12]:

$$
\log(\frac{n_{ij}/\max(n)}{\text{avg}_d^2_{ij}/\max(\text{avg}_d^2)}), \ i \neq j,
$$

where $n_{ij}$ is the co-occurrence of concept $i$ and $j$ within a sentence and $\text{avg}_d^2_{ij}$ is sum of the distance of two keywords divided by the number of times two keywords appeared in the same sentence. If two concepts are close within a lot of sentence in articles, implies that their relationship is also stronger than the others.

$$
\text{avg}_d^2_{ij} = \frac{n_{ij}}{\sum_{m=1}^{||W||} d_m^2}, \ i \neq j,
$$

- RefD: [55] defines a new metrics measuring the prerequisite relationships between concepts using Wikipedia links between two concepts. If most related concepts of $A$ refer to $B$ but few related concepts of $B$ refer to $A$, then $B$ is more likely to be a prerequisite of $A$.

$$
\frac{\sum_{i=1}^{||W||} v(w_i, B) \cdot u(w_i, A)}{\sum_{i=1}^{||W||} w(w_i, A)} - \frac{\sum_{i=1}^{||W||} v(w_i, A) \cdot u(w_i, B)}{\sum_{i=1}^{||W||} w(w_i, B)},
$$

where $W = \{w_1, ..., w_{|W|}\}$ is the concept space and $|W|$ is the size of the concept space; $u(w_i, A)$ weights the importance of $w_i$ to $A$; and $v(w_i, A)$ is an indicator.
showing whether \( w_i \) has a Wikipedia link to \( A \).

### 3.2.2.2.4 Textbook Structure

The TOC of textbooks contains implicit prerequisite relationships between concepts since textbooks usually introduce concepts based on their learning dependencies. Therefore, we define TOC distance between two concepts as the distance between their subchapter numbers. This feature is used to measure complexity level difference between concepts and applied in the second term in Equation 3.10.

Given two concepts \( A \) and \( B \), \( a_i \) and \( b_i \) are used to denote their chapter number arrays. For example if \( A \) is in chapter 1.1, then \( a_1 = 1 \) and \( a_2 = 1 \). We define the TOC distance in textbooks between \( A \) and \( B \) as:

\[
\text{TOCdistance}(a, b) = \frac{(a_i - b_i)}{\beta^{i-1}}
\]

where \( i \) is the smallest index such as \( a_i \neq b_i \) and \( \beta \) is a pre-specified decay parameter which is empirically set as 2. For instance, given a concept “HTTP” from chapter 2.3.1 and “HTTP message body” from chapter 2.3.2, TOC distance between them is 0.25 and “HTTP” could be “HTTP message body”’s prerequisite. Notice that a concept can serve as the key concept in multiple chapters and the value of the TOC distance feature between two concepts is the average TOC distance of all pairs of TOC of these two concepts. This measure is used in second term in the second term in Equation 3.10.
3.2.3 Experiment Settings

3.2.3.1 Dataset

In order to build a test bed for concept map extraction, we manually construct concept maps using six widely-used textbooks: computer networking\(^4\), macroeconomics\(^5\), precalculus\(^6\), databases\(^7\), physics\(^8\), and geometry\(^9\).

To construct the final dataset, we first manually label key concepts: 1) Extract all Wikipedia concepts that appear in each book chapter. 2) Given a candidate concept \(c_i\) with title \(tw\), we select it as a key candidate of subchapter \(j\) if \(\text{Titlematch}(tw, tb_j) = 1\) where \(tb_j\) is the title of the subchapter \(j\), or \(c_i\) is ranked within \(top - 30\) among all candidates in subchapter \(j\) based on \(\text{Content cosine similarity}\) feature. 3) Label the candidates as “key concept” or “not key concept” and obtain a set of key concepts for this area. Then for each pair of key concepts \(A\) and \(B\), we manually label them as “A is B’s prerequisite”, “B is A’s prerequisite” or “No prerequisite relationship”. Table 3.5 shows characteristics of the dataset. For each area, three graduate students with corresponding background knowledge are recruited to label the data and we take a majority vote of the annotators to create

---

9Dan Greenberg, Lori Jordan, Andrew Gloag, Victor Cifarelli, Jim Sconyers, Bill Zahnerm, 'CK-12 Basic Geometry'

68
final labels. We achieve an average 79% correlation for the key concept labeling task and an average 83% correlation for the concept relationship labeling task.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Network</th>
<th>Economics</th>
<th>Precalculus</th>
<th>Geometry</th>
<th>Database</th>
<th>Physics</th>
</tr>
</thead>
<tbody>
<tr>
<td># subchapter</td>
<td>98</td>
<td>37</td>
<td>23</td>
<td>48</td>
<td>90</td>
<td>152</td>
</tr>
<tr>
<td># Key concepts per chapter</td>
<td>3.72</td>
<td>4.54</td>
<td>3.99</td>
<td>3.16</td>
<td>1.41</td>
<td>3.54</td>
</tr>
<tr>
<td>Candidate concepts per subchapter</td>
<td>70.53</td>
<td>84.16</td>
<td>70.21</td>
<td>55.89</td>
<td>49.17</td>
<td>68.94</td>
</tr>
<tr>
<td># labeled pairs</td>
<td>1500</td>
<td>877</td>
<td>1171</td>
<td>1305</td>
<td>529</td>
<td>1517</td>
</tr>
<tr>
<td># pairs with relationships</td>
<td>257</td>
<td>157</td>
<td>222</td>
<td>186</td>
<td>96</td>
<td>208</td>
</tr>
</tbody>
</table>

Table 3.5: Physical characteristics of books. # labeled pairs is the number of candidate concept pairs labeled as whether two concepts have prerequisite relationships. # pairs with relationships is the number of concept pairs with prerequisite relationships in all the labeled pairs.

3.2.3.2 Baseline - Key Concept Extraction

3.2.3.2.1 TextRank TextRank is a method widely used in key sentence and keyphrase extraction [66]. The general procedure of text rank is to build up a graph using candidate key concepts as vertices and co-occurrence of two candidates within a sentence as the weight on the edge between them. Then the algorithm iterates over the graph until it converges and sorts vertices based on their final scores to identify key concepts.

3.2.3.2.2 Wikify Wikify detects significant Wikipedia concepts within unstructured texts. We use Wikipedia Miner developed in [69] to link book contents with Wikipedia concepts.

3.2.3.2.3 Supervised Key Concept Extraction (Supervised KCE): Based on the local relatedness and global coherence attributes proposed in Section 3.2.2.2.
we propose the following features for key concept learning from each subchapter.

**Local Features:** We use features defined in Section 3.2.2.2 which capture the relatedness between concepts and book subchapters, i.e., *Title Match, Content cosine similarity, Title Jaccard similarity, Sustained periods in subchapters* of the concept, are used as local features in concept extraction.

**Global Features:** Global features include two sub-set of features: redundancy features and order coherence features.

*Redundancy Features:* This set of features measure the information overlap that a candidate $c_i$ can possibly bring into the extracted concept set. Given the $i^{th}$ candidate in $j^{th}$ chapter, we calculate the similarity between this candidate and other candidates in different subchapters as the value of the redundancy feature of this candidate:

$$Red(c_i) = \sum_{i,j=1}^{|C|} \sum_{p \neq q} cs_{ip}cs_{jq}f(c_i, c_j),$$

where $f(cs_{ki}, cs_{pj})$ is the similarity between candidate $cs_{ki}$ and $cs_{pj}$ and where $I(\cdot) \in \{1, -1\}$ is an indicator function and returns 1 if the statement holds and returns $-1$ otherwise.

Section 3.2.2.2 defines different semantic relatedness measurements and all these measurements can be applied to calculate redundancy features.

