Harnessing Textbooks for High-Quality Labeled Data: An Approach to Automatic Keyword Extraction

Lorenzo Pozzi[1], Isaac Alpizar-Chacon[2] and Sergey Sosnovsky[2]

[1]Piazza Copernico, Rome, Italy
[2]Utrecht University, Utrecht, The Netherlands

Abstract

As textbooks evolve into digital platforms, they open a world of opportunities for Artificial Intelligence in Education (AIED) research. This paper delves into the novel use of textbooks as a source of high-quality labeled data for automatic keyword extraction, demonstrating an affordable and efficient alternative to traditional methods. By utilizing the wealth of structured information provided in textbooks, we propose a methodology for annotating corpora across diverse domains, circumventing the costly and time-consuming process of manual data annotation. Our research presents a deep learning model based on Bidirectional Encoder Representations from Transformers (BERT) fine-tuned on this newly labeled dataset. This model is applied to keyword extraction tasks, with the model's performance surpassing established baselines. We further analyze the transformation of BERT's embedding space before and after the fine-tuning phase, illuminating how the model adapts to specific domain goals. Our findings substantiate textbooks as a resource-rich, untapped well of high-quality labeled data, underpinning their significant role in the AIED research landscape.

Keywords

textbooks, labeled data, automatic keyword extraction, BERT fine-tuning

1. Introduction

As educational landscapes continue to shift towards digital platforms, textbooks have become a rich source of structured information and present a unique opportunity for Artificial Intelligence in Education (AIED) research. This study delves into a novel approach of utilizing textbooks as a source of high-quality labeled data for automatic keyword extraction, providing a cost-effective and efficient alternative to traditional, manual data annotation methods. The fundamental premise of this research is the transformation of textbook content into labeled data, thereby creating a methodology for annotating corpora across diverse domains. To facilitate this transformation, an extensive set of rules is developed to capture common conventions and guidelines for textbook formatting, structuring, and organization.

Automatic Keyword Extraction (AKE) concerns the identification of representative words or phrases, also known as keyword or keyphrase[1], to reduce the complexity of natural language

iTextbooks'23: Fifth Workshop on Intelligent Textbooks, July 03, 2023, Tokio, Japan
lorenzopozzi17@yahoo.it (L. Pozzi); i.alpizarchacon@uu.nl (I. Alpizar-Chacon); s.a.sosnovsky@uu.nl (S. Sosnovsky)
0000-0002-6931-9787 (I. Alpizar-Chacon); 0000-0001-8023-1770 (S. Sosnovsky)

[1]To lighten the use of terminology, "keyphrase" will generally refer to expressions containing at least one word.

and condense the meaning of a passage into fewer terms. Many Natural Language Processing (NLP) applications, such as text classification, document clustering, information mining, or web search, often require to efficiently encode only the essential information via keyphrases, which aid the processing of larger amounts of documents with fewer resources.

In the landscape of AKE, three major families of models have been utilized: *unsupervised*, *supervised*, and *deep learning* approaches. A good deal of the most recent research has focused on this last category. More specifically, large-scale pre-trained Language Models (LMs) that rely on the self-attention mechanism, also known as Transformers [1], have been studied in a large body of literature, demonstrating superior effectiveness than previous approaches. Despite the dominance of Transformers in the field, their hunger for data still represents a restrictive factor: the existence of an ad-hoc corpus is an inevitable prerequisite to match before being able to train any model on the desired task. This bound is particularly consequential when it comes to domain-specific applications. In contrast to the more diffused general-purpose LMs, which are trained to have a broad understanding of language, domain-specific LMs [2, 3, 4, 5] address a particular application domain, recognizing terms that general NLP approaches fail to capture. As an example, if working in the field of Statistics, an algorithm should identify the acronym GLM (Generalized Liner Model), a term strongly related to the statistical domain.

To address the aforementioned challenges, this paper leverages the potential of textbooks to generate high-quality labeled data for domain-specific keyword extraction. Textbooks, in their structured and comprehensive nature, encompass extensive domain-specific knowledge, terminology, and hierarchical structures, making them an ideal data source for this purpose. In the rest of the paper, we describe our approach to create labeled datasets from textbooks and two experiments designed to evaluate the effectiveness of our proposed methodology. The findings from these experiments support the argument for textbooks as a rich, yet largely untapped, resource for labeled data.

