

Expanding the Web of Knowledge: One Textbook at a Time

Isaac Alpizar-Chacon
Utrecht University
Utrecht, The Netherlands

Sergey Sosnovsky
Utrecht University
Utrecht, The Netherlands

ABSTRACT

Textbooks are educational documents created, structured and formatted in a way that facilitates understanding. Most digital textbooks are released as mere digital copies of their printed counterparts. We present a mechanism that extracts knowledge models from textbooks and enriches their content with additional links (both internal and external). The textbooks essentially become hypertext documents where individual pages are annotated with important concepts in the domain. We also show that extracted models can be automatically connected to the Linked Open Data cloud, which helps further facilitate access, discovery, enrichment, and adaptation of textbook content. Integrating multiple textbooks from the same domain increases the coverage of the composite model while keeping its accuracy relatively high. The overall results of the evaluation show that the proposed approach can generate models of good quality and is applicable across multiple domains.

CCS CONCEPTS

• **Information systems** → **Document representation**; Information extraction; • **Applied computing** → **Document analysis**.

KEYWORDS

Knowledge Extraction; Named Entity Disambiguation; Textbook; DBpedia; Semantic Linking

ACM Reference Format:

Isaac Alpizar-Chacon and Sergey Sosnovsky. 2019. Expanding the Web of Knowledge: One Textbook at a Time. In *30th ACM Conference on Hypertext and Social Media (HT '19)*, September 17–20, 2019, Hof, Germany. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3342220.3343671>

1 INTRODUCTION

Nowadays, a lot of high-quality expository content is accessed online. Textbooks, tutorials, manuals, etc. are available on the WWW for many subjects. Most of them are carefully-written and well-formatted collections of thematic texts that are selected, ordered and structured in a way that facilitates understanding. Textbooks in particular tend to be logical and consistent in their structural organization and formatting elements. These components of a good textbook design are vehicles for improving structure awareness in readers, which is an important factor for helping text comprehension and recall [19]. Unfortunately, digital textbooks are rarely

published as hypertext documents. Even fewer such textbooks are provided with additional tools, components or models that facilitate knowledge discovery within a book, or linking to external knowledge models and/or relevant resources outside of it. Considering how much digital content is available online on virtually any topic, readers of these textbooks could greatly benefit from some level of support when they try to search for or navigate to information on a certain topic.

In our recent work [2], we have presented an approach towards automated extraction of machine-readable domain models from textbooks taking into account their formatting rules and internal structure. We have focused on PDF as the most common textbook representation format; however, the overall approach is also applicable to other formats that are more explicit and coherent in their structural specifications than PDF. In this paper, we take the next step and demonstrate how these models can automatically link textbooks from the same domain and integrate textbooks with a global reference model, such as DBpedia. As a result, textbooks can potentially become ubiquitous sources of knowledge for a growing Linked Open Data initiative¹. Textbooks themselves also transform into collections of documents linked to this Linked Open Data cloud and become available for a rich variety of services that can facilitate their discovery, enrichment, adaptation, etc.

At the end of every well-designed textbook, is a manually created and curated index of important terms. Index creation is a tedious process (e.g., Wiley estimates that "adequate index preparation requires 10-15 hours per 100 typeset pages"²). This process follows elaborate guidelines stipulating index length and style, suggesting what can be good and bad candidates for index terms, advising on how to maintain consistency when creating hierarchical indices, etc. [3]. Every index term is selected by a textbook author or a dedicated human indexer. A result is not just a collection of words, but, essentially, a reference model produced by a domain expert according to a predefined set of rules. Each entry in this model is provided with one or more links to the pages within a textbook. And these pages do not simply mention index entries, but provide meaningful references by either introducing corresponding terms or elaborating them.

To implement the first step of the proposed approach, we have analyzed the guidelines specified by several textbook publishers for creating indices, generalized them into a set of heuristics and implemented a rule-based component that extracts and formally represents term-based knowledge models and their annotations of textbook pages.

Each of these models is not guaranteed to provide high-quality representation of a domain. They can potentially suffer from several drawbacks:

¹<https://lod-cloud.net/>

²<https://authorservices.wiley.com/author-resources/book-authors/prepare-your-manuscript/indexing.html>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

HT '19, September 17–20, 2019, Hof, Germany

© 2019 Association for Computing Machinery.

ACM ISBN 978-1-4503-6885-8/19/09...\$15.00

<https://doi.org/10.1145/3342220.3343671>

- (1) **Subjectivity:** they can contain terms that an author has used for explanatory purposes, but which are only marginally related to an objective picture of the domain semantics.
- (2) **Coverage:** on the other hand, they can miss important terms if a textbook does not cover (enough of) a particular part of a domain.
- (3) **Granularity:** although the authors are recommended to select index terms cohesively, sometimes, the terms in an index can be too detailed, too broad, or inconsistently vary in their granularity.
- (4) **Lack of semantics:** hierarchical indices can help specifying structural relations between sub- and super-entries, "see-also" cross-references can further facilitate linking related terms; however, unfortunately, most indices do not provide input even for these relations.

Nevertheless, at the very least, each extracted index is essentially a formally represented collection of manually selected labels for important notions in the domain and it comes with a manually annotated corpus of text, i.e., it becomes a machine-readable glossary.

To combat the potential problems of subjectivity, low coverage, poor granularity and lack of semantics, glossaries extracted from individual textbooks can be integrated with each other and linked to external models. In this study, we focus on one of the most popular such models - DBpedia³ [5]. It is a machine-understandable knowledge base automatically-extracted from Wikipedia. Currently, DBpedia contains 4.58 million resources and is available in 125 languages. Its information is stored as RDF triples that can be queried using a SPARQL endpoint. The entire model can be also downloaded as a database dump.

The remainder of this paper is organized as follows. Section 2 presents related work. Section 3 provides the details of the proposed approach. Evaluation is described and discussed in Section 4. Finally, Section 5 outlines conclusions and future work.

2 RELATED WORK

2.1 Knowledge Discovery and Terminology Extraction

DBpedia and other knowledge bases (KB) have been used before for exploratory knowledge search. Semantic Wonder Cloud [35] is a graphical tool where the user selects an initial DBpedia resource, and then, the tool displays the ten most similar resources using a hybrid ranking algorithm. The user can continue the exploration by selecting one of the resources to get a new set of related resources. Discovery Hub [30] is an exploratory search engine where the user selects one or several topics of interest, and then, the system selects and ranks on-the-fly a meaningful subset of resources using a spreading activation algorithm and a sampling technique. Medelyan, Witten, & Milne [31] use Wikipedia articles as topics and their titles as controlled terms to discover topics in documents.

Terminology Extraction tasks deal with automated identification of core vocabulary of a specialized domain in un- and semi-structured corpora. Dwarakanath, Ramnani, & Sengupta [12] presented a two-step method for the automatic extraction of glossary terms from software requirements documented in natural language.

³<http://dbpedia.org>

Lopes et al. [27] used three different methods to extract compound terms from a text corpus of the domain of pediatrics.

Our approach is not guided by one initial resource in particular, but instead, it uses a domain-specific glossary extracted from the index sections of textbooks to discover the resources in DBpedia that belong to the same target domain.

2.2 Linking Resources to Entities

The process of identifying the true sense of a resource or linking it a formally defined entity has been researched in closely related fields. Named Entity Recognition (NER) is an approach to identify and classify named entities in free text [18]. Word Sense Disambiguation (WSD) is the task of choosing the right sense or meaning for a word in a context using an exhaustive dictionary [7, 36]. Entity Linking (EL) and Named Entity Disambiguation (NED) are sometimes used indistinctly, where EL refers first to identifying entities and then linking them to a resource in a KB, and NED focuses on the potential ambiguity among several possible candidates in a KB [7]. Typically, lexical ontologies, such as WordNet⁴, have been used in WSD [1, 37], and global knowledge bases such as DBpedia in EL and NED [24, 33]. This paper focuses on NED.

Different approaches and tools to solve NED tasks have been proposed. DBpedia Spotlight [32] receives a text, detects the entities, and applies a Vector Space Model using context around the resources in DBpedia and the input text to disambiguate possible candidates for the entities. Babelify [36] computes semantic signatures for concepts using Wikipedia, and then it links entities to the concepts based on a high-coherence densest subgraph algorithm. TAGME [15] uses a collective agreement between a candidate entity and the possible candidates for other entities in the text for disambiguation. KORE [21] uses keyphrase overlap relatedness to relate entities in the text to pre-extracted entities from Wikipedia. Other approaches use a notion of exclusivity-base relatedness to measure how similar two nodes in a graph are [17], and a common subsumers algorithm between ambiguous entities and close entities that are unambiguous [22] for disambiguation.

Our disambiguation strategy differs from others in several aspects. First, we make use of the fact that textbooks belong to a specific domain, and we use a DBpedia category that matches the domain to create a core set of resources for context, if one is not provided. The DBpedia category is the only manual input data that our algorithm requires, in comparison to keywords [38] and seed entities [34] in other approaches. Second, the list of keywords from textbooks are in most cases abstract concepts and not concrete PLO (Person, Location, and Organization) entities, so the use of individual categories or special formatting patterns for identifying those entities cannot be used [6, 10, 24]. Finally, relying on prior probabilities to get the most popular [20, 33] or authoritative [42] entities is not useful in our case because we deal with domain-specific terms and not general entities. Our disambiguation algorithm uses a DBpedia category to construct a core set of resources, then it extracts the abstracts from the core set to be used as context information, and finally, it uses a similarity measure based on the cosine coefficient to disambiguate between similar candidates for each index term.

⁴<https://wordnet.princeton.edu/>

2.3 Knowledge Models of Textbooks

Research has been carried out on the creation of knowledge models of textbooks. Larrañaga et al. [25] described a system that uses natural language processing techniques, heuristic reasoning, and ontologies for the semiautomatic construction of a representation of the knowledge to be learned from electronic books. Wang et al. [45] have extracted concept hierarchies from textbooks based on their Table of Contents. Finally, Sosnovsky et al. [40] experimented with harvesting topic-based models from HTML textbooks based on the structure of their headings. However, none of those works relied on automatic extraction of knowledge from a textbook index.

2.4 Open Knowledge Bases

Finally, it is important to mention that DBpedia is not the only openly available global knowledge base that can be used to retrieve resources in the Semantic Web [14]. Wikidata [43] started in 2012, and is an open KB that can be read and edited by both humans and computer agents. It stores facts extracted from the structured data of Wikipedia, Wiktionary, and other projects of the Wikimedia foundation. Currently, Wikidata contains more than 63 million items. It provides dumps for its database in different standard formats, and also offers a SPARQL endpoint for direct querying. YAGO⁵ has been developed since 2007. It has imported information from Wikipedia, WordNet, and GeoNames⁶. YAGO stores more than 10 million entities derived from about 120 million facts. Its latest version, YAGO3, is available for download. KBpedia⁷ is another project combining knowledge from several public knowledge bases: Wikipedia, Wikidata, schema.org, DBpedia, GeoNames, OpenCyc⁸, and UMBEL⁹. It includes 55 thousand reference concepts, links to about 32 million entities, and 5 thousand relations and properties. KBpedia provides an online search tool and it is also available for download. Even though our approach uses DBpedia, it could be adapted to work with these knowledge bases.

3 APPROACH

As the first step, our approach processes the content of a textbook, finds its index, extracts all its terms and formally represents them as a machine-readable glossary linked to pages across the textbook. At the next step, we extract DBpedia resources that belong to the target domain of the textbook and link them with relevant concepts from the extracted glossary. Lastly, the glossary is enriched with additional semantic information from DBpedia and creates a bridge between the textbook and the Linked Open Data cloud. The only manual intervention that the approach requires is the selection of a DBpedia category matching the domain of the textbook. We also argue that adding more textbooks from the same domain should result not only in a joint hyperspace of cross-linked textbooks, but also in a more complete and objective model.

Figure 1 presents an outline of the approach, identifying its three main steps: 1) construction of a glossary of terms from textbooks, 2) linking terms to DBpedia, and 3) enrichment of terms with semantic

information. Each step is divided into smaller tasks. The following subsections describe the details.

3.1 Construction of the Glossary

An index section provides a natural source for harvesting important terms in a domain of a textbook. The two main challenges one needs to address when creating a glossary from an index section of a textbook are index extraction and term recognition. The former refers to the task of parsing a textbook and extraction of its index terms, and the latter refers to the identification of the right reading order for each term.

3.1.1 Index Section Parsing. Regardless of the format of the textbook (e.g., PDF or EPUB), the first task for the glossary construction is to parse the textbook, recognize the index section, and retrieve the index terms. Depending on the textbook format and the index section layout (multi-column, hierarchically structured, or multi-line entries), this process can be less or more challenging. We consider this task as a prerequisite for our approach, and it is out of the scope of this paper. Our index extraction process for PDF textbooks is described in [2].

3.1.2 Term Recognition. Once a list of index terms is available, the next step is to identify for each index element its reading *label*, that is, the right order of words in which the element should be read for compound index terms in a hierarchical structure. For example, the hierarchical index entry *distribution <> (gamma, normal)* has three index terms: *distribution*, *distribution gamma* (with possible labels *distribution gamma* and *gamma distribution*) and *distribution normal* (with possible labels *distribution normal* and *normal distribution*). For those terms, *gamma distribution* and *normal distribution* are the right reading labels. In this task, we use a term recognition algorithm that checks the possible labels against the reference text of each term (according to the page numbers of the index entry) to identify the right labels. The algorithm receives as input the candidate labels of an index term and the text (in the form of sentences) of the page where that index term appears in the textbook. The possible labels for a term are created permuting all the parts or lines of a hierarchical index entry.