*Order Coherence Features:* Besides less redundancy attributes, we also expect consistent learning order in concepts extracted from the book, i.e., given a concept $cs_{ki}$ in subchapter $i$, we expect that all $cs_{ki}$’s prerequisites appear in subchapters
before $i$ and all $cs_{ki}$’s subsequent concepts appear in subchapters after $i$. Given candidate $cs_{ki}$ in the $k^{th}$ subchapter, we define features $orderCorr$ to capture the global learning order of the extracted concepts:

$$orderCorr(c_i) = \frac{\sum_{j=1}^{C} \sum_{p,q=1}^{S} |I(p < q)cs_{ip}cs_{jq}r_{ij}|}{\sum_{j=1}^{C} \sum_{p,q=1}^{S} cs_{ij}cs_{jq}|r_{ij}|}.$$ 

Equation 3.2.3.2.3 computes the percentage of concepts that are appropriately ordered based on the $c_i$’s prerequisite relationships.

We use $SVM_{rank}$ to predict rankings of Wikipedia candidates for each subchapter with data from one book as testing data and data from other five as training data.

### 3.2.3.3 Baseline - Prerequisite Relationship Identification

**3.2.3.3.1 Hyponym-Hypernym** A hyponym is a concept whose semantic field is included within that of another concept (hypernym) and in this work, we use hyponym-Hypernym to as a baseline method of deriving prerequisite relationships. Lexico-syntax pattern based extraction methods are popular methods for extracting hyponym relationships between concepts because they offer effective text processing text. We adopt the 10 lexico-syntactic patterns selected for hyponymy-hypernymy pattern matching in [112], as shown in Table 3.6.

**3.2.3.3.2 Supervised Relationship Identification (Supervised RI)** For concept relationship extraction, we utilize Topically Relatedness Features and
Table 3.6: Extracted Lexico-syntax patterns. NP1 represents a subsequent Noun Phrase (NP) and NP2 represents a prerequisite Noun Phrase (NP).

| NP2 such as NP1 | NP1, one of NP2 |
| NP1, one of NP2 as NP1 | NP1 (and|or) other NP2 |
| NP1 is (a|an) NP2 | NP2 consist(s) of NP1 |
| NP2 includ(s|es|ing) NP1 | NP2 (like|, specially) NP1 |
| NP1 (is|are) called NP2 | NP1 (in|belong to) NP2 |

Complexity Level Difference Features introduced in Section 3.2.2.2 to identify concept prerequisite relationship. Topically Relatedness measures include Title match, Content cosine similarity, Title Jaccard similarity, Wikipedia link based Jaccard similarity, Wikipedia link based semantic similarity, Relational strength in textbook/Wikipedia. Complexity Level Difference features include Supportive relationship in concept definition, RefD, Number of in-links/out-links, TOC Distance.

Then we perform a binary class classification using SVM to identify prerequisite relationships with five books as training data and one book as testing data.

3.2.3.4 Wikipedia Coverage of Concepts

Wikipedia has previously been utilized as a controlled vocabulary for topic indexing [63,64] and key phrase extraction [73]. A few studies have examined Wikipedia coverage of academically related topics [78,93–95]. Though some work showed that Wikipedia does not properly cover academic content on the front end of science, previous studies [78,95] have demonstrated that Wikipedia’s coverage of topics is comprehensive for secondary school and undergraduate education.

In order to further validate the coverage of the extracted concept maps, we
conducted the following experiments. For each book, three graduate students with corresponding background knowledge are recruited to manually extract all concepts from each subchapter (randomly sampled from the book), and label whether these concepts have a corresponding Wikipedia page. We found that 88% of the concepts (Computer network: 85%, Macroeconomics: 86%, Precalculus: 91%, Geometry: 97%, Physics: 85%, Database: 89%) in the books are covered by Wikipedia and this provides some empirical evidence of reasonable coverage of the extracted concept maps.

### 3.2.3.5 Parameter Selection

As shown in Equation 3.9 and Equation 3.10, our concept maps are shaped by parameters $\alpha = \{\alpha_i, i = 1, 2, 3, 4\}$ and $\beta = \{\beta_j, j = 1, 2\}$ where $\alpha$s are the term weight in the optimization function and $\beta$s are the weight of L1-regularization. We test different methods in a “leave one book out” manner, i.e., when testing on one book, we train our model using the other 5 books to select the optimal combination of parameters.

### 3.2.3.6 Model Initialization

- **Concept-Subchapter Matrix Initialization:** To initialize $CS(\cdot)$, we use two features Title match and Content cosine similarity proposed in Section 3.2.2.2 which measure the local similarity between a candidate and a book chapter. We set $cs_{ij} = 1,$
i.e., candidate $c_i$ is a key concept in subchapter $j$, if $Titlematch(c_i, tb_j) = 1$ where $tb_j$ is the title of the subchapter $j$, or $c_i$ is ranked within $top - 5$ based on cosine similarity between chapter/concept contents feature.

- **Concept Relationship Matrix Initialization:** To initialize the concept relationship matrix $R(\cdot)$, given two concepts $c_i$ and $c_j$, we set $r_{ij} = 1$ if their complexity level difference is higher than threshold $t_1$ and topically relatedness is higher than threshold $t_2$. Empirically, $t_1$ is set as mean value of the overall complexity level difference and $t_2$ as mean value of the overall topically relatedness.

### 3.2.4 Experimental Results

#### 3.2.4.1 Effect of Textbook Information

In this section, we present how textbook structures help concept map extraction.

Figure 3.5 shows ranking precisions of key concept extraction on six books. For the baseline methods presented, we needed to manually decide the number of key concepts in each subchapter. We thus present the performance of $top - 1$, $top - 3$, and $top - 5$ candidates from the concept extraction phase respectively. As shown, we test different combinations of features, with the local features derived from different aspects of relatedness between book subchapter and Wikipedia candidates, and global features which consider the global coherence of the book structure. The results show that incorporating our proposed global features (See “Supervised KCE” in Figure 3.5) into the extractor does achieve significantly higher precision than
other methods which do not consider book structure (TextRank, Wikify and Local features).

![Bar charts for key concept extraction from six textbooks: Computer network, Macroeconomics, Precalculus, Geometry, Database, Physics.]

Figure 3.5: Precision@n (n=1,3,5) for key concept extraction from six textbooks. Local refers to the supervised learning model using local features defined in Section 3.2.3.2.3 with same experiment settings as Supervised KCE.

In Table 3.7, we present the F-1 score of concept relationship identification using top − 1, top − 3, and top − 5 candidates from the concept extraction phase respectively.

The results show that both features derived from Wikipedia and textbooks features achieve significantly higher F-1 score than hyponym-hypernym pattern does. Moreover, we observe that textbook features outperform Wikipedia features (Textbook features include Concept co-occurrence in book chapters, Relational strength in book contents and TOC distance measures. Wikipedia features include concept co-occurrence in Wiki pages, Content cosine similarity, RefD, Wikipedia
Table 3.7: F-1 score for concept relationship prediction. *Hyponym* refers to the hyponym-hypernym baseline method. *Textbook/Wikipedia* are supervised learning methods with features textbooks/Wikipedia features using same experiment settings as *Supervised RI*. * indicates when textbook features are statistically significantly better ($p < 0.01$) than the Wikipedia features.

<table>
<thead>
<tr>
<th></th>
<th>Hyponym</th>
<th>Wiki</th>
<th>Textbook</th>
</tr>
</thead>
<tbody>
<tr>
<td># candidate</td>
<td>1 3 5</td>
<td>1 3 5</td>
<td>1 3 5</td>
</tr>
<tr>
<td>Network</td>
<td>0.21 0.32 0.19</td>
<td>0.45 0.56* 0.54</td>
<td>0.49 0.6* 0.57</td>
</tr>
<tr>
<td>Economics</td>
<td>0.25 0.36 0.3</td>
<td>0.55 0.56 0.5</td>
<td>0.55 0.58 0.52</td>
</tr>
<tr>
<td>Precalculus</td>
<td>0.29 0.42 0.36</td>
<td>0.52 0.56 0.47</td>
<td>0.44 0.52 0.57</td>
</tr>
<tr>
<td>Geometry</td>
<td>0.28 0.36 0.41</td>
<td>0.48 0.56* 0.53</td>
<td>0.55 0.61* 0.59</td>
</tr>
<tr>
<td>Database</td>
<td>0.17 0.38 0.44</td>
<td>0.49 0.55 0.45</td>
<td>0.5 0.53 0.52</td>
</tr>
<tr>
<td>Physics</td>
<td>0.23 0.37 0.44</td>
<td>0.5 0.55 0.58*</td>
<td>0.6 0.58 0.62*</td>
</tr>
</tbody>
</table>

3.2.4.2 Joint Optimization

Table 3.6 shows the prediction accuracy of the baseline methods and the joint optimization model.