2. Extraction of Textbook Models

Our annotation approach is built upon academic textbooks. We have developed a workflow for the automated extraction of textbook knowledge models [6, 7, 8, 9]. The workflow uses an extensive set of rules that capture common conventions and guidelines for textbook formatting, structuring, and organization. The textbook's structure, content, and domain terms are extracted. Structural information contains the list of chapters and subchapters of the textbook. The textbook's content is represented in a structured way (words, lines, text fragments, pages, and sections). Lastly, the domain terms are extracted from the book index, which contains the terminology used in the textbook and the domain. Each term is identified on each referenced page using a term recognition algorithm [6]. Later in the workflow, the domain terms are used as a bridge to link the textbooks to an external knowledge graph. Specifically, domain terms are matched to corresponding entities in DBpedia[2]—a publicly available knowledge graph based on Wikipedia. As the next step, terms from multiple textbooks are integrated into a single model to get better coverage of the target domain and to discover synonyms, which are found with the help of DBpedia [10]. Additionally, terms are categorized according to

[2] https://www.dbpedia.org/

their relevance to the target domain (domain-specificity) using four classes: *core-domain* (the most important concepts in the domain), *in-domain* (other concepts in the domain), *related-domain* (concepts form related domains), and *out-of-domain* (unrelated concepts). Finally, all the extracted knowledge is serialized as a descriptive XML file using the Text Encoding Initiative[3] to produce a textbook knowledge model.

3. Dataset Construction

3.1. Dataset Annotation

We leverage the indexes that are typically associated with textbooks to extract domain-specific knowledge necessary for labeling the dataset. Book indexes were first introduced as a form of navigational tool to help readers orient in such documents. In them, *main headings* and *subheadings* are the access point used to search for information. For instance, in a textbook about statistics, an index entry could be: "Bernoulli distribution, 11". Meaning that the term "Bernoulli distribution" can be found on page 11 of the book. Thus, the annotation is performed by identifying all the instances of the index headings in the body of the books.

We believe textbooks to be a valid resource for the annotation of AKE datasets due to three main reasons. First, the production of indexes is traditionally entrusted to professionals who are conversant in the field for which they are hired. This expertise guarantees the quality of the produced indexes and the corresponding index terms. Second, while working on a book, indexers always consider the so-called *metatopic*, the central matter discussed. Consequently, the great majority of the index entries represent concepts that are related to the target domain. Lastly, the quantity and variety of textbooks available online allow us to generate enough data to train deep architecture in numerous fields.

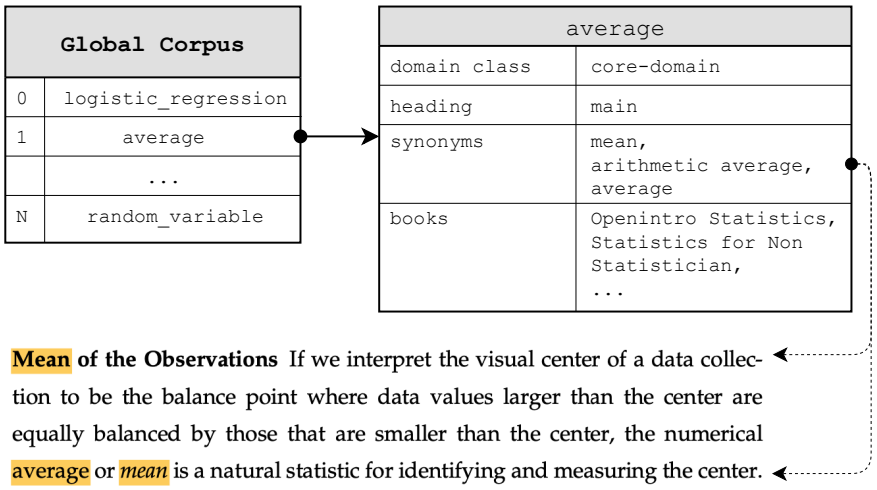

Figure 1: In the GC, each index term is associated with four properties. Among these, the `synonyms` are used to annotate the body of the books.

[3] https://tei-c.org/

The data collection proceeds as follows. Textbook models are extracted using the approach described in §2. From the models, the chapter headings, paragraphs, and indexes are used. The index entries are then tabulated in a common repository that we refer to as Global Corpus (GC). Each entry is also associated with four "properties" (extracted from the textbook models): (i) a domain-specificity class; (ii) an attribute declaring if the term is a *main heading* or *subheading* depending on how it was inserted in the index; (iii) a list of synonyms; (iv) the books in which at least an occurrence of the index term is recognized. Figure 1 illustrates GC structure and how paragraphs in the books are labeled: once a passage is lowercased and lemmarized, the annotation works with an exact match comparison between the text and the terms included in `synonyms`. It is to be heeded that the instances of an index term are identified in all the documents in the corpus, regardless of whether it appears in the corresponding indexes. This is to ensure coherence across the textbooks.

3.2. Dataset Preprocessing

Before starting to annotate the textbooks, the GC and all the paragraphs go through a series of filters to clean the corpus of irrelevant or noisy data.

3.2.1. Index Terms Filtering

Textbooks are not solely meant to be informative but also didactical. Thus, the nature of an index entry varies based on the didactical function that it is supposed to accomplish. Indeed, most index terms concern facts and notions linked to the metatopic and therefore define those concepts that are relevant to the central subject. However, some may refer to examples, exercises, or case studies included by the authors as means of communication to clarify a specific point; others again may point to non-textual elements, such as tables, graphs, formulas, and pseudo-code fragments, that the indexer thinks to be relevant for the reader.

In order to maintain a topical coherence among the terms included in the GC, we made three necessary assumptions: (1) only notional entries that are relevant to the target domain are considered; (2) entries indicated as *out-of-domain* in `domain class` are discarded; (3) subheadings are included only if they present a domain-specificity class.

Notional entries are automatically considered to be domain-related if classified as *core-domain*, *in-domain*, and *related-domain* in the `domain specificity` field. Unfortunately, we were not able to assign all the entries to a category. For unclassified index terms, it was necessary to check them manually to determine which ones were related to the target domain. This procedure was conducted trying to respect as closely as possible the approach adopted by professional indexers. Consulted by experts in the field, we relied on a diversified group of sources of information, i.e. Wikipedia web pages, ISI glossary[4], and other statistical textbooks, to conclude if a term was meaningful or not for the target domain.

As regards subheadings, we opted to discard them if devoid of domain specificity class. After analyzing the available data, we noticed the rate of unclassified index subentries to be higher compared to main headings. Leaving such entries would have meant significantly increasing the number of entries to manually check.

[4] https://www.isi-web.org/isi.cbs.nl/glossary/

In addition to these assumptions, ambiguous entries were also filtered. On occasions, index terms lacking a proper modifier may be associable with more domains if taken out of context. An example is the word "process". A process is quite a generic term that can be found in statistical textbooks, e.g. in "stochastic process", as well as in documents not necessarily related to Statistics. From biochemistry, "apoptosis process" is an example.

3.2.2. Paragraphs Filtering

In parsing the PDFs, non-textual elements such as graphs, images, tables, or mathematical formulas might not be correctly recognized and therefore generate noise sequences of characters. To mitigate this effect and maintain consistent quality across the paragraphs, four filters were applied. First, short paragraphs were discarded to maintain a good quality of contextualized word embedding. Then, paragraphs with a high ratio of characters are also removed. For example, the string "p(a ∪ b) = p(b) + p(a ∩ bc)" is an outlier. Third, paragraphs with a high ratio of digits are discarded. Finally, paragraphs with a high ratio of special characters, such as æ, ©, or μ, are also removed.

In all four filters, thresholds are used. Depending on the focus of the textbook, these four parameters may vary significantly. For example, in the specific case of Statistics, digits and special characters (e.g. greek letters or mathematical operators) are more frequent than in other domains. On the contrary, in History books, the latter is rarely found but digits may still be consistently present in the form of dates. For this case study, we opted for the following values: α, β, γ, δ = {3, 0.30, 0.40, 0.05}.

3.3. Dataset Specifics

In the present research, we investigated the statistical domain. Nine textbooks [5] focusing on Statistics were included in our dataset that we refer to as StatCorpus.

4. Methods

4.1. Problem Statement

Suppose a book is defined by a sequence of *n* text units $\mathcal{U}^{(i)}$, such that $\mathcal{B} = \{\mathcal{U}^{(1)}, ..., \mathcal{U}^{(n)}\}$, where each unit is initially identifiable as a paragraph and the keyphrases to be identified in that span of text. Therefore, $\mathcal{U}^{(i)}$ contains a set of *m* tuples: $\mathcal{U}^{(i)} = \{(\mathcal{X}^{(1)}, \mathcal{Y}^{(1)}), ..., (\mathcal{X}^{(m)}, \mathcal{Y}^{(m)})\}$ where $\mathcal{X}^{(i)}$ is a list of words and $\mathcal{Y}^{(i)}$ is a list of terms. The final goal is to obtain a model able to consistently identify the terms $\mathcal{Y}^{(i)}$ that are in $\mathcal{X}^{(i)}$. It is worth noting that not all the paragraphs may contain a keyword.

In the dataset, each pair $\left(\mathcal{X}^{(i)}, \mathcal{Y}^{(i)}\right)$ is assigned with the heading title of the chapter or subchapter where it occurs. Therefore, there is a triplet $\mathcal{U}^{(i)} = \left(\mathcal{X}^{(i)}, \mathcal{Y}^{(i)}, \mathcal{Z}^{(i)}\right)$ for each row in the corpus.

[5] A concise guide to statistics, A Modern Introduction to Probability and Statistics, Modern mathematical statistics with applications, OpenIntro statistics, Statistics and probability theory, Statistics for non-statisticians, Probability and statistics for engineers and scientists, Statistics for scientists and engineers, and Introductory statistics with R

4.2. Model Definition

Before feeding the paragraphs into BERT, each heading was prepended to the corresponding passage to form the following composite query: $\mathcal{Q}^{(i)} = [\,[CLS]\ \mathcal{Z}^{(i)}\ [SEP]\ \mathcal{X}^{(i)}]\,]$ where [CLS] is the *classification token* and [SEP] indicates a *separation token* interposed between the heading and the passage to inform the model of the two different elements in the input sequence. Given that chapter titles and subtitles are often used to summarize the focus of the section in a few words, we believe the model benefits from this additional information, actively looking at the heading to better recognize the importance of candidate keywords for the encoded passage.

Once the query is passed to BERT's encoding layer, natural language is first tokenized and then univocally converted into a sequence of vector embeddings $\boldsymbol{e} = \{e_1, e_2, ..., e_N\}$, so that $f(\boldsymbol{tk}_j) = \boldsymbol{e}_j$, where $\boldsymbol{e} \in \mathbb{R}^{N \times d_{BERT}}$ and $\boldsymbol{tk}_j$ correspondents to the j-th token from $\mathcal{Q}^{(i)}$.

It is relevant to notice that BERT has a limited input capacity so that $N \in [1, 512]$. When the concatenation of heading and paragraph exceeded this limit, the sequence was truncated to respect the length limit.

Concluding this step, $\boldsymbol{e}$ passes through the encoder. The three inputs of the multi-head self-attention layer are query matrix, key matrix, and value matrix from left to right. The vector generated by the last layer of BERT will be referred to as $\boldsymbol{h}$ from now on. In the process, the embeddings go through two sub-layers. The first is a multi-head self-attention mechanism, responsible for the attention vector that represents how much each word in the sequence has to pay attention at the other ones, and the second is a standard fully connected feed-forward network. Both are combined with a layer normalization and a residual connection.

The contextualized word vectors generated from BERT continue into a Bidirectional Long-Short Term Memory (BiLSTM) and a Linear Classifier stacked on top of it. A BiLSTM is a composite model consisting of two LSTM modules: one taking the input in a forward direction and the other in a backward direction. The sequence generated by the last layer of the BiLSTM is here denoted $\boldsymbol{m} = \{m_1, m_2, ..., m_N\}$ such that $\boldsymbol{m} \in \mathbb{R}^{N \times 2d_{BiLSTM}}$.

Lastly, $\boldsymbol{m}$ is fed into the Linear Classifier that computes a score $\boldsymbol{y} \in \mathbb{R}$ reflecting the semantic closeness of each token $\boldsymbol{tk}_j$ in the initial sequence respect to the target domain.

Once the classification is concluded, the model returns a collection of candidate keyphrases. Before comparing against the ground truth, the candidates are further filtered using three *principles* formulated to align the extracted keywords to the entries produced by professional indexers.

Modifiers Principle: valid terms should not be modifiers. This comes directly from the assumption that modifiers should not be found as independent index entries [11] in order to avoid redundancies in the index.

Structural Principle: valid terms should reflect the syntactical structures used by professional indexers' guidelines. Based on an analysis of the compositional patterns used by indexers, we elaborated the following rules: (i) single words predictions are limited to nouns, proper nouns, and verbs; (ii) taking into account dependency tags, the first or the last word in an extracted phrase must not be coordinating conjunction, punctuation, determiner, preposition or subordinating conjunction, and adposition.

Completeness Principle: incomplete terms truncated by BERT's tokenization algorithm [12] were removed from the pool of candidate keywords.

5. Experiments

Two experiments are performed to evaluate the proposed AKE architecture, here referred to as IndexBERT. For both experiments, cross-validation was used. In each iteration, the test set included only one textbook, