Concept Hierarchy Extraction from Textbooks

Shuting Wang[†], Chen Liang[‡], Zhaohui Wu[†], Kyle Williams[‡],
Bart Pursel[‡*], Benjamin Brautigam[*], Sherwyn Saul[*], Hannah Williams[*],
Kyle Bowen[*], C. Lee Giles[‡†]
[†]Computer Science and Engineering
[‡]Information Sciences and Technology
[*]Teaching and Learning with Technology
Pennsylvania State University, University Park, PA 16802, USA
sxw327@cse.psu.edu, cul226@ist.psu.edu,
{zzw109,kwilliams,bkp10,bjb40,sps20,hrw115,kbowen}@psu.edu, giles@ist.psu.edu

ABSTRACT

Concept hierarchies have been useful tools for presenting and organizing knowledge. With the rapid growth of online knowledge resources, automatic concept hierarchy extraction is increasingly attractive. Here, we focus on concept extraction from textbooks based on the knowledge in Wikipedia. Given a book, we extract important concepts in each book chapter using Wikipedia as a resource and from this construct a concept hierarchy for that book. We define local and global features that capture both the local relatedness and global coherence embedded in that textbook. In order to evaluate the proposed features and extracted concept hierarchies, we manually construct concept hierarchies for three well used textbooks by labeling important concepts for each book chapter. Experiments show that our proposed local and global features achieve better performance than using only keyphrases to construct the concept hierarchies. Moreover, we observe that incorporating global features can improve the concept ranking precision and reaffirms the global coherence in the book.

Categories and Subject Descriptors

I.2.6 [**Learning**]: Knowledge acquisition; Concept learning; I.7.5 [**Document and Text Processing**]: Document Capture—*Document Analysis*; H.3.3 [**Information Storage And Retrieval**]: Information Search and Retrieval

Keywords

Open education; concept hierarchy; textbooks; Web knowledge;

1. INTRODUCTION

A concept hierarchy is a powerful tool for representing and organizing knowledge; it has been widely used in learning and education [16, 17]. It forms a valuable component for numerous science education tasks, including documenting and exploring concept change [26], knowledge sharing [24, 4, 11] and knowledge acquisition [9, 8]. 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 [21, 14].

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 [25, 6].

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). Based on this, we argue that a useful concept hierarchy should have:

- **Less redundancy in the sense that chapters do not always discuss all of the same concepts**: The concept hierarchy is a possible summary of the book. Thus,

DocEng'15, September 8-11, 2015, Lausanne, Switzerland.
© 2015 ACM. ISBN 978-1-4503-3307-8/15/09 ...$15.00.
http://dx.doi.org/10.1145/2682571.2797062.

information overlap between concepts in different subchapters should be small. For instance, if subchapter 1.1 covers "Gross Domestic Product" in detail, subchapter 2.1 should not cover this concept in detail again.

• **Consistency with other concepts in the same chapter**: Concepts within in a subchapter should be highly correlated to each other. For instance, our concept hierarchy will put "Interest Rate" and "Real Interest Rate" together rather than putting "Interest Rate" and "Unemployment" in the same subchapter.

• **Consistent learning order in that concepts follow each other as with prerequisites**: For each concept, the concept hierarchy should follow the learning order of concepts. Given a concept, prerequisite concepts should be introduced before this concept and subsequent concepts should be introduced after the concept. For example, the concept hierarchy should discuss "Gross Domestic Product" before "Real Gross Domestic Product".

In order to capture the global coherence, *CHEB* utilizes the Wikipedia link graph and page content to estimate the pairwise Wikipedia candidate relatedness and learning order. Corresponding to the three characteristics of an our concept hierarchy, three sets of global features are proposed based on their estimated relatedness and learning order.

To evaluate the quality of the extracted concept hierarchy, we conduct experiments on three well used textbooks. By manually labeling the important concepts for each chapter in the books, we obtain a concept hierarchy for each book. We empirically train the concept hierarchy extractor using the proposed features and perform extraction on the testing data. Our results show that incorporating both local and global features achieves significantly better performance and confirms our definition of global coherence in the book.

To the best of our knowledge, this work represents the first attempt to combine the properties of local relatedness and global coherence to automatically extract concept hierarchies from textbooks. Our the major contributions are:

• Automatic extraction of concept hierarchies from textbooks using Web knowledge.

• Propose three sets of global features, which ensure less redundancy, consistency and appropriate learning order for a concept hierarchy that captures the global coherence embedded in a book.

• Manually build concept hierarchies for three well used books and utilize a Learning-to-Rank approach to train and test our concept hierarchy extractor.

The paper is organized as follows. We first define the **Concept Hierarchy Extraction from Books (CHEB)** approach and introduce its work flow in Section 2. Local and global features are introduced in Section 3. In Section 4, we discuss the data preparation and evaluation metrics. In Section 5 analyzes the experimental results for three well used textbooks and presents an example of the generated concept hierarchy. Related work is in Section 6 followed by conclusion and future work in Section 7.

2. 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.

SYMBOL	DESCRIPTION
B	the input book
N	number of subchapters
tb_i	title of i^{th} subchapter
cb_i	content of i^{th} subchapter
cs_{ip}	p^{th} important concepts in i^{th} subchapter
$\lambda(i)$	chapter number of the i^{th} subchapter
W	domain specific dictionary
w_i	i^{th} Wikipedia concept in the dictionary
$L(w_i, w_j)$	prerequisite relation between w_i and w_j
$I(i, j)$	order of subchapter i and subchapter j
Γ	extracted concept hierarchy

Table 1: Symbol Notation

2.1 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 [21], 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 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.

2.2 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

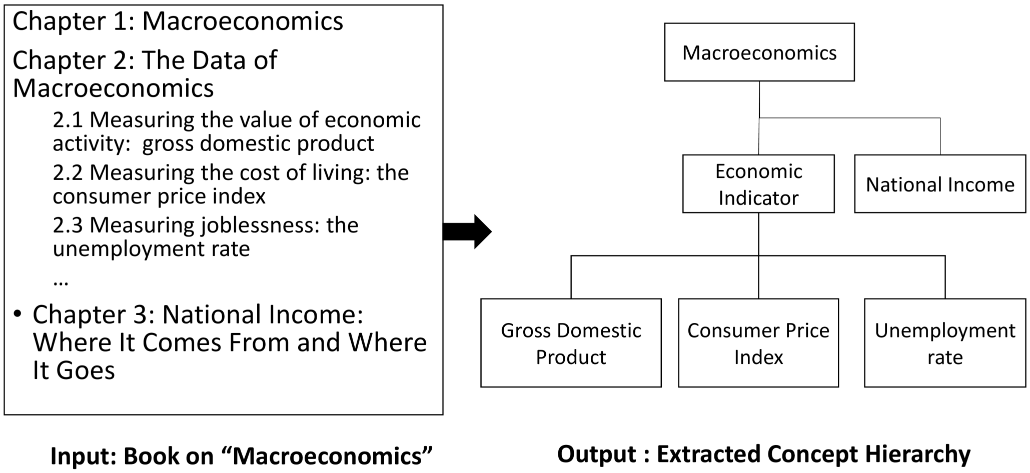

Chapter 1: Macroeconomics
Chapter 2: The Data of Macroeconomics
2.1 Measuring the value of economic activity: gross domestic product
2.2 Measuring the cost of living: the consumer price index
2.3 Measuring joblessness: the unemployment rate
…
• Chapter 3: National Income: Where It Comes From and Where It Goes
Macroeconomics
Economic Indicator
National Income
Gross Domestic Product
Consumer Price Index
Unemployment rate
Input: Book on "Macroeconomics"
Output : Extracted Concept Hierarchy

Figure 1: Example of an extracted concept hierarchy

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:

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

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 a **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:

$$\Gamma^* = \arg\max_{\Gamma}\sum_{i}^{N}\sum_{p\in cs_i}[\Phi(cs_{ip}|tb_i, cb_i) - \Psi(\Gamma) + \Theta(\Gamma) + \gamma(\Gamma)] \qquad (2)$$

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.

$\Psi(\cdot)$ 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. 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* Γ_1, Γ_2, and Γ_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 in an approximate form:

$$\begin{aligned}\Gamma^* \approx \arg\max_{\Gamma}\sum_{i=1}^{N}\sum_{p\in cs_i}[\Phi(cs_{ip}|tb_i, cb_i) - \sum_{cs_{jq}\in\Gamma_1}\Psi(cs_{ip}, cs_{jq})\\ + \sum_{cs_{jq}\in\Gamma_2}\Theta(cs_{ip}, cs_{jq}) + \sum_{cs_{jq}\in\Gamma_3}\gamma(cs_{ip}, cs_{jq})]\end{aligned} \qquad (3)$$

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 (Γ_1 in Eq. 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 Γ_2 and Γ_3 for Θ and γ respectively. Basically, for each candidate concept w_j in the i^{th} subchapter, Γ_2 only includes w_k from subchapter i since we focus on the consistency of concepts within the same subchapter; as for Γ_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 as the following form:

$$\begin{aligned}\Gamma^* \approx \arg\max_{\Gamma} \sum_{i=1}^{N} \sum_{p\in cs_i} [\Phi(cs_{ip}|tb_i, cb_i) \\ - \sum_{CN(j)\neq CN(i)}^{N} \sum_{q=1}^{|cs_j|} \Psi(cs_{ip}, cs_{jq}) + \sum_{q=1}^{|cs_i|} \Theta(cs_{ip}, cs_{iq}) \\ + \sum_{i\neq j}^{N} \sum_{q=1}^{|cs_j|} \gamma(cs_{ip}, cs_{jq})]\end{aligned} \quad (4)$$

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, $\sum_{\lambda(j)\neq\lambda(i)}^{N} \sum_{q=1}^{|cs_j|} \Psi(cs_{ip}, cs_{jq})$ 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. $\sum_{i\neq j} \sum_{q=1}^{|cs_j|} \gamma(cs_{ip}, cs_{jq})$ 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:

$$L(w_i, w_j) = \begin{cases} 1 & if\ w_i\ is\ the\ prerequisite\ concept\ of\ w_j \\ -1 & if\ w_i\ is\ the\ subsequent\ concept\ of\ w_j \\ 0 & otherwise \end{cases} \quad (5)$$

$$I(i, j) = \begin{cases} 1 & if\ i\ is\ a\ subchapter\ before\ j \\ -1 & if\ i\ is\ a\ subchapter\ after\ j \\ 0 & i = j \end{cases} \quad (6)$$

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. 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 Φ, Θ and γ in the optimization function.

3. CONCEPT HIERARCHY EXTRACTION FROM BOOKS

In this section we present our method, *CHEB*, for solving the optimization problem defined in Eq. 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 Φ 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 ω_i is learned using a Support Vector Machine over training data from the constructed data set, described in Section 4.1.

Similarly, the redundancy function Ψ and consistency function Θ 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 γ 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 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.

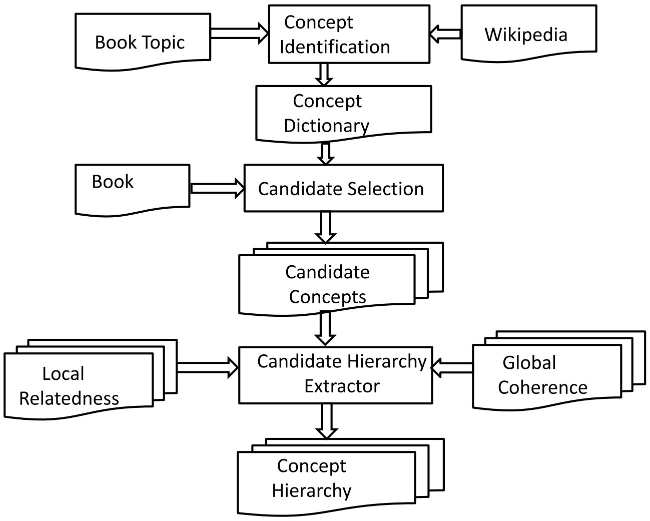


Figure 2: Workflow of the CHEB system

In the following sections, we will describe three modules of *CHEB* as suggested in Fig 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 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 [23]. 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.

3.2 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 consine 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.

3.3 Concept Hierarchy Generation

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

3.3.1 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.

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

3.3.2 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.

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 [28]. 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:

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

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

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

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 *titleMatch* and *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:

$$cosSimCons(cs_{ip}) = \sum_{q=1}^{min(|cs_j|,K)} cosineSim(cs_{ip}, cs_{iq})$$

$$JaccardCons(cs_{ip}) = \sum_{q=1}^{min(|cs_j|,K)} Jaccard(cs_{ip}, cs_{iq})$$

$$semSimCons(cs_{ip}) = \sum_{q=1}^{min(|cs_j|,K)} semSim(cs_{ip}, cs_{iq})$$

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

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 2.2. 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:

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

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

Similarly, we consider only top-K candidates in a subchapter when calculating the learning order features. Eq. 3.3.2 and Eq. 3.3.2 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.

3.3.3 Concept Hierarchy Extractor Training

After generating the features for concept hierarchy extraction, we learn the coefficients for the extractor using SVM^{rank} [12] 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.

4. DATA SETS AND EVALUATION METRICS

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

4.1 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 book) [1], "Principles of macroeconomics" (hereafter, the macroeconomics book) [2], and "Precalculus: Mathematics for calculus" (hereafter, the precalculus book) [3]. 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

[1] Kurose, James. F. (2005). Computer networking: a top-down approach featuring the Internet. Pearson Education India.

[2] Mankiw, N. Gregory.(2014). Principles of macroeconomics. Cengage Learning.

[3] Stewart, James, Lothar Redlin, and Saleem Watson. Precalculus: Mathematics for calculus. Cengage Learning, 2015.

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 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. 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 network book, 7981 for the Macroeconomics book and 11766 for the the Precalculus book.

Candidate Selection The *top-30* Wikipedia candidates are selected using the two features described in Section 3.2 (*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 2 and Figure 3 provide some statistics for the book structures.

4.2 Evaluation Metrics

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

5. 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.

5.1 Baseline Method

SimSeerX [27] 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 [18], 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* [18]. *Maui* begins by identifying candidate keyphrases in the text based on n-grams of words, and then features [18] 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.

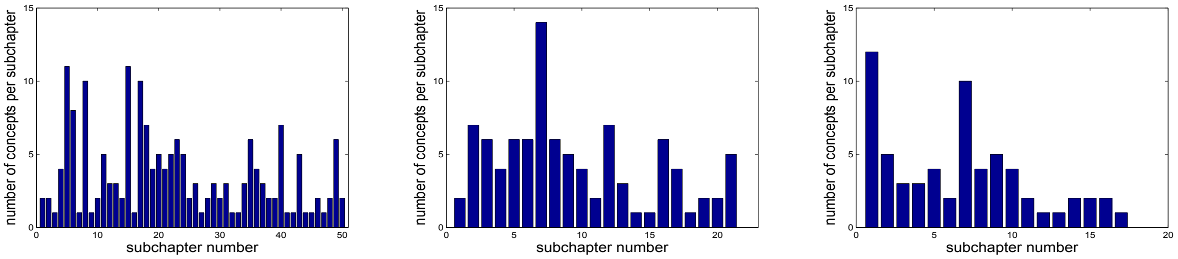
5.2 Book-Level Concept Hierarchy Extraction

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 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

	Computer Network	Macroeconomics	Precalculus
toc depth	3	2	2
# subchapters	50	21	17
avg # important concepts per subchapter	3.6	4.5	4.3
avg # candidate concepts per subchapter	69.933	80.1071	69.2903
avg length of title (words)	3.34	6.19	2.53

Table 2: Physical characteristics of books



(a) Computer network book

(b) Macroeconomics book

(c) Precalculus book

Figure 3: Number of important concepts in each book subchapter

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 2.

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.

5.3 Subchapter-Level Concept Hierarchy Extraction

From the numbers in Table 2, we see the book structures are quite different. In order to capture their different structures, we also conduct experiments on each book separately.

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 out 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).

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.

5.4 Concept Hierarchy Analysis - Computer Network Concept Hierarchy

In this section, we show the concept hierarchy extracted from the computer network book in figure 4 where each rectangle represents a subchapter in the book and the concept extracted from this subchapter. Rectangles with thick borders represent subchapters with length less than 100 words. There was no extraction on these subchapters. These subchapters do not introduce any concepts in details and thus are filtered in the preprocessing step as discussed in Section 4.1. In this hierarchy, the subchapters “’Computer Networks and the Internet” and “Delay, loss and throughput” are such subchapters.

As we can see, our extractor captures most of the important concepts in each subchapter and provides a reasonable concept hierarchy for the computer network domain.

6. RELATED WORK

Our work is primarily related to two areas of research: Wikification [3, 5, 7, 10, 19, 23] and knowledge extraction from education resources [1, 13, 14, 29, 30, 32].

Wikification automatically links terms in the plain text to appropriate Wikipedia articles. Bunescu and Pasca [3] first explored Wikipedia as a resource for detecting and disambiguating named entities 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 [19] performed 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. Semantic relatedness between Wikipedia candidates [7, 10, 20, 23] 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 for the same article should be semantically close to each other. This work also stressed the semantic relatedness between the Wikipedia candidates and optimized the disambiguation results.

Other related research is knowledge extraction from text, course materials and papers [2, 22]. Our focus is primarily on knowledge extraction for educational purposes. Agrawal et al. [1] proposed a method to identify deficient sections and

	Computer Network				Macroeconomics				Precalculus			
	P@1	P@3	P@5	MAP@10	P@1	P@3	P@5	MAP@10	P@1	P@3	P@5	MAP@10
Baseline Method	0.42	0.19	0.16	0.21	0.4	0.33	0.24	0.23	0.23	0.19	0.15	0.18
TitleMatch Feature	0.3	0.13	0.08	0.08	0.5	0.37	0.24	0.27	0.36	0.13	0.08	0.08
CosineSim Feature	0.74	0.48	0.4	0.35	0.57	0.61	0.46	0.33	0.6	0.51	0.41	0.32
Local Features	0.79	0.52	0.43	0.34	0.8	0.52	0.44	0.32	0.8	0.49	0.4	0.34
Global Features	0.38	0.34	0.3	0.3	0.5	0.35	0.32	0.29	0.43	0.3	0.25	0.25
Local+Global Features	0.84	0.52	0.42	0.37	0.83	0.54	0.42	0.34	0.83	0.46	0.39	0.34

Table 4: Subchapter-level experimental results

	P@1	P@3	P@5	MAP@10
Baseline Method	0.4	0.2	0.16	0.29
TitleMatch Feature	0.28	0.12	0.07	0.05
CosineSim Feature	0.74	0.5	0.43	0.36
Local Features	0.76	0.5	0.39	0.35
Global Features	0.45	0.38	0.33	0.25
Local+Global Features	0.8	0.52	0.42	0.36

(a) Ranking precisions on computer network book

	P@1	P@3	P@5	MAP@10
Baseline Method	0.38	0.31	0.24	0.21
TitleMatch Feature	0.57	0.31	0.17	0.12
CosineSim Feature	0.61	0.58	0.54	0.4
Local Features	0.83	0.57	0.46	0.4
Global Features	0.47	0.42	0.31	0.30
Local+Global Features	0.85	0.58	0.45	0.41

(b) Ranking precisions on macroeconomics book

	P@1	P@3	P@5	MAP@10
Baseline Method	0.29	0.23	0.18	0.17
TitleMatch Feature	0.41	0.15	0.09	0.07
CosineSim Feature	0.64	0.52	0.44	0.31
Local Features	0.76	0.56	0.49	0.34
Global Features	0.47	0.42	37.25	0.27
Local+Global Features	0.82	0.51	0.47	0.34