The proposed optimization model often outperforms all others with only an exception on precision@1 of database, as a trade-off to performance of concept
relationship prediction, as shown in Table 3.8. Our joint optimization model consistently outperforms the strongly baseline in F1-score on all the six textbooks. In addition, the proposed model can decide the number of concepts in each subchapter automatically by optimizing the proposed objective function while the baseline models depend on a manually decided value.

![Figure 3.6](image)

Figure 3.6: Precision@n (n=1,3,5) for key concept extraction from six textbooks with/without joint optimization.

### 3.2.4.3 Measurement Importance

In this section, we develop some insights regarding feature importance by reporting performance of the concept extractor and relationship identifier using different feature combinations.

Table 3.9 shows the ranking precisions of measurements using title information...
Table 3.8: F-1 score for concept relationship prediction with/without joint optimization. * indicates when the joint optimization model is statistically significantly ($p < 0.01$) better.

<table>
<thead>
<tr>
<th># candidate</th>
<th>Supervised RI</th>
<th>Joint Opt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network</td>
<td>0.52 0.62 0.57</td>
<td><strong>0.63</strong></td>
</tr>
<tr>
<td>Macroeconomics</td>
<td>0.55 0.67* 0.63</td>
<td><strong>0.7</strong></td>
</tr>
<tr>
<td>Precalculus</td>
<td>0.54 0.61 0.63*</td>
<td><strong>0.66</strong></td>
</tr>
<tr>
<td>Geometry</td>
<td>0.52 0.64 0.67*</td>
<td><strong>0.71</strong></td>
</tr>
<tr>
<td>Database</td>
<td>0.51 <strong>0.58</strong> 0.49</td>
<td>0.55</td>
</tr>
<tr>
<td>Physics</td>
<td>0.58 0.65* 0.62</td>
<td><strong>0.69</strong></td>
</tr>
</tbody>
</table>

Table 3.9: Precision@n (n=1,3,5) for key concept extraction from six textbooks using different feature combinations. Title/Content features include local and global features derived from title/content information as defined in Section 3.2.3.2.3.

<table>
<thead>
<tr>
<th># candidate</th>
<th>Title Features</th>
<th></th>
<th>Content Similarity Features</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1  3  5</td>
<td></td>
<td>1  3  5</td>
<td></td>
</tr>
<tr>
<td>Network</td>
<td>0.32 0.29 0.19</td>
<td></td>
<td>0.61 0.64 <strong>0.65</strong></td>
<td></td>
</tr>
<tr>
<td>Macroeconomics</td>
<td>0.43 0.39 0.32</td>
<td><strong>0.72</strong></td>
<td>0.65 0.54</td>
<td></td>
</tr>
<tr>
<td>Precalculus</td>
<td>0.41 0.36 0.27</td>
<td></td>
<td>0.66 <strong>0.68</strong> 0.59</td>
<td></td>
</tr>
<tr>
<td>Geometry</td>
<td>0.46 0.38 0.32</td>
<td><strong>0.79</strong></td>
<td>0.76 0.7</td>
<td></td>
</tr>
<tr>
<td>Database</td>
<td>0.24 0.18 0.11</td>
<td></td>
<td>0.66 <strong>0.69</strong> 0.58</td>
<td></td>
</tr>
<tr>
<td>Physics</td>
<td>0.3  0.22 0.16</td>
<td><strong>0.68</strong></td>
<td>0.62 0.59</td>
<td></td>
</tr>
</tbody>
</table>

and that using content information. As shown, content similarity features outperform title features since compared to subchapter titles, subchapter contents contain more information and achieve much higher recall rate in key concept extraction.

Figure 3.10 shows the performance of relationship prediction using Topically Relatedness features and Complexity Level Difference features defined in Section 3.2.3.3.2 respectively. We observe that complexity level difference features perform better than content similarity features. This suggests that by capturing which feature is more basic, we can achieve better performance than only consid-
Table 3.10: F1-score@n (n=1,3,5) for the relationship prediction using different measures.

<table>
<thead>
<tr>
<th># candidate</th>
<th>Topically Relatedness</th>
<th>Concept Complexity Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Network</td>
<td>0.41</td>
<td>0.49</td>
</tr>
<tr>
<td>Macroeconomics</td>
<td>0.47</td>
<td>0.52</td>
</tr>
<tr>
<td>Precalculus</td>
<td>0.53</td>
<td>0.58</td>
</tr>
<tr>
<td>Geometry</td>
<td>0.42</td>
<td>0.45</td>
</tr>
<tr>
<td>Database</td>
<td>0.52</td>
<td>0.59</td>
</tr>
<tr>
<td>Physics</td>
<td>0.3</td>
<td>0.22</td>
</tr>
</tbody>
</table>

We also observe that more fundamental subjects such as precalculus, geometry and physics have better performance than advanced subjects such as computer networks and database. A potential reason is that for those domains, different textbooks provide very similar learning and TOC structures, while for advanced subjects textbooks organize knowledge quite differently. However, this remains to be determined.

### 3.2.5 Case Study

Here we present a case study on concept maps extracted for geometry. From Figure 3.7c and 3.7d, we can observe that by considering both the lexical similarity and semantic relatedness, the *Supervised KCE + Supervised RI* method, here and after, the supervised learning method, and *joint optimization* model achieve better performance in both concept extraction from book chapters and relationship identification than the *TextRank+Hyponym-hyponym* method (see Figure 3.7a).
and Wikify + Hyponym-hypernym method (see Figure 3.7b). Moreover, supervised method and joint optimization model consider book structure information and this reaffirm the effectiveness of textbook structure in key concept extraction and concept relationship identification.

By capturing the mutual dependencies between two sub-problems, the joint optimization model achieves better prediction accuracy than the supervised learning method. For instance, supervised learning method fails to extract “Ray” and the prerequisite dependency between “Ray” and “Angle” while joint optimization model makes the correct prediction. A possible reason is that “Ray” is not extracted as a
key concept because its content is not very similar to the content of the book chapter (compared to other candidates in this chapter). However, the joint optimization model identifies that “Ray” is likely to be prerequisite of “Angle” which is highly ranked as a key concept in some book chapters.

3.3 Multi-Relation Concept Map Extraction

Used as a learning and teaching technique, knowledge graph visually illustrates the relationships between concepts and ideas. While prerequisite is an important relation for learning and education, other relations, such as “subclass of”, “instance of” and “has part” are also helpful to student learning. Previous work on extracting knowledge graph with multiple relations mainly focuses on real-world entities such as people, locations or organizations. For these work, information extracted from sentence syntactic structure and pos-tagging [85] are used as handcrafted features to train a supervised learning model which predicts relations between two testing entities. Also, Name Entity Recognition (NER) and Semantic Role Labeling (SRL) are used for this purpose. However, it may not be very effective to apply these information in constructing knowledge graph for education purpose due to the following reasons: 1) NER and SRL are very powerful for identifying semantic meaning for real-world entities as they provide information associated with type of a given mention, such as whether this mention is a location, people or time. However, such information is not very effective in predicting relations between scholarly
concepts. From this perspective, knowledge graph construction from academic resources is even more challenging than for real-world entities; 2) Most of the existing knowledge base focus on real-world entities and not many of them contain a complete graph about academic scholar. This issue makes the proposed problem even more challenging; 3) Compared to real-world entities, the relationship among entities are usually more sparse. For instance, we sample 10 entities including people, locations and organizations and another 10 scholar entities from Wikidata\textsuperscript{10} which contains statements (semantic triples with entities and relations) for each entity. On average, real-world entities have 9.8 statements while scholarly entities have 4.2 statements in Wikidata. In this section, we propose to extract a concept map with different types of relationships such as “is-a” and “instance of”. There are at least three lines of paradigms been applied for knowledge graph extraction from text. A supervised approach takes a set of sentences with entities and their relationship as label. Then a wide variety of lexical syntactic, and semantic features are extracted from these sentences. A supervised classifier is then used to predict the label of the relation mention holding between a given pair of entities in a test set sentence. However, labeled sentences are expensive to generate and thus supervised approach does not fit in our problem very well. Another way of conducting this job, unsupervised approach, extracts texts between two given mentions and clusters these texts as relationships. One problem of unsupervised information extraction is that the resulting relations is hard to map to relationships of an existing knowledge

\textsuperscript{10}https://www.wikidata.org
base. An alternative approach is bootstrapping which uses a small set of mentions or patterns as seeds. This method uses existing seeds to extract new patterns/mentions which will be used to extract more mentions/patterns. The resulting patterns often suffer from low precision and semantic drift.

To overcome the drawbacks mentioned above, we use distantly-supervised relation extraction \cite{71} to perform knowledge graph extraction. The intuition of distantly-supervised is that, given a pair of entities, any sentence that contains these two entities may express their relationship. Therefore, given a knowledge base and a document corpus, this method can extract large number of sentences serving as our training samples. Then features can be extracted from these sentences and fed into a classifier for relationship prediction.

In presenting these contributions, the rest of this section is laid out as follows. Local and global features are introduced in Section \ref{3.2.2}. In Section \ref{3.1.3}, we discuss the data preparation and evaluation metrics. In Section \ref{3.2.3}, we analyze the experimental results for geometry domain and precalculus domain.

**3.3.1 Distant Supervision Relation Extraction**

Relation Extraction here can be formulated as the task of predicting the relations expressed in natural language text. Consider, for example, the following sentence.

For the following sentence in Wikipedia page “Triangle”: “A triangle is a polygon with three edges and three vertices.”
In order to extract semantic triple “triangle is polygon” from this sentence, we first match entities “Triangle” and “Polygon” from this sentence using a pre-built dictionary. Since there are more than one entity appearing in this sentence, we proceed to the next step. After identifying the entities, we extract the following features from the sentence. Previous work usually extract a wide variety of lexical syntactic, and semantic features. While these features may not be powerful in our context as we analyzed above, we investigate a set of features using word embedding vector \[51\] information.

- \( E(h) \): word embedding of head entity \( h \). Here we use glove \[82\], a pre-trained word embedding vector to initialize our word representation.

- \( E(t) \): word embedding of tail entity \( t \). Similarly, we use glove as word embedding vector.

- \( E(\sum_i^k E(h_i) \ast idf(h_i)) \): word embedding of head entity \( h \) with word idf(inverse-document-frequency). Instead of simply taking average of words’ embedding, we multiply each word’s embedding by its idf to address different importance of different words.

- \( E(\sum_i^k E(t_i) \ast idf(t_i)) \): we adopt the same strategy for tail entity. For each word in the tail entity, we multiple its word embeddings with its idf.

- \( E(h) - E(t) \): difference between embedding of the head entity and the tail entity. The underlying reason to use feature is similar as mentioned in \[52\].
Relations are reflected in vector offsets between two entities $h$ and $t$, and by using simple vector arithmetic one could reveal nature of the relation.

- $E(\sum_i^k E(h_i) \times idf(h_i)) - E(\sum_i^k E(t_i) \times idf(t_i))$: difference between embedding of $h$ and $t$ with word idf. This feature leverages the difference between the embedding vector of $t$ and that of $t$. Meanwhile, it considers the importance of each word in the entity.

- $E(VP)$: word embedding vector of the Verb Phrase (VP) between $h$ and $t$. Verb phrases can provide important information in relationship prediction as one relationship usually uses similar verbs.

- $E(\sum_i^k E(VP_i) \times idf(VP_i))$: similar to other features with word idf information, this feature considers importance of each word in the phrase.

After extracting the above features, we fed all training samples into a SVM Multi-class classifier.

### 3.3.2 Data Preparation and Experiment Settings

#### 3.3.2.1 Data Preparation

In order to obtain our training data, we first extract triples with entities and relations between entities from web knowledge base. Our resource for triple extraction contains two parts:
• **Wikidata**: Wikidata is a free and open knowledge base. Wikidata currently contains 26,168,216 items. For each item, Wikidata provides statements that describe detailed characteristics of each item which can serve as semantic triples for our work. Figure 3.8 shows a Wikidata example for entity “Triangle”. As we can see from this example, it contains several statements for entity “Triangle”. For instance, “Triangle” is “subclass of ” “Bientric polygon”. We extract 26352 semantic triples from wikidata.

Figure 3.8: Example of an extracted concept map in economics.

• **Wikipedia**: Wikipedia is an open-source encyclopedia which contains more than 4 million entities. Given a domain, we follow the following steps to extract semantic triples from Wikipedia: 1) For each domain, we extract a domain dictionary following the method proposed in Chapter 3.1.2.1 2) For each entity in the dictionary, we use ReVerb [23], which is a program that automatically identifies and extracts binary relationships from English sentences, to extract triples from sentences in its Wikipedia page; 3) For each extracted triple, if its head entity and its tail entity are both in the dictionary
and two entities are in the same clause in the original sentence, we add this triple to our candidate pool; and 4) For each candidate triple in the candidate pool, we manually label the relationship between two entities as “instance-of”, “part-of”, “has-part”, “has-consequence”, “same as” or “invalid”.

### 3.3.3 Experimental Results

We conduct relation prediction on geometry domain and precalculus domain and report F-1 score in this section. Table 3.11 shows experiment result for knowledge graph embedding based method. Entity embedding features include embedding vectors of the head and title entity, i.e., the first two features described in Section 3.3.2.1. Entity embedding (idf) features include entity embedding vector with word idf information, i.e., the 3rd and the 4th features in Section 3.3.2.1.

As shown in the table, features using verb phrases information have the best performance as verb phrases contains information about relationships directly. Entity embedding related features have the worst performance. One possible reason is a specific type of relations may correspond to many different pairs of entities. This can be proved by the improvement of using features entity embedding diff as relations are reflected in vector offsets between two entities. Lastly, incorporating word idf information does not improve prediction result significantly and sometimes even decreases the result (see prediction results of using entity embedding diff and entity embedding diff (idf) for geometry domain). One possible reason is that entity
Table 3.11: F-1 score for different feature combinations.

titles are usually quite short and traditional text similarity methods such as tf-idf cosine-similarity, based on word overlap, mostly fail to produce good results in this case, since word overlap is little or non-existent [18].
For this section, we shift focus from creation of knowledge graph to applying knowledge graph in education. To be specific, we investigate two educational applications using knowledge graph: 1) BBookx, an automatic book creation system which supports dynamic book creation and concept recommendation; and 2) an automated assessment system using prerequisite concept maps.

4.1 BBookx: Dynamic Book Creation

This section is based on the following work:


With the rapid growth of available education resource, creating a contemporary personalized book has many advantages. Creation customized and personalized
learning materials has attracted more and more attention. While resources such as MOOCs[^1], Wikibooks[^2] and Wikiversity[^3] provide accessible and flexible learning options, creating high-quality learning resources on a large scale is still a challenging problem, especially for fast changing domains such as computer science.

In this work, we investigate the BBookX system which automatically collects online resources related to user input and organizes them in a format of a personalized learning resource similar to a textbook. Specifically, we leverage Wikipedia by automatically linking user input to Wiki articles. With an interactive user interface, BBookX supports real-time book creation with an interactive user input and adopt an explicit relevance feedback mechanism which utilizes user feedback to enhance the book creation. The query for additional searches.

In the following section, we give an overview for the system. An example of creating a textbook using provided interactive user interface is also given in this section.

### 4.1.1 System Overview

There are four major components in the system of BBookX: 1) Open Educational Resources (OER) Retrieval; 2) Indexing; 3) Querying and 4) Interactive book creation interface.

[^1]: https://www.mooc-list.com/
[^2]: https://en.wikibooks.org/
[^3]: https://en.wikiversity.org/
Pre-Calculus

Get Started

Book Title: Pre-Calculus

Describe the Book:

TECHNIQUES OF CALCULUS I (4) Functions, graphs, derivatives, Integrals, techniques of differentiation and integration, exponentials, Improper integrals, applications. Students may take only one course for credit from MATH 110, 140, 140A, and 140B. Prerequisite: MATH 122 or satisfactory performance on the mathematics proficiency examination.

Course Overview:
The goal for the course is to cover the chapters/sections detailed in the tentative class schedule below. Chapter 1 is considered review material for the students. Each student should confirm that they understand the material in Chapter 1 during the first week of the course.

(a) Interface for book creation.

Add a Chapter

Chapter 1

Chapter Title:

Describe the Chapter:

In mathematics, a function\([f]\) is a relation between a set of inputs and a set of permissible outputs with the property that each input is related to exactly one output. An example is the function that relates each real number \(x\) to its square \(x^2\). The output of a function \(f\) corresponding to an input \(x\) is denoted by \(f(x)\) (read "\(f\) of \(x\)"). In this example, if the input is \(-3\), then the output is \(9\), and we may write \(f(-3) = 9\). Likewise, if the input is \(3\), then the output is also \(9\), and we may write \(f(3) = 9\). (The same output may be produced by more than one input, but each input gives only one output.) The input \(x\) values are sometimes referred to as the argument(s) of the function.

(b) Interface for adding chapters.

View Results

Chapter 1

Functions

1. Function (mathematics)
2. Inverse function
3. Coherence (signal processing)
4. Trapdoor function
5. Miller effect
6. Inverse relation
7. Perturbation function
8. Brillouin and Langevin functions
9. Composition of relations

(c) Initial results for the chapter.
• OER Retrieval. Wikipedia is a large-scale and online encyclopedia and provide students flexible learning opportunities. In this work, we propose to use Wikipedia as our backend knowledge base. BBookX is currently using a Wikipedia dump which consists of more than 4.9 million English Wiki articles.

• Indexing. In order to allow real-time access to the backend database, we create a local index for Wikipedia. We first apply preprocessing including tokenization, stop word and punctuation removal, conversion to lower case, and stemming, then build an index for each document using Apache Solr. Keyphrases of each document are also extracted using Maui and a similarity score between the document and each keyphrase is pre-computed.

• Querying. Given the user input $q$, BBookX first extracts a list of keyphrases from $q$ and then retrieves a list of candidate Wikipedia pages using pre-calculated phrase-document similarity. BBookX then re-rank candidates by calculating the similarity score based on a weighted combination of title similarity, content similarity, and keyphrase similarity. The system also improves search results by allowing users to choose whether they want to accept or discard a specific page. Specifically, the top 20 keyphrases from pages kept by users are selected based on term frequency and used to enrich the existing query $q$. 
• Interactive Book Creation Interface. Figure 4.1 shows an example of creating a book using the provided interactive book creation interface.

As shown in Figure 4.1a, users start with register and login. After logging into the system, users can create a chapter by clicking the "Add Chapter" button as shown in Figure 4.1c. Chapter title and a piece of description about the chapter is required as system input to extract related Wikipedia pages. After providing these information, users can click the "Run" button to ask the system to retrieve a list of relevant Wikipedia pages. Here we provide an explicit user-feedback mechanism to enhance the extracted content by allowing user to modify the search result. For each retrieved Wikipedia page, users can either keep it or discard it. They can also click "Regenerate Results" to re-run the search and our system will reformulate the query using the page content that is kept by users.

4.2 Automated Assessment using Prerequisite Concept Maps

This section is based on the following work:

The increasing growth of massive online education resources promise new possibilities for educational tools and services. Assessment, which is the task of assessing learning achievements and providing feedbacks to learners is crucial to academic success and has been rapidly changed recent years [20]. Concept map, which provides a clear view of knowledge structure, is widely used in assessments [4,117]. In this paper, we develop an automatic personalized assessment learning system to improve the efficiency for students’ problems fixing using concept maps. We utilize a prerequisite concept map, which represents domain concepts and their learning dependencies, to detect and fix each student’s own knowledge gaps. For instance, a student does not understand “Multiplication” might because he fails to know how to do “Addition”, which is a prerequisite for “Multiplication”. In this situation, instead of keeping the student working with “Multiplication”, a better idea may be move him backwards by re-learning “Addition”.

A running example of using prerequisite concept maps for assessment is shown in Figure 4.2. Students can start with any concept in the concept map, for instance, ”Edge”. LearningAid will provide student questions about rectangle. If the student gives the correct answer, which means that he probably understand this concept, the system will recommend him to move forward along the prerequisite relationship, i.e., moving to the subsequent concepts of “Edge”, for example, “Rectangle”. When the student accepts the recommendation, he will answer questions about “Rectangle”.
If the ”Rectangle” question is correctly answered, the system will repeat the process above, i.e., recommending the subsequent questions of “Rectangle”. Otherwise, if the question is not correctly answered, it means that the student does not understand the concept “Rectangle” well and probably he also has problems with prerequisites of “Rectangle”. Therefore, the system will recommend some prerequisite of “Rectangle” for the student to study, for example, “Right Angle”. The system then repeats process describe above to facilitate assessments. By automatically recommending next step for students based on the current assessment results, our assessment systems with prerequisite concept maps offer personalized assessment learning experience. In summary, the system makes the following novel contributions:

• We propose a novel automatic personalized assessment system which finds students’ learning gaps and work on closing the gaps using prerequisite concept maps.

• We investigate a two-phase model for prerequisite concept map construction which includes domain concepts extraction and prerequisite relationships identification.

• We deliver the concept map to students in a hierarchical way and allow them to view concepts in different levels.
4.2.1 Prerequisite Concept Map Construction

To automatically construct prerequisite concept map, we propose a two-phase method which extracts domain key concepts from expert created educational resources such as textbooks and papers, and then identifies prerequisite relationships between extracted concepts.

Figure 4.2: A running example for using prerequisite concept map to help student fix learning gap.

4.2.1.1 Key Concept Extraction

Given a domain, our goal is to exact key concepts related to this domain from educational resources. In this paper, we use Wikipedia to help the key concept extraction and enrich article content. We first construct a domain specific concept dictionary in which each concept is the title of a domain related Wikipedia page. Then given an article, we identify all Wikipedia concepts in the article using this dictionary and obtain a list of Wikipedia candidates. At last we select top-k concepts using the following features:

**titleMatch:** A Wikipedia concept is likely to be a key concept in an article if
its title appears in the article’s title.

**cosineSim**: A Wikipedia concept is likely to be a key concept in an article if it has similar lexical contents with the article. \textit{cosineSim} captures the cosine similarity between the concept vector of Wikipedia candidate and that of the article.

Therefore, the domain key concept set consists of the top-\textit{k} candidates based on \textit{cosineSim} score and those candidates with \textit{titleMatch(·)} score equals to 1.

### 4.2.1.2 Prerequisite Relation Identification

Previous works mainly focus on hyponym relationship inference, which does not serve education purpose appropriately. Here we argue that there is likely to be prerequisite relationship between concept \textit{A} and \textit{B} if:

**Usage of one concept in the definition of the other concepts**: If concept \textit{A} is used in \textit{B}’s definition, \textit{A} is likely to be \textit{B}’s prerequisite.

**Similar Content and Different Learning Levels**: If two concepts cover similar topics, it is likely that they have some learning dependencies. For instance, “Network congestion” and “TCP congestion-avoidance algorithm” share a lot of topics such as packet loss and additive increase/multiplicative decrease, and “TCP congestion-avoidance algorithm” depends on “Network congestion”. However, ”TCP congestion-avoidance algorithm” and “Network security” do not share a lot of common topics and it is unlikely that there is a prerequisite relationship between them.
However, not all pairs of concept with similar content have prerequisite relationships. For example, “Transmission Control Protocol” and “User Datagram Protocol” cover similar topics while they are concepts at equivalent level of learning. Therefore, given two concepts, it is necessary to identify whether they are at different learning levels for prerequisite relationship inference.

Base on above arguments, we investigate three sets of features given pair of concepts $A$ and $B$:

**Usage in Definition Feature** Feature $usage(\cdot)$ captures whether a concept is used in another concept’s definition. $usage(A, B) = 1$ if $A$ appears in $B$’s definition. A challenge is to obtain definitions of Wikipedia concepts. Here given a Wikipedia concept, we use the first sentence in its Wikipedia page as its definition. The reason of doing this is that most Wikipedia pages have a unified pattern with their first sentences as “concept is definition of the concept”. For instance, Wikipedia concepts “Logarithm” and “$e$ (mathematical constant)” are used to define “Natural logarithm” (definition for “Natural logarithm”) is “The natural logarithm of a number is its logarithm ... ” and $usage(logarithm, naturallogarithm) = 1$. In this case, “Logarithm” is a prerequisite of “Natural Logarithm”.

**Content Similarity Feature** This feature captures the lexical similarity between Wikipedia concepts. We use feature $cosineSim(\cdot)$ defined in Section 4.2.1.1 to capture the content similarity between $A$ and $B$.

**Learning Level Features** Learning level features measure whether a concept
has a lower learning level and should be learned first. In order to capture this more precisely, we investigate three features to calculate the learning level of a concept: range of topic coverage, number of in-links and number of out-links. Given a collection of Wikipedia concepts $C$ and a Wikipedia concept $c_i$, we define the following measurements:

**Range of topic coverage:** We first measure the learning level of a concept based on the range of topics covered by the concept [44]. Essentially, the more topics that a concept covers, the more basic the concept is. For instance, “Computer Network” covers more topics than “TCP” does and has a lower learning level and should be learned first. The range of topic coverage score $\text{tc}(c_i)$ of $c_i$ is the Shannon Entropy over topics discussed in $c_i$. To be more specific, we run a topic model on $C$ to generate $s$ topics and the topic distributions for each concept in $C$. A topic model generates a document-topic matrix $F_{n \times s}$ where $n$ is the number of concepts in the corpus. $F_{im}$, where $F_{im}$, with $i < |C|$ and $m < s$, is the probability that topic $m$ is assigned to $c_i$. Then we compute the Shannon Entropy $H(c_i)$, i.e., the range of topic coverage score $\text{tc}(c_i)$, based on $F_{n \times s}$ as follows:

$$ \text{tc}(c_i) = H(c_i) = \sum_m -F_{im}\log(F_{im}). $$  \hspace{1cm} (4.1)

**Number of in-links/out-links received:** Besides the content information, millions of cross-page links in Wikipedia is also useful in detecting concept learning levels. If $c_i$ receives a lot of in-links from other concepts, it is likely that $c_i$ is fundamental
in $C$ and should be learned first. Similar conclusion can be drawn on the number of out-links of a concept.

Based on features defined above, for each concept, we then normalize value of each measurement to [0,1] range, and take their weighted sum as the learning level score of the concept. The higher the learning level score is, the more basic the concept is.

We then investigate a threshold-based algorithm which selects pairs of concepts with prerequisite relationships. Given concepts $A$ and $B$, if $usge(A, B) = 1$, we consider $A$ to be the prerequisite of $B$ and the algorithm stops; Otherwise, if the learning level difference and the content similarity between $A$ and $B$ are greater than some thresholds, $A$ is considered as a prerequisite of $B$.

### 4.2.1.3 Prerequisite Concept Map Delivery

One challenge in assessment system using prerequisite concept map is that there might be hundreds of concepts in a domain. In this case, how can the system deliver the concept map to students? To meet these challenges, we investigate a top-$k$ concept selection algorithm where $k$ is a user specified parameter. In this case, the system constructs a subgraph of the original concept map which consists of $k$ concepts in the domain where students can zoom in or zoom out to view different numbers of concepts. The rule of selecting top-$k$ concepts is to choose domain important concepts while preserve connectivity of the subgraph. Given a
Wikipedia concept $A$, we define $A$’s importance $I_A$ as the similarity score between $A$’s contents and article’s content. For the connectivity of the extracted subgraph $G = (V, E)$, where $V$ is the concept set and $E$ is the edge set in $G$, we use its edge density $D$ as a measurement for its connectivity, which is defined as $\frac{2|E|}{|V|^2(|V|-1)}$.

The algorithm selects top-$k$ concepts by maximizing the weighted sum of overall concept importance and subgraph edge density which is defined as $\arg\max_{V} (\alpha \sum_{k=1}^{V} I_k + (1 - \alpha)D(V))$, where $\alpha$ is the weight of concept importance and $1 - \alpha$ is the weight of subgraph connectivity.

The procedure of the algorithm is described as below: 1) Initialize the concept set $V^{(0)}$ using top-$k$ important concepts in the article. 2) For each iteration, concept set $V^{(n+1)} = V^{(n)} \setminus \{s\} \cup \{c\}$, where the replacement of concept $s$ by concept $c$ maximizes the value of objective function. 3) Iteratively update the concept set until there is no replacement that could increase the value of objective function.

4.2.2 Experiments

4.2.2.1 Experiments Setup

In order to evaluate the extracted pairs of prerequisite relationship, we manually create dataset using a mathematics textbook\textsuperscript{4} and a big data textbooks\textsuperscript{5}. We first extract domain key concepts from the book following the methods proposed


\textsuperscript{5}McKinsey Global Institute. Big data: The next frontier for innovation, competition, and productivity
in [109] by extracting candidate concepts from each book chapter using feature titleMatch and cosineSim and manually label each concept as “Important” and “Unimportant”. Then we randomly select $N$ concepts from the key concepts and manually label its subsequent and prerequisite concepts. Then we use this dataset as ground truth and report the F-1 score of our extracted concept map.

To affirm the effectiveness of our proposed learning level features, we propose a baseline which only uses features usage in definition and Content Similarity to identify concept relationships between concepts.

4.2.2.2 Experiment Results

Table 4.1 and Table 4.2 show the accuracy of extracted concept relationships using different thresholds of concept relatedness and learning level. The column title is the threshold of concept relatedness and the row title is that of concept learning levels.

Table 4.1: Accuracy for mathematics concept map extraction

<table>
<thead>
<tr>
<th></th>
<th>40%</th>
<th>60%</th>
<th>80%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.31</td>
<td>0.46</td>
<td>0.49</td>
</tr>
<tr>
<td>TwoPhase</td>
<td>0.11</td>
<td>0.05</td>
<td>0.16</td>
</tr>
<tr>
<td></td>
<td>0.43</td>
<td>0.61</td>
<td>0.39</td>
</tr>
<tr>
<td>Baseline</td>
<td>0.08</td>
<td>0.04</td>
<td>0.1</td>
</tr>
<tr>
<td>TwoPhase</td>
<td>0.13</td>
<td>0.55</td>
<td>0.41</td>
</tr>
<tr>
<td></td>
<td>0.42</td>
<td>0.04</td>
<td>0.08</td>
</tr>
<tr>
<td>Baseline</td>
<td>0.13</td>
<td>0.55</td>
<td>0.41</td>
</tr>
<tr>
<td>TwoPhase</td>
<td>0.13</td>
<td>0.55</td>
<td>0.41</td>
</tr>
</tbody>
</table>

Take “60% 80%” in Table 4.1 as example, this means that if we select pairs of concepts with relatedness score higher than 60 percent of relatedness scores between all pairs of concepts and learning level difference higher than 80th percent of all
learning level difference scores as prerequisite concept pairs, the prediction accuracy is 0.39. As shown in Tables 4.1, incorporating the proposed learning level features does achieve better results than the baseline model. The major reason is that learning level features considers the complexity of a concept, which is a key factor in deciding prerequisite relationships between concepts while the other two sets of features only consider the relatedness between concepts. Moreover, we observe that precalculus have better performance than big data and, when not, is very close. A potential reason is that precalculus is a more fundamental subject than big data and concepts within this domain have more clear learning dependencies.

<table>
<thead>
<tr>
<th></th>
<th>40%</th>
<th>60%</th>
<th>80%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.35</td>
<td>0.43</td>
<td>0.42</td>
</tr>
<tr>
<td>TwoPhase</td>
<td>0.157</td>
<td>0.114</td>
<td>0.128</td>
</tr>
<tr>
<td>Baseline</td>
<td>0.39</td>
<td>0.52</td>
<td>0.39</td>
</tr>
<tr>
<td>TwoPhase</td>
<td>0.032</td>
<td>0.1359</td>
<td>0.154</td>
</tr>
<tr>
<td>Baseline</td>
<td>0.39</td>
<td>0.51</td>
<td>0.389</td>
</tr>
<tr>
<td>TwoPhase</td>
<td>0.198</td>
<td>0.032</td>
<td>0.098</td>
</tr>
</tbody>
</table>
Chapter 5  |

Implications

This chapter discusses implications from the results presented in previous chapters for both knowledge graph extraction and its education applications. The first section focuses on the implications of constructing the knowledge graph with academic resources and knowledge bases. This section extends from the practices of identifying key concepts, extracting concept relations, and constructing knowledge graphs in ways that differ from using various domains, different features, and multiple concept complexity levels for different types of knowledge graphs regarding relations between concepts in a graph. The second section discusses the dataset used in knowledge graph extraction, pointing out the assumptions and limitations of the proposed method for dataset construction.
5.1 Implications for Knowledge Graph Construction using Academic Resources

We found that using academic resources, specifically, textbooks, can improve knowledge graph extraction significantly. The rich structure information embedded in textbooks is particularly useful for identifying hierarchies among concepts. Moreover, experimental results show that we can obtain a better knowledge graph with the aid of web knowledge, in our case, Wikipedia and Wikidata. On the other hand, the lack of effectiveness of the resources (textbooks and web knowledge bases) for extracting the knowledge graph with multiple relationships, such as “is-a”, “instance-of”, and “has-part”, is also interesting as it raises questions about predicting generic relationships among scholarly concepts. In the following section, we describe: 1) the proposed features for extracting the hierarchical structure and compare these two sets of features; 2) the proposed method for identifying a knowledge graph with prerequisite relationships; 3) the proposed method for identifying generic relations among concepts; and 4) how performance on different domains vary. For each perspective, we discuss both the advantages and limitations.

5.1.1 Hierarchical Structure Extraction

We propose a method to extract concept hierarchies from books and formalize the Concept Hierarchy Extraction from Book (CHEB) task as an optimization
problem with local and global invariants. We first extract key concepts from each book chapter and organize these concepts using the book’s TOC to formulate a concept hierarchy for the given book. We consider the local relatedness between the extracted concepts and a book chapter. Moreover, we proposed a set of global features by considering the textbook’s TOC. Global features ensure that the extracted concept hierarchy is less redundant, more consistent, and follows a consistent learning order. Taken altogether, the major implications of this work are: 1) local features are helpful for concept hierarchies extraction. Among all local features, similarity between the concept candidate (Wikipedia page) title and the chapter title is particularly important for chapter key concept identification; 2) the extracted concept hierarchies can be optimized through combining local and global features. Moreover, as we can see from the results, the global approaches are very effective for improving the prediction of the top-one ranked concepts. The potential reason is that each book chapter usually only discusses one or two concepts; and 3) textbooks are useful for constructing concept hierarchies. Textbooks are written by domain experts and contain high-quality information. The textbook structure is helpful to ensure the consistency and learning order and can reduce the redundancy of the extract concept hierarchies.

However, this work has several limitations: 1) no explicit relationship between concepts is extracted in this work. We only perform key phrase extraction for each chapter and utilize the textbook’s TOC to identify the implicit hierarchical
structure among the concepts; 2) we formulate our problem as a learning-to-rank problem and assume that there are a fixed number of concepts in each book chapter for evaluation purposes. This assumption does not always hold as different book chapters can discuss a different number of concepts.

5.1.2 Knowledge Graph with Prerequisite Relationships

In this work, we describe measures that identify prerequisite relationships between concepts. We propose a joint optimization model for knowledge graph extraction from textbooks that utilizes the mutual inter-dependency between key concept extraction and the identification of related concepts from knowledge structures in Wikipedia. Experiments on six knowledge graphs created manually from six different textbooks show that the proposed method gives promising results. We utilize the implicit prerequisite relationships embedded in textbook TOC’s for prerequisite relationship extraction.

Similar to extracting concept hierarchies from textbooks, this study shows that both textbooks and web knowledge are helpful in identifying key concepts for each book chapter.

Incorporating textbook structure information and web knowledge can enhance prerequisite relationship extraction. We found that the textbook structure is particularly useful. The underlying reason is that textbooks are usually written with the order that prerequisites of a concept are introduced. Therefore, the
textbook structure can reveal prerequisite relationships between concepts effectively. Other important information in determining relationships between concepts is the first sentence in a concept’s Wikipedia page. The first sentence in a Wikipedia page is usually well formatted and a brief description of the concept. Thus, we can easily extract hypernym-hyponym relations, which usually infer prerequisite relationships between two concepts, from the sentence.

The most important, joint optimization of key phrase extraction and relationship identification can enhance the extracted knowledge graph. This model optimizes results by considering the mutual reinforcement of these two steps utilizing the textbook structure.

The main limitation of this work is the evaluation part. Currently, to validate the proposed local and global features, we manually construct concept hierarchies for six well-used textbooks. Though the manually created data set is of high quality and precise enough for us to try to study the global coherence embedded in the books, it is rather small. Moreover, it is very expensive to manually label the knowledge graph from a textbook and it is difficult to validate our model on new domains. This issue makes the need for an unsupervised way of learning the knowledge graph from academic resources necessary.
5.1.3 Performance for Different Textbooks

This section addresses the model’s performance on different levels of textbooks and domains. The proposed model is tested on six different domains: computer network, macroeconomics, database, pre-calculus, geometry, and physics. A popular textbook was chosen as the test bed for each domain. To test our model on different levels of textbooks, we choose entry-level textbooks for pre-calculus, geometry, and physics, and advanced level textbooks for computer networks, macroeconomics, and databases. We have the following implications: 1) for entry-level textbooks, we have a high agreement between annotators. One potential reason disagreement exists is that annotators have different types of understanding towards a concept. Thus, it is easier for annotators to achieve agreement for relationships between two simple concepts than that between two complex concepts; 2) Our model performs better on entry-level textbooks than on advanced-level textbooks. First, as we mentioned above, datasets constructed from entry-level textbooks are of higher quality. Thus, our proposed model can achieve better performance on entry-level textbooks. Second, though having the same learning objectives, advanced textbooks usually perform different ways of teaching methods, e.g., top-down and bottom-up teaching method, while entry-level textbooks have a more consistent way of teaching method. The proposed model integrates both textbook’s structure and web knowledge base information, and entry-level concepts usually have consistent order between these two types of resources. Therefore, it is easier for the proposed model
to perform better on entry-level textbooks.

5.2 Dataset for Knowledge Graph Extraction

5.2.1 Textbooks

To prepare the textbooks for our input, we manually choose several domains and collect high-quality textbooks from this domain. Due to the limited availability of Extensible Markup Language (XML) or Hypertext Markup Language (HTML), we use the Portable Document Format (pdf) version of textbooks as the initial input to our model. A pdf-parser based on pdfbox was built to convert text in a pdf document into plain text. However, this setting for data collection has several limitations related to the textbook data: 1) availability of high-quality textbooks: it is very expensive to manually collect high-quality textbooks for each domain; 2) textbook parser: there are limits to this parser as identifying the text structure from a pdf document is very difficult. Not all textbook pdfs have the same structure (e.g., single-column and double-column issue), it is essentially not easy for the parser to parse different textbooks without manually changing the settings in the code. Therefore, being able to automatically collect and parse these textbooks will be pivotal for further research applying knowledge graph extraction to this context.
5.2.2 Domain Dictionaries

For efficiency purposes, we propose a heuristic for constructing a dictionary which contains related concepts for each domain. However, not all the concepts mentioned in the textbook can be extracted by our method. Although this method has good coverage for most textbooks, there are some concepts that are not included in the dictionary. Moreover, not all concepts appear in Wikipedia; some emerging domains such as deep learning and 3D printing make some problem more challenging. “Unrelated concepts” such as people, locations, and organizations are sometimes collected by our crawler as well. To manage this, we design a heuristic to filter out those concepts using its category information. While this ad-hoc rule-based filter can remove some of the unrelated concepts, many still remain unresolved and influence the prediction precision.
Chapter 6 | Conclusion and Future Work

The knowledge graph visually illustrates the relationships between concepts and ideas and links concepts by relationships to help students organize and structure their thoughts [76]. The knowledge graph serves multiple purposes for learners by allowing them to discover new concepts and clearly communicate ideas, thoughts, and information. Nowadays, with the larger and growing amount of online education data, automation of the knowledge graph is becoming increasingly important. Earlier work in knowledge graph extraction for educational purposes solely treated academic resources as plain text and did not make full use of the rich structure information in the resources [12]. In this thesis, we propose to extract the knowledge graph from academic resources by fully considering the structure information embedded in the articles. To be specific, we propose to use high-quality textbooks to enhance the extracted knowledge graph. Moreover, we leverage web knowledge bases to enrich the inside textbook knowledge and optimize the extracted knowledge
We first present a learning-to-rank model with local and global features to extract concept hierarchies from a textbook. By identifying key concepts for each book chapter and formulating a concept hierarchy by organizing these concepts using the textbook’s TOC, this model considers both the local similarity between a concept candidate and book chapter content and the global structure of the extracted concept hierarchy. Experimental results show that incorporating global features can enhance the extracted knowledge graph by ensuring consistency, less redundancy, and the correct learning order in the knowledge graph.

Previous work only [109] represented a hierarchical structure without explicit relationships between concepts. However, relationships between concepts that explain the connections between the concepts by words and phrases can help students to further understand information. Therefore, the second work we propose aims to extract knowledge with explicit relationships between concepts. In this work [110], we are interested in prerequisite relationships among concepts. A prerequisite is a concept that one needs to complete before preceding to another one under this context. Prerequisites are a way of making sure that students enter learning with prior knowledge and is important in education. There are two major steps to extract a knowledge graph with relationships among concepts: key phrase extraction and relationship identification, and previous work usually performed these two steps separately. However, these two steps are not independent:
information from key phrase extraction can enhance the extracted relationship and vice versa. In this work, we propose a joint optimization model to improve both phases by considering the mutual reinforcement between these two phases. The proposed model is tested on a high-quality dataset manually extracted from six textbooks and experimental results prove the strength of our model in enhancing the extracted knowledge graph.

Though prerequisites are important for learning and education, it will be helpful to construct a knowledge graph with multiple types of relationships as some generic relationships such as “is-a”, “has-part”, and “instance-of” can encourage students to brainstorm information and understand new relationships. Therefore, we propose another work which aims to construct a knowledge graph with multiple types of relationships. Earlier literature [58, 85] mainly focused on real-world entities such as people, locations, and organizations. However, it is more challenging to extract such a knowledge graph for scholarly concepts. First, as we mentioned in Chapter 3.3, training data for scholarly entities is more sparse than that for real-world entities; second, some well-used features are not as powerful in our work as in previous literature. For instance, semantic role labeling is frequently used in previous work as it is useful to distinguish between location and time. However, it is not that helpful in our context. To build a knowledge graph with multiple types of relationships, we first collect triples (two entities and their relationship) from wiki data and manually label part of the triples using Wikipedia. After collecting
the dataset, we conduct two different ways of knowledge graph construction: 1) for each triple, we extract features from sentences which contain both entities which are extracted and split the data into training data and testing data. A supervised classifier is then trained on the collected training samples and tested on the testing samples; 2) we also perform knowledge graph embedding learning which aims to learn a low-dimension vector for each entity and relationship. This method is more extensible and flexible as it does not need any handcraft feature.

The last part of our thesis shifts focus from knowledge graph construction to application of the extracted knowledge. The first system we discussed is BBookX, which automatically creates textbooks given a user specified input. BBookx can automatically build high-quality textbooks by extracting Wikipedia pages related to user specific topics. Another application we discussed is to use a prerequisite knowledge graph to perform automatic assessment. We utilize prerequisite knowledge graphs to find students’ learning gaps: for each concept in the graph, this system will provide a question. If the student provides the correct answer, the system will randomly select another concept for the next step; otherwise, the system will go back to the prerequisites of the original concept until the student provides the correct answer.

This dissertation shows the strengths of the combination of academic resources and web knowledge bases in knowledge graph extraction. The investigated applications prove that knowledge graph construction is important and promising for
educational purposes. There are several future directions to explore.

One future direction is knowledge graph construction using academic papers. As mentioned in Chapter 1, one of the challenges of automated knowledge graph construction is the various natures of the different domains. To generalize the proposed solution to different domains, gathering sufficient data from different subjects is necessary. One solution is to collect open textbooks. However, for some emerging areas like big data or deep learning, it is difficult to obtain up-to-date textbooks. An alternative way is to utilize academic papers, which are more advanced and of high quality, as resources to construct knowledge graphs. Moreover, we could use citation relations to help infer the concept relations in the paper. We propose two assumptions here to perform the pilot study: 1) all citations are made by prerequisite purposes, and 2) concepts in the citing paper rely on all the concepts in the cited paper. There are over 40,000 papers from ACM in CiteSeerX [58], which is a digital library that contains approximately 3.5 million scholarly documents and receives between 2â€“4 million requests per day.

Another challenge in knowledge graph construction is to consider different levels of abstraction when constructing concepts for users of different levels of expertise. To present an easyâ€“difficult knowledge graph, one way is to first present concepts which have few prerequisites. To present knowledge graphs of different expertise levels, it is necessary to obtain concept complexity. Concept prerequisites can represent concept complexity to some extent. If one concept has
many prerequisites, this concept is likely to be complicated. However, the number of prerequisites of a concept is sometimes not an effective metric for its complexity. For instance, concepts 'Right Triangle' has prerequisites 'Triangle', 'Right Angle', 'Edge', and 'Vertex' while concept 'Similar Triangle' has prerequisites 'Triangle', 'Edge', and 'Vertex'. From the number of prerequisites of the two concepts, 'Right Triangle' is more complicated than 'Similar Triangle'. However, 'Similar Triangle' is more complicated than 'Right Triangle' and students usually have more questions understanding the former. To the author’s best knowledge, there is no previous work that conducts concept complexity for different levels of abstraction. Two related techniques fall into concept prerequisite relationship, namely identify and term informativeness. One future direction would be to construct knowledge graphs from multiple books from the same area or use similar textbooks to modify each other’s knowledge graphs. Another direction would be to develop a semi-automatic method for building large-scale education area knowledge graphs.
Bibliography


120


124

[77] Joseph D Novak and Alberto J Cañas. The theory underlying concept maps and how to construct and use them. 2008. 1


[87] Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. Local and global algorithms for disambiguation to wikipedia. pages 1375–1384, 2011. [20] [37]


[115] Ian Witten and David Milne. An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In AAAI, pages 25–30, 2008. [40] [65]


Vita
Shuting Wang

RESEARCH INTEREST

Machine Learning, Text Mining and Natural Language Processing

EDUCATION

The Pennsylvania State University 08/2012 - 08/2017
Ph.D. Computer Science and Engineering
Advisor: Dr. C. Lee Giles
Harbin Institute of Technology 08/2008 - 07/2012
Bachelor of Engineering

EXPERIENCE

Facebook, Search 06/2016 - 09/2016
Software Engineer Intern Menlo Park, CA
HP Labs, 3D & Printing 06/2015 - 09/2015
Research Intern Palo Alto, CA
Baidu, Web Data Mining 01/2012 - 06/2012
Software Engineer Intern Beijing, China

SELECTED PUBLICATIONS


Shuting Wang and Lei Liu. Prerequisite concept maps extraction for automatic assessment. In Learning & Education with the Web of Data (Best Paper Award), 2016.