The algorithm uses the spaCy¹⁰ library for Python to break each candidate and each sentence from the text into noun chunks. A noun chunk is a flat phrase that has a noun as its head. Breaking the text into noun chunks helps discovering meaningful words that are equal in both: the candidates and the sentences, and discarding auxiliary phrase parts like articles. Words in a noun chunk that are not adjectives, verbs, nouns, or proper nouns are discarded. After the candidates and the sentences are broken into noun chunks, the algorithm tries to identify the candidates in each of the sentences by comparing the noun chunks. The lemma forms of words are used to compare two noun chunks. Lemmas forms are the canonical or dictionary forms of words, and their use allows the algorithm to find the candidate labels even if a term is written differently (e.g., in a plural form, or with a different conjugation for verbs). If all the tokens from the noun chunks of the candidate appear in the same order in one sentence, the candidate is returned as the label for the index term. This algorithm is necessary because simple

⁵<https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/downloads/>

⁶<http://www.geonames.org/>

⁷<http://kbpedia.org/>

⁸<https://www.cyc.com/opencyc/>

⁹<http://umbel.org/>

¹⁰<https://spacy.io/>

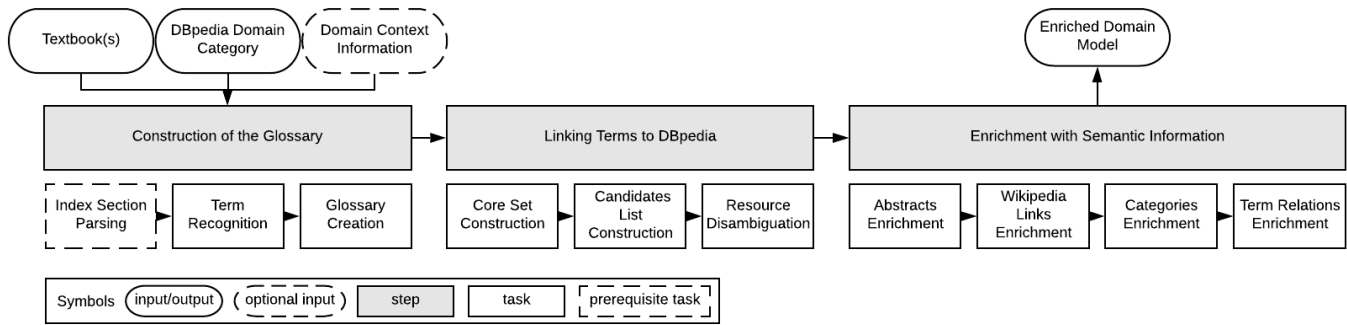


Figure 1: Outline of the implemented approach

textual search is not enough to detect the different variants of index terms in a book. For example, a textual search will fail to find the "Bernoulli distribution" index term in the sentence "Bernoulli and binomial distributions," but our algorithm will be able to detect such an occurrence.

3.1.3 Glossary Creation. The last task of this step is to create a glossary from index terms of a textbook. Such a glossary contains for each index term the main label and a set of alternative labels. If the term recognition algorithm described above confirms a label for a term, it will use it as the main label without alternatives. For terms without an identified label, the main label corresponds to the name as it appears in the index section (read from top to bottom and from left to right) and the alternative labels correspond to all possible label permutations. This glossary is used in the next step of the linking process of the textbook with DBpedia.

3.2 Linking Terms To DBpedia

In this step, the terms from the constructed glossary are linked to DBpedia resources. Finding the right resource for a term involves querying DBpedia and constructing a list of possible candidates, and then using a disambiguation algorithm to choose the best match. Initially, the algorithm selects from the glossary a core set of terms that only have one possible matching resource in DBpedia. Then, the algorithm continues with the terms that have multiple candidate resources. Our disambiguation strategy computes cosine similarity between the extended abstracts of candidate DBpedia entities and the provided context text to find the corresponding resource from the list of candidates. If no initial context information is provided, the abstracts in DBpedia from the resources in the core set are used as context. This step has three tasks.

3.2.1 Core Set Construction. An initial core set of terms includes the terms from the glossary for which only a single DBpedia resource has been found, which is unambiguously the right resource for the term. The resources are found with the help of DBpedia categories. Categories¹¹ correspond to a special type of resources in DBpedia used to classify and group together resources on similar subjects. They are ordered in a broad and non-strict hierarchy (i.e., sub-categories can have multiple super-categories). We claim that

if a term has only one possible candidate resource in DBpedia and that resource belongs to a (sub-)category that is a part of the target DBpedia domain, it is safe to unambiguously link this term to this resource.

Inspired by Mirizzi et al. [34], and Slabbekoorn, Hollink, & Houben [39], our algorithm uses the main DBpedia domain category given as input to find a set of (sub-)categories that belong to the domain of the glossary. Using the `skos:broader` property¹² it is possible to query DBpedia and get all categories corresponding to a domain. For example, 13 subcategories one level down in the hierarchy are obtained when querying the `dbc:Statistics` category. They include `dbc:Statistical_models`, `dbc:Applied_statistics`, `dbc:Statistical_theory`, and `dbc:Sampling_(statistics)`. This process can be done recursively to extract categories from increasingly deeper sub-levels of the hierarchy. As the number of categories increases exponentially, the chance to select marginally relevant or even irrelevant resources for the core set increases fast. After running several experiments, the number of recursive levels has been set to three.

After the set of in-domain categories has been constructed, the algorithm takes the glossary and checks for each term if it belongs to the core set. For each term, the algorithm queries DBpedia to get the corresponding resource. Then, the redirect properties (`dbo:wikipediaRedirects`), if any, are followed. Redirects are used in DBpedia to indicate the final URI of a resource and include synonyms, common misspellings, and acronyms. For example, the resource `dbr:Gaussian_distribution` is redirected to `dbr:Normal_distribution`. After that, the algorithm retrieves the categories of the resource using the `dct:subject` property and checks if any of them belongs to the set of domain categories. The resource is discarded if none of its categories belongs to the domain. Finally, to ensure that there are no other possible resources for the term, the `dbo:wikipediaDisambiguates` property is used. This property groups different resources that have the same or similar names. A term can have other possible resources if one of two possible cases happens. Case 1 is when the found resource has the `dbo:wikipediaDisambiguates` property. For example, the `dbr:Distribution` resource links to 33 resources using the `dbo:wikipediaDisambiguates` property. Case 2 is when the

¹¹<https://en.wikipedia.org/wiki/Help:Category>, <https://wiki.dbpedia.org/services-resources/datasets/dbpedia-datasets#h434-7>

¹²The list of the namespace prefixes used in this paper can be found at <https://dbpedia.org/sparql?help=nsdecl>

resource does not have the `dbo:wikiPageDisambiguates` property, but it is listed with other entities in a page that uses the `dbo:wikiPageDisambiguates` property. For example, the *dbr:Median_lethal_dose* resource appears in the *dbr:MLD* resource through the mentioned property. If none of the two cases occurs, the term is linked to the resource and it is added to the core set.