(c) Ranking precisions on precalculus book

Table 3: Book-level experimental results

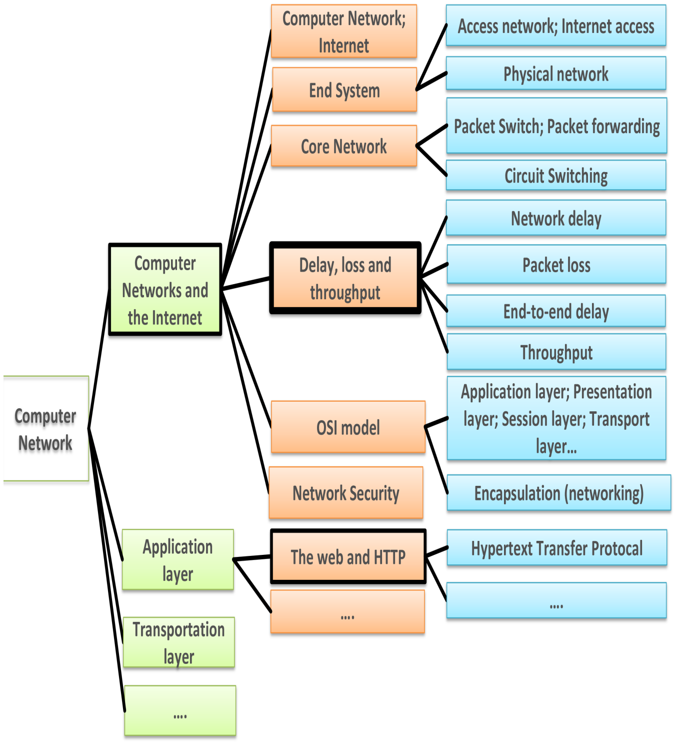

Computer Network
Computer Networks and the Internet
Application layer
Transportation layer
....
Computer Network; Internet
End System
Core Network
Delay, loss and throughput
OSI model
Network Security
The web and HTTP
....
Access network; Internet access
Physical network
Packet Switch; Packet forwarding
Circuit Switching
Network delay
Packet loss
End-to-end delay
Throughput
Application layer; Presentation layer; Session layer; Transport layer...
Encapsulation (networking)
Hypertext Transfer Protocal
....

Figure 4: Part of Concept Hierarchy extracted from the computer network book. Rectangles with thick borders represent subchapters with length less than 100 words and are removed in the pre-processing step(See Section 4.1)

enhance these sections using web knowledge. They used concept dispersion and syntactic complexity to identify deficient sections and augmented these sections using Wikipedia. Informativeness was measured based on semantic similarity between a term's context and the its most featured contexts in Wikipedia [29] and was used to automatically create back-of-the-book indexes [31]. Liang et al. [15] proposed an automated book creation framework which incorporated the method proposed in this work. Recently, prerequisite relationships among courses and papers were derived: Yang et al. proposed a learning-to-rank approach to explore prerequisite relationships among courses and constructed a concept graph based on the relationships [32]. Koutrika et al. generated a reading tree for papers by measuring the generality score of a paper and the overlap between two papers [13].

7. CONCLUSION AND FUTURE WORK

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 consider the local relatedness between the extracted concepts and a book chapter. Moreover, global features ensure that the extracted concept hierarchy is less redundant, more consistent and follows a consistent learning order. To validate the proposed local and global features, we manually construct concept hierarchies for three well used textbooks. Experimental results show that incorporating the global features can improve the ranking precision. Though the data set used is small, the manually created data set is of high quality and precise enough for us to make an attempt to study the global coherence embedded in the books.

To our knowledge this is the first study that utilizes both a local relatedness and global coherence to extract concept hierarchies from books. Future directions would be to construct concept hierarchies for different domains or from multiple books from the same domain. We will also attempt to infer the prerequisites between concepts in book chapters. Another interesting problem is to use domain specific concept hierarchies to build applications for science education, instruction, and learning.

8. ACKNOWLEDGEMENTS

We gratefully acknowledge partial support from NSF and useful comments from the reviewers.

9. REFERENCES

[1] R. Agrawal, S. Gollapudi, A. Kannan, and K. Kenthapadi. Data mining for improving textbooks. *ACM SIGKDD Explorations Newsletter*, 13(2):7–19, 2012.

[2] H. Alani, S. Kim, D. E. Millard, M. J. Weal, W. Hall, P. H. Lewis, and N. R. Shadbolt. Automatic ontology-based knowledge extraction from web documents. *Intelligent Systems, IEEE*, 18(1):14–21, 2003.

[3] R. C. Bunescu and M. Pasca. Using encyclopedic knowledge for named entity disambiguation. In *EACL*, pages 9–16, 2006.

[4] J. W. Coffey, R. R. Hoffman, A. J. Cañas, and K. M. Ford. A concept map-based knowledge modeling approach to expert knowledge sharing. *IKS*, pages 212–217, 2002.

[5] S. Cucerzan. Large-scale named entity disambiguation based on Wikipedia data. In *EMNLP*, 2007.

[6] S. Downes. E-learning 2.0. elearn magazine, 10.2005. *Online http://elearnmag. org/subpage. cfm*, pages 29–1, 2005.

[7] P. Ferragina and U. Scaiella. Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). pages 1625–1628, 2010.

[8] S. E. Gordon, K. A. Schmierer, and R. T. Gill. Conceptual graph analysis: Knowledge acquisition for instructional system design. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 35(3):459–481, 1993.

[9] A. C. Graesser and S. P. Franklin. Quest: A cognitive model of question answering. *Discourse processes*, 13(3):279–303, 1990.

[10] X. Han and J. Zhao. Named entity disambiguation by leveraging wikipedia semantic knowledge. In *CIKM*, pages 215–224. ACM, 2009.

[11] G.-J. Hwang, Y.-R. Shi, and H.-C. Chu. A concept map approach to developing collaborative mindtools for context-aware ubiquitous learning. *British Journal of Educational Technology*, 42(5):778–789, 2011.

[12] T. Joachims. Training linear svms in linear time. In *KDD*, pages 217–226. ACM, 2006.

[13] G. Koutrika, L. Liu, and S. Simske. Generating reading orders over document collections. 2015.

[14] M. Larranaga, A. Conde, I. Calvo, J. A. Elorriaga, and A. Arruarte. Automatic generation of the domain module from electronic textbooks: Method and validation. *Knowledge and Data Engineering, IEEE Transactions on*, 26(1):69–82, 2014.

[15] C. Liang, S. Wang, Z. Wu, K. Williams, B. Pursel, B. Brautigam, S. Saul, H. Williams, K. Bowen, and C. Giles. Bbookx: An automatic book creation framework. In *The ACM Symposium on Document Engineering*, 2015.

[16] K. M. Markham, J. J. Mintzes, and M. G. Jones. The concept map as a research and evaluation tool: Further evidence of validity. *Journal of research in science teaching*, 31(1):91–101, 1994.

[17] J. R. McClure, B. Sonak, and H. K. Suen. Concept map assessment of classroom learning: Reliability, validity, and logistical practicality. *Journal of research in science teaching*, 36(4):475–492, 1999.

[18] O. Medelyan, E. Frank, and I. H. Witten. Human-competitive tagging using automatic keyphrase extraction. In *EMNLP*, pages 1318–1327. Association for Computational Linguistics, 2009.

[19] R. Mihalcea and A. Csomai. Wikify!: linking documents to encyclopedic knowledge. In *CIKM*, pages 233–242. ACM, 2007.

[20] D. Milne and I. H. Witten. Learning to link with wikipedia. In *CIKM*, pages 509–518. ACM, 2008.

[21] A. M. Olney. Extraction of concept maps from textbooks for domain modeling. In *Intelligent Tutoring Systems*, pages 390–392. Springer, 2010.

[22] M. Rajman and R. Besançon. Text mining-knowledge extraction from unstructured textual data. In *Advances in Data Science and Classification*, pages 473–480. Springer, 1998.

[23] L. Ratinov, D. Roth, D. Downey, and M. Anderson. Local and global algorithms for disambiguation to wikipedia. pages 1375–1384, 2011.

[24] W.-M. Roth and A. Roychoudhury. The social construction of scientific concepts or the concept map as conscription device and tool for social thinking in high school science. *Science education*, 76(5):531–57, 1992.

[25] S. H. Usman and I. O. Oyefolahan. Encouraging knowledge sharing using web 2.0 technologies in higher education: A survey. *arXiv preprint arXiv:1406.7437*, 2014.

[26] J. D. Wallace and J. J. Mintzes. The concept map as a research tool: Exploring conceptual change in biology. *Journal of research in science teaching*, 27(10):1033–1052, 1990.

[27] K. Williams, J. Wu, and C. L. Giles. Simseerx: a similar document search engine. In *DocEng'14*, pages 143–146. ACM, 2014.

[28] I. Witten and D. Milne. An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In *AAAI'08*, pages 25–30, 2008.

[29] Z. Wu and C. L. Giles. Measuring term informativeness in context. In *HLT-NAACL*, pages 259–269, 2013.

[30] Z. Wu, Z. Li, P. Mitra, and C. L. Giles. Can back-of-the-book indexes be automatically created? In *CIKM*, pages 1745–1750. ACM, 2013.

[31] Z. Wu, P. Mitra, and C. L. Giles. Table of contents recognition and extraction for heterogeneous book documents. In *ICDAR*, pages 1205–1209. IEEE, 2013.

[32] Y. Yang, H. Liu, J. Carbonell, and W. Ma. Concept graph learning from educational data. In *WSDM*, pages 159–168, 2015.