At the end of this task, if no domain context information was supplied as input, it is constructed. For each term in the core set, the abstract of the linked resource is retrieved using the `dbo:abstract` property. The concatenation of all the abstracts forms the context information for the glossary.

3.2.2 Candidates List Construction. After the core set has been created, the remaining terms in the glossary still need to be linked to resources in DBpedia. The linking process starts by constructing for each term a candidates list of resources. Similarly to the core set construction, for each term the algorithm queries DBpedia and resolves any redirects to get a matching resource. Then, using the two cases for the `dbo:wikiPageDisambiguates` property, as in the previous task, the candidates are identified. In case 1, if the resource has the `dbo:wikiPageDisambiguates` property, the candidates are all the other resources that the property is linking to. If case 1 does not apply, case 2 is checked. If the resource appears in a group with other resources using the `disambiguate` property, all the resources in that group are taken as the candidates. If none of the two cases occurs, there is only one candidate resource for the term. One example of case 2 is the term "mean". DBpedia returns the *dbr:Mean* resource when querying the term. This resource does not have the `dbo:wikiPageDisambiguates` property, but it appears in a group with other resources in the *dbr:Mean_(disambiguation)* resource using the `dbo:wikiPageDisambiguates` property¹³. Figure 2 shows the candidates list for the term "mean" after resolving the disambiguation property.

- `dbr:Mean`
- `dbr:Ethic_mean`
- `dbr:MEAN_(software_bundle)`
- `dbr:Mean_(album)`
- `dbr:Meane`
- `dbr:Meanness`
- `dbr:Means_(disambiguation)`
- `dbr:Mean_(song)`
- `dbr:Mean_(magazine)`

Figure 2: Candidates list for the term "mean"

When each resource candidate is added to the list, its context information is retrieved, to be used in the disambiguation process. Since our disambiguation algorithm uses cosine similarity, the primary context information for a resource is its abstract, which is obtained using the `dbo:abstract` property of the resource. Additionally, based on other approaches [32, 37] that gather context information such as all paragraphs mentioning a candidate resource in Wikipedia or the titles of other resources that link to the candidate, the algorithm expands the abstract of the candidate by adding the abstract of all other resources where its Wikipedia page links to

¹³[http://dbpedia.org/resource/Mean_\(disambiguation\)](http://dbpedia.org/resource/Mean_(disambiguation))

the page of the candidate. We assume that the candidate resources that belong to the same domain as the term will get abstracts also from other resources in the domain, and this will boost the candidate when applying cosine similarity.

3.2.3 Resource Disambiguation. The final task of this step is to apply a disambiguation strategy to choose the best resource in each candidates list. The disambiguation algorithm receives the glossary, candidates list for each term, DBpedia domain category, domain context information, and a threshold. The basic instruction of the algorithm is to compare the extended abstract of each candidate with the domain context information using a similarity function based on cosine similarity, and to select for the term the resource with the highest similarity score that surpasses the minimum threshold. The selection of the resources is done in an incremental manner to first select the resources that are more likely to be the right match.

The **similarity function** computes a similarity score for each candidate resource of a term. First, the standard cosine similarity between the extended abstract of one resource and the domain context information is obtained. Then, the cosine similarity value is modified depending on the names of the candidates and the term. Sometimes candidates for a term are closely related and depending on the extended abstract for each of them, the wrong one can be chosen. To take into account this scenario, two rules are applied. First, if one of the candidates has the same name as the term, the candidate is in the same domain as the glossary, and the cosine similarity value is higher than the threshold, the candidate gets a similarity score of 1. To detect the main domain of the candidate, the algorithm uses the fact that some resources have explicitly encoded the domain for which they belong in the URI. For example, when querying the candidates for the term *Method of moments* for the "statistics" domain, two candidate resources are retrieved: *dbr:Method_of_moments_(statistics)* and *dbr:Method_of_moments_(probability_theory)*. In our example, the *dbr:Method_of_moments_(statistics)* resource gets a similarity value of 1. The second rule is that if only one of the candidates has the same name as the term, that resource does not have its domain in the URI, and its cosine value is higher than the threshold, the candidate gets a similarity score using the following formula: $0.9 + (\text{cosine_similarity}/10)$. The formula seeks to increment the chances of the candidate to be chosen, taking into account the obtained cosine similarity value.

Using the similarity function, the algorithm chooses the right resources from the candidates in an **incremental process**. First, the algorithm goes through the list of candidates of each term, computes their similarity scores and selects only the candidates with a similarity score of 1. Then, it adds the extended abstracts of the chosen resources to the domain context information. After that, the cycle starts again, but now the minimum similarity score for a candidate to be selected is 0.9. The process continues decreasing the minimum similarity score by 0.1 each time until the given threshold is reached. The idea behind our strategy is that by first selecting the resources that get a higher similarity score, the algorithm can expand the domain context information with the extended abstracts from the selected resources, which in turn will help in the next cycles of the algorithm to distinguish the candidates that belong to the same domain from the rest and select the right resource for

each term. At the end of this task, the terms from the glossary with an identified matching DBpedia resource are linked together in our model.

3.3 Enrichment With Semantic Information

The final step of the process is to use the discovered DBpedia resources to extract other semantic information and enrich the terms from the glossary. The enriched glossary can be seen as a rich domain model that contains representative concepts in a domain.

3.3.1 Abstracts Enrichment. Each linked term is enriched with the abstract from its matching DBpedia resource. The abstract from the resources are retrieved querying DBpedia for the `dbo:abstract` property of the resources. Abstracts bring summarized definitions for the index terms of the textbooks.

3.3.2 Wikipedia Links Enrichment. The Wikipedia pages from where the information was extracted to create the resources in DBpedia are used to further enrich the terms. Those links are retrieved using the `foaf:primaryTopic` property of the resources. This enrichment allows to quickly redirect a user who is navigating the domain model to the corresponding Wikipedia page of the concepts (index terms).

3.3.3 Categories Enrichment. DBpedia categories for each linked resource are retrieved using the `dct:subject` property and assigned to linked terms. Adding categories to the terms will allow to have more information to create relations between the terms of the glossary. For example, the categories can be used to create clusters of terms that are related or to find specific pages of textbooks that have concepts in the same category.

3.3.4 Term Relations Enrichment. Finally, we exploit the information about linked Wikipedia pages stored in DBpedia to discover relations among the terms in the glossary. The general idea is that if two resources in DBpedia are linked together the corresponding terms in the glossary are also related. For each term with a linked resource, the algorithm queries DBpedia using the `dbo:wikiPageWikiLink` property to retrieve all other entities that the current resource links to in its corresponding Wikipedia page. Then, for each retrieved resource the algorithm looks in the glossary to find if that resource is linked to a term and if that is the case, a relation is created between the two terms. For example, the term "mean" is linked to the resource `dbr:Mean`, which links to 87 resources. One of those resources is `dbr:Mode_(statistics)`. If in the glossary the term "mode" is linked to `dbr:Mode_(statistics)`, then a relation between "mean" and "mode" is created.

For the moment we only specify that two terms are related in a general manner without detailing the type of relationship. In the future we plan to exploit both the hierarchical structure of index terms and the content in textbooks to be able to discover subtopic hierarchies [4] and more specific relations (e.g., *is-a* relations) [26].

4 EVALUATION

We conducted three evaluations of our approach. In the first and second evaluation, we tested the quality of the first two steps of the approach to see whether the terms are linked to the correct resources in DBpedia. For the last evaluation, we were interested

to examine how adding more textbooks would affect the ability of the approach to connect DBpedia resources from a target domain. All the evaluations were conducted using a local copy of the latest dump of DBpedia (version 2016-10¹⁴).

4.1 Step One: Precision of Term Recognition

Our goal for this evaluation was to find out how many of the index terms extracted from the textbooks are recognized in the pages of the textbooks using our term recognition algorithm. In this evaluation, we compared the number of recognized index terms and reference pages from our algorithm against a baseline. We hypothesized that the use of Part-of-speech tagging to recognize the nouns in the index terms would increase the number of recognized index terms in the pages of the textbook in comparison of textual search.

For this evaluation we used three textbooks. The first two in the statistics domain: *STAT#1* [9], and *STAT#2* [44]. The third book is for information retrieval: *IR#1* [29].

4.1.1 Procedure. First, for each of the textbooks, the index terms and the reference pages were extracted. Index terms are the names of each index entry, and the reference pages are the pages in the textbooks annotated with an index term; these reference pages appear in the form of page numbers next to the index terms in the index section. Then, we applied to the index terms and reference pages our term recognition algorithm (*TRA*) and a baseline strategy for comparison. The *baseline* (*BL*) tried all possible labels of each index term (see section 3.1.2) using simple textual search to find the index term in the reference pages.

4.1.2 Results. Results are presented in Table 1. We report the percentage of index terms where the label (the right reading order) is recognized in at least one reference page (% of recognized index terms) and the percentage of reference pages where the index term is recognized (% of recognized reference pages).

Results show that the performance of our term recognition algorithm is more effective than the baseline in both index term and reference page recognition. In the *STAT#1* textbook the term recognition algorithm increases in 14.1 % the number of recognized index terms and in 34 % the number of pages where the index terms are recognized to the total number. The algorithm achieves the smallest increases in the *IR#1* textbook, 4.5 % and 7.8 % respectively.

Our algorithm can effectively detect index terms that appear fragmented in the text pages (see the "Bernoulli" example in section 3.1.2), but its performance decreases recognizing terms written using synonyms in the pages. For example, in *STAT#1*, the term "Agresti-Coull method" appears as "agresti-coull interval" in its reference page. Another case where our algorithm cannot recognize the index terms is when variants of proper names are used. For example, the term "Billingsley, P." appears only as "Billingsley." Also, it could happen that an index term does not appear in one of its reference pages; the reference is only used to indicate that the topic of that page is related to the index term. We plan to add an external model (e.g., a dictionary) to deal with synonyms, and increase the flexibility of the algorithm to recognize variants of proper names.

¹⁴<https://wiki.dbpedia.org/downloads-2016-10>

Table 1: Evaluation results for the term recognition algorithm

Textbook	# of index terms	# of reference pages	% of recognized index terms	% of recognized reference pages	% of recognized index terms	% of recognized reference pages
			BL	BL	TRA	TRA
STAT#1	539	775	53.9	49.1	68.0	83.1
STAT#2	591	942	82.0	79.2	93.2	90.1
IR#1	624	793	84.7	83.1	89.2	90.9

4.2 Step Two: Precision of Linking

In this second evaluation, the goal was to find out if the index terms with candidate resources in DBpedia are linked to the right resources. For this evaluation, we compared the number of correctly selected resources using our approach against two baselines. We believed that the construction of a core set of resources, and the use of the context information extracted from the resources in that set will perform better at disambiguating and choosing the right resource for each term than approaches that do not use context information.

4.2.1 Procedure. To validate our hypothesis, we tested the precision and accuracy of the second step, the linking of the index terms to DBpedia, using the same three books from the previous evaluation (STAT#1, STAT#2, and IR#1). For STAT#1 and STAT#2, we ran our approach given as input the *dbc:Statistics* category to indicate the domain of the textbooks. For IR#1, the given category was *dbc:Information_retrieval*. In none of the three cases was domain context information given; instead, it was extracted using the created core set. After the "Core Set Construction" and "Candidates List Construction" tasks, there were 43 terms in the core set for STAT#1, 45 for STAT#2, and 39 for IR#1. The number of terms with candidates lists were 129 for STAT#1, 121 for STAT#2, and 291 for IR#1.

For the resource disambiguation task, we used our algorithm and two baselines that do not use context information for disambiguation. Based on Mendes et al. [32], we used the following baselines:

- *Random Baseline (BL1)* selected randomly one of the resources in the candidates list as the right resource. This baseline will show if our algorithm selects the right resource better than chance.
- *Default Sense Baseline (BL2)* selected the resource in the candidates list which is the most referenced resource in Wikipedia from other pages. This baseline will show the impact of using a specific domain for disambiguation in contrast to choosing just the most used resource in the candidates list.

For precision we used the number of **correctly selected** resources from the candidates lists divided by the number of **selected** resources from the candidates lists. Recall was computed dividing the number of **correctly selected** resources by the number of **relevant** resources. For evaluating the selected resources, we manually marked from each of the candidates lists the right resource. The total number of relevant items correspond to the number of terms that belong to the domain of the textbook and had a matching resource in their candidates lists. Since our algorithm uses a minimum

threshold for selecting a candidate as the right resource, we also tested which value would give the best results.

4.2.2 Results. Figures 3, 4, and 5 show for each textbook the precision and recall of our algorithm using different thresholds, and the precision and recall using the two baselines. The values for the baselines are constant since they are not affected by a threshold.

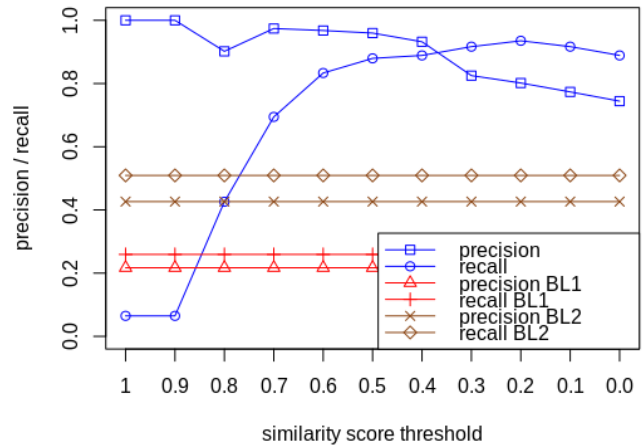


Figure 3: Precision and recall for the linking evaluation of STAT#1 textbook

The performance of the random baseline is the lowest of the three strategies, which confirms that a disambiguation algorithm is needed since chance does not yield good results. The default sense baseline performs better for the two statistics books with a precision and recall up to 0.5. For the information retrieval book, the second baseline achieves a recall of 0.62. Nevertheless, our algorithm obtains the best results. The use of context information achieves better disambiguation than just selecting the most used resources because the terms of the glossary belong to the same domain, and our algorithm exploits this characteristic by constructing the core set of resources. For the statistics books, precision and recall are balanced when the minimum threshold is set anywhere between 0.3 and 0.6. For the information retrieval book, the values 0.4 and 0.5 for the threshold get the best balance for precision and recall. By manipulating the threshold value, we can favor precision or recall depending on the final use of the algorithm. In most use cases, we prefer higher precision (given a reasonable recall) to minimize the number of errors in the model. In general, as the threshold

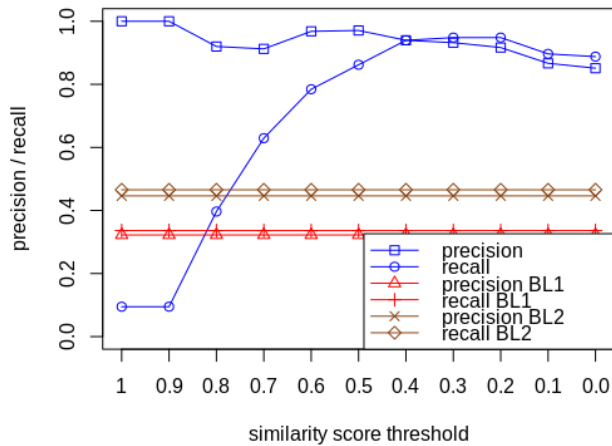


Figure 4: Precision and recall for the linking evaluation of STAT#2 textbook

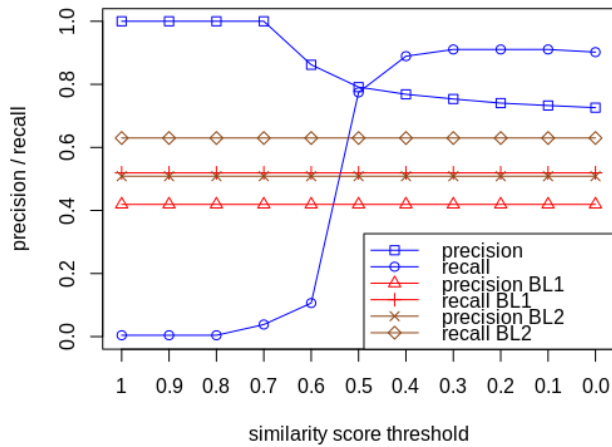


Figure 5: Precision and recall for the linking evaluation of IR#1 textbook

decreases and gets closer to 0, the algorithm selects not only incorrect resources, but also irrelevant ones; they do not belong to any of the related domains of the textbooks.

We also carried out three evaluations given specific context information and not using the one constructed from the core set. The context information was obtained using the textbooks. First, for each index term, we extracted the sentences in which the term appears. All the sentences concatenated were given as the context information. Also, other context information was created by extracting the paragraph, instead of just the sentence, where the index terms appear in the textbooks. The results obtained using the context information from the core set and the context information from the sentences and paragraphs of the textbooks are comparable. The mean absolute error for the precision fluctuates from 0.015 to 0.033 depending on the threshold and the context, and for the recall the mean absolute error fluctuates from 0.022 to 0.156. These evaluations show that the context information extracted from the

core set is as useful for the disambiguation algorithm as the context directly taken from the textbooks.

4.3 Discovery of Resources

The goal of the final evaluation experiment was to determine the effect of adding more textbooks in the glossary construction task and test how many resources the approach will find in DBpedia for a target domain. In this evaluation, we used a ground truth of DBpedia resources in the statistics domain to quantify the resources in the domain that can be discovered. We believed that by integrating glossaries from multiple textbooks in the same domain we can build a consolidated model providing a more complete and objective representation of the domain knowledge. In more practical terms, with every new textbook integrated, the consolidated model should better approximate the ideal model, which is for us the subset of DBpedia resources that belong to the same domain as the textbooks.

4.3.1 Procedure. To test our hypothesis, we used nine introductory statistics textbooks [8, 9, 11, 13, 16, 23, 28, 41, 44]. We wanted to find out how many of the linked resources belong to the statistics domain, and compared that to the total number of resources that belong to the domain. For this evaluation, we defined precision as the number of **linked resources that belong to the domain** divided by the number of **linked resources**. The recall was the number of **linked resources that belong to the domain** divided by the **total number of resources in DBpedia that belong to the domain**. To determine the actual resources in DBpedia that belong to the statistics domain we constructed a ground truth using the ISI Multilingual Glossary of Statistical Terms¹⁵. The ISI Glossary translates more than 3500 statistical terms into 31 languages. It was created by The International Statistical Institute which recognized the need for an affordable multilingual glossary of statistical terms. We used our approach to link the terms from the comprehensive ISI Glossary to DBpedia resources, and after manually checking the obtained linked terms, we got a ground truth of 1042 resources in the statistics domain.

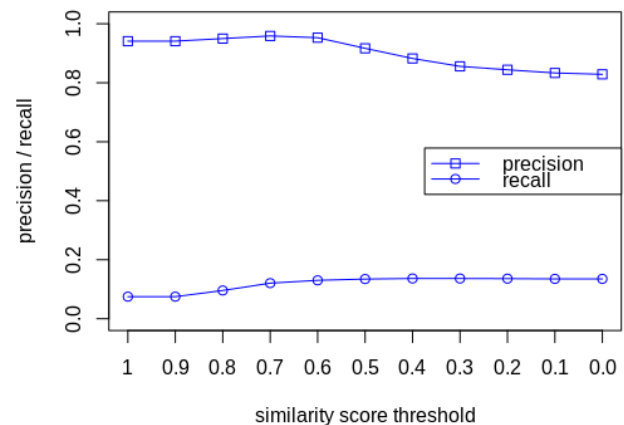


Figure 6: Precision and recall for resource discovery using one textbook (averaged over nine books)

¹⁵<http://isi.cbs.nl/glossary/>

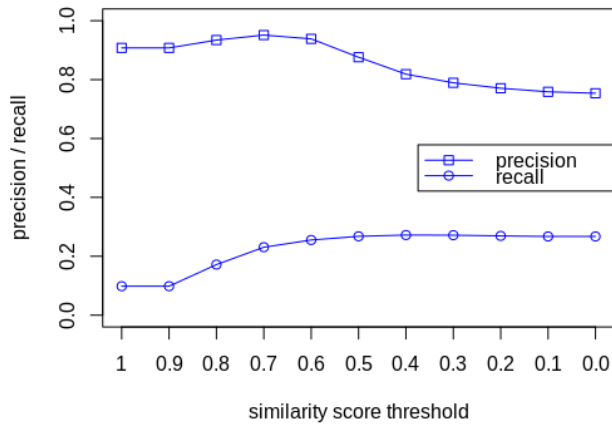


Figure 7: Precision and recall for resource discovery using four textbooks (averaged over two sets of four books)

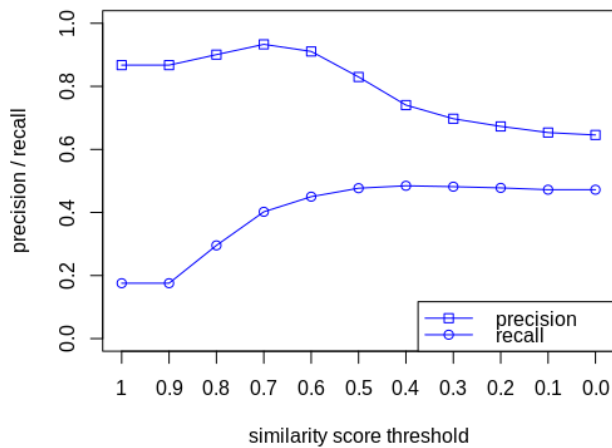


Figure 8: Precision and recall for resource discovery using nine textbooks

4.3.2 *Results.* We evaluated three configurations using the nine textbooks. Figure 6 presents the results for resource discovery using individual textbooks; the results are averaged across all nine textbooks to account for potential differences between them. Figure 7 shows the average discovery of resources of two sets of four textbooks each. Finally, figure 8 shows the discovery of resources when using all of the nine textbooks combined. When using just one textbook for the construction of the glossary, the average precision varies from a maximum of 0.958 to a minimum of 0.828, while the average recall grows from 0.074 to 0.136 (figure 6). The recall is rather low, as a single textbook provides a limited number of index terms (average of 424) of which only a portion has candidate resources in DBpedia (average of 177). The second configuration of four textbooks in two sets has an average of 1626 index terms and an average of 402 index terms with candidates in DBpedia. As shown in figure 7, as the number of terms increases, the recall goes up, reaching up to 0.272, but the precision goes down to the minimum value of 0.753. The final configuration (figure 8) which includes

nine textbooks and 3328 terms (762 with candidate resources) gets a recall of almost 0.5, which means that nearly half of the possible resources in the domain were identified. In all of the three cases, the reduction of the precision as the recall increments is explained by the fact that the index sections of the books contain terms that are not only related to statistics but also to other domains like probability and general mathematics. For example, the resources *dbr:Slope*, *dbr:Law_(stochastic_processes)*, and *dbr:Logarithmic_scale* are linked to terms in the configurations, but those resources do not correspond to terms from the ISI glossary; therefore, they are not part of the ground truth. Filtering linked resources that do not belong to the domain could be accomplished by further exploiting the categories of the resources or by analyzing the graph structure of DBpedia to detect resources that do not belong to the cluster of in-domain resources. The obtained recall values support our hypothesis that it is possible to discover all resources in a domain using textbooks as the source of concepts in a domain.

5 CONCLUSIONS AND FUTURE WORK

In this paper, we have looked into textbooks as a ubiquitous, yet underused source of extractable knowledge models. In particular, we have focused on indices containing manually selected and curated structured glossaries of important terms. When extracted and formally represented these glossaries can become both the backbones of knowledge-driven hyper-structures of the textbooks and the semantic bridges allowing to connect textbooks to one another and to the Linked Open Data sets and knowledge bases such as DBpedia.

Experiments evaluating the proposed approach have demonstrated that even for a single textbook the quality of term recognition and model linking strongly outperforms the baselines. When more textbooks from the same domain are added to generate a composite model, the coverage of model grows significantly. For a threshold value around 0.6-0.7, the recall reaches almost 50%, while precision remains as high as 80-90%.

We plan several directions for future work. The overall approach for textbook model extraction and linking to external models can be further improved. In particular, it is interesting to explore how to maintain high accuracy of the composite model when more and more textbooks are integrated. Some mechanism to enforce consensus among individual glossaries need to be implemented. Another direction for our work is to test this approach across multiple domains. We have observed some differences when comparing Statistics and IR. It would be interesting to discover the limits of applicability of our approach by trying less formal domains such as history or art. Generated models provide unique opportunities to facilitate information access to the content of textbooks or enrich them with external resources connected to existing open knowledge bases. Exploring practical applications of our approach is an exciting task as well. Finally, on a grander scale, this line of research potentially leads towards generation of a global hyperspace of high-quality educational content, where textbooks from the same and relevant domains are linked thematically and the models extracted from these textbooks are built into the global Web of knowledge enabling new generation of information services.

ACKNOWLEDGMENTS

This work was supported by the INTERREG-IVA-GR program (grant 138 GR DeLux 32274) and the Ministry of Science, Technology and Telecommunication of Costa Rica (grant 2-1-4-17-1-021).

REFERENCES

- [1] Eneko Agirre and German Rigau. 1996. Word Sense Disambiguation using Conceptual Density. *Proceedings of the 16th conference on Computational linguistics - 1* (1996), 8.
- [2] Isaac Alpizar-Chacon, Özgün Erensoy, and Sergey Sosnovsky. 2019. Order out of Chaos: Construction of Knowledge Models from PDF Textbooks. In *Proceedings of the 10th International Conference on Knowledge Capture (Submitted) (K-CAP '19)*. ACM, New York, NY, USA.
- [3] Kurt Ament. 2001. *Indexing: a nuts-and-bolts guide for technical writers*. William Andrew.
- [4] Mostafa Bayomi and Séamus Lawless. 2018. C-HTS: A Concept-based Hierarchical Text Segmentation approach. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*.
- [5] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. DBpedia-A crystallization point for the Web of Data. *Web Semantics: science, services and agents on the world wide web* 7, 3 (2009), 154–165.
- [6] Wissem Bouarroudj and Zizette Boufaïda. 2018. A Candidate Generation Algorithm for Named Entities Disambiguation Using DBpedia. Springer, Cham, 712–721.
- [7] Angel X. Chang, Valentin I. Spitzkovsky, Christopher D. Manning, and Eneko Agirre. 2016. A comparison of Named-Entity Disambiguation and Word Sense Disambiguation. *Lrec* (2016), 860–867.
- [8] Peter Dalggaard. 2011. *Introductory statistics with R*. Lightning Source UK Ltd.
- [9] Frederik M. Dekking, Cor Kraaikamp, Hendrik P. Lopuhaä, and Ludolf E. Meester. 2005. *A modern introduction to probability and statistics: understanding why and how*. Springer.
- [10] Gianluca Demartini, Djelle Eddine Difallah, and Philippe Cudré-Mauroux. 2013. Large-scale linked data integration using probabilistic reasoning and crowdsourcing. *The VLDB Journal* 22, 5 (oct 2013), 665–687.
- [11] Jay L. Devore and Kenneth N. Berk. 2012. *Modern Mathematical Statistics with Applications*. Springer.
- [12] Anurag Dwarakanath, Roshni R. Ramnani, and Shubhashis Sengupta. 2013. Automatic extraction of glossary terms from natural language requirements. In *2013 21st IEEE International Requirements Engineering Conference (RE)*. IEEE, 314–319.
- [13] Michael Havbro. Faber. 2012. *Statistics and probability theory: in pursuit of engineering decision support*. Springer Verlag.
- [14] Michael Färber, Basil Ell, Carsten Menne, and Achim Rettinger. 2015. A comparative survey of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web Journal* 1, 1 (2015), 1–5.
- [15] Paolo Ferragina and Ugo Scaïella. 2012. Fast and Accurate Annotation of Short Texts with Wikipedia Pages. *IEEE Software* 29, 1 (2012), 70–75. arXiv:1006.3498
- [16] Michael O. Finkelstein. 2009. *Basic concepts of probability and statistics in the law*. Springer.
- [17] Silvia Giannini, Simona Colucci, Francesco M. Donini, and Eugenio Di Sciascio. 2015. A Logic-Based Approach to Named-Entity Disambiguation in the Web of Data. Springer, Cham, 367–380.
- [18] Younggyun Hahm, Jungyeul Park, Kyungtae Lim, Youngsik Kim, Dosam Hwang, and Key-Sun Choi. 2014. Named Entity Corpus Construction using Wikipedia and DBpedia Ontology. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)* (2014), 2565–2569.
- [19] Kendra M. Hall, Brenda L. Sabey, and Michelle McClellan. 2005. Expository text comprehension: Helping primary-grade teachers use expository texts to full advantage. *Reading psychology* 26, 3 (2005), 211–234.
- [20] Xianpei Han and Le Sun. 2011. A generative entity-mention model for linking entities with knowledge base. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 945–954.
- [21] Johannes Hoffart, Stephan Seufert, Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. 2012. KORE: keyphrase overlap relatedness for entity disambiguation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 545–554.
- [22] Ioana Hulpus, Narumol Prangnawarat, and Conor Hayes. 2015. Path-Based Semantic Relatedness on Linked Data and Its Use to Word and Entity Disambiguation. In *Proceedings of the 14th International Semantic Web Conference (ISWC 2015)*, Vol. 9366. 442–457.
- [23] Hans-Michael Kaltenbach. 2012. *A concise guide to statistics*. Springer.
- [24] Georgi Kobilarov, Tom Scott, Yves Raimond, Silver Oliver, Chris Sizemore, Michael Smethurst, Christian Bizer, and Robert Lee. 2009. Media Meets Semantic Web - How the BBC Uses DBpedia and Linked Data to Make Connections. Springer, Berlin, Heidelberg, 723–737.
- [25] Mikel Larrañaga, Angel Conde, Inaki Calvo, Jon A. Elorriaga, and Ana Arruarte. 2014. Automatic Generation of the Domain Module from Electronic Textbooks: Method and Validation. *IEEE Trans. on Knowl. and Data Eng.* 26, 1 (Jan. 2014), 69–82.
- [26] Mikel Larrañaga, Urko Rueda, Jon A. Elorriaga, and Ana Arruarte. 2004. Acquisition of the Domain Structure from Document Indexes Using Heuristic Reasoning. In *Intelligent Tutoring Systems*, James C. Lester, Rosa Maria Vicari, and Fábio Paraguaçu (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 175–186.
- [27] Lucelene Lopes, Renata Vieira, Maria José Finatto, and Daniel Martins. 2010. Extracting compound terms from domain corpora. *Journal of the Brazilian Computer Society* 16, 4 (01 Nov 2010), 247–259.
- [28] Birger Madsen. 2011. *Statistics for Non-Statisticians*. Springer.
- [29] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2009. *An Introduction to Information Retrieval*. Cambridge University Press.
- [30] Nicolas Marie, Fabien Gandon, Myriam Ribière, and Florentin Rodio. 2013. Discovery hub: on-the-fly linked data exploratory search. In *Proceedings of the 9th international conference on semantic systems*. ACM, 17–24.
- [31] Olena Medelyan, Ian H. Witten, and David Milne. 2008. Topic indexing with Wikipedia. In *Proceedings of the AAAI WikiAI workshop*, Vol. 1. 19–24.
- [32] Pablo N Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. 2011. DBpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th international conference on semantic systems*. ACM, 1–8.
- [33] David Milne and Ian H. Witten. 2008. Learning to link with wikipedia. In *Proceedings of the 17th ACM conference on Information and knowledge management*. ACM, 509–518.
- [34] Roberto Mirizzi, Azzurra Ragone, Tommaso Di Noia, and Eugenio Di Sciascio. 2010. Ranking the linked data: The case of DBpedia. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 6189 LNCS. Springer, Berlin, Heidelberg, 337–354.
- [35] Roberto Mirizzi, Azzurra Ragone, Tommaso Di Noia, and Eugenio Di Sciascio. 2010. Semantic wonder cloud: exploratory search in DBpedia. In *International Conference on Web Engineering*. Springer, 138–149.
- [36] Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity Linking meets Word Sense Disambiguation: a Unified Approach. *Transactions of the Association for Computational Linguistics (TACL)* 2 (2014), 231–244.
- [37] Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence* 193 (2012), 217–250.
- [38] Oscar Rodriguez Rocha, Catherine Faron Zucker, and Alain Giboin. 2018. Extraction of Relevant Resources and Questions from DBpedia to Automatically Generate Quizzes on Specific Domains. Springer, Cham, 380–385.
- [39] Kristian Slabbekoorn, Laura Hollink, and Geert-Jan Houben. 2012. Domain-Aware Ontology Matching. Springer, Berlin, Heidelberg, 542–558.
- [40] Sergey Sosnovsky, I-Han Hsiao, and Peter Brusilovsky. 2012. Adaptation "in the Wild": ontology-based personalization of open-corpus learning material. In *European Conference on Technology Enhanced Learning*. Springer, 425–431.
- [41] Jan Ubøe. 2017. *Introductory statistics for business and economics: theory, exercises and solutions*. Springer International Publishing AG.
- [42] Ricardo Usbeck, Axel Cyrille Ngonga Ngomo, Michael Röder, Daniel Gerber, Sandro Athaide Coelho, Sören Auer, and Andreas Both. 2014. AGDISTIS - Graph-based disambiguation of named entities using linked data. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 8796. 457–471.
- [43] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. *Commun. ACM* 57, 10 (Sept. 2014), 78–85.
- [44] Ronald E. Walpole, Raymond H. Myers, Sharon L. Myers, and Keying Ye. 2012. *Probability & statistics for engineers & scientists*. Prentice Hall.
- [45] Shuting Wang, Chen Liang, Zhaohui Wu, Kyle Williams, Bart Pursel, Benjamin Brautigam, Sherwyn Saul, Hannah Williams, Kyle Bowen, and C Lee Giles. 2015. Concept hierarchy extraction from textbooks. In *Proceedings of the 2015 ACM Symposium on Document Engineering*. ACM, 147–156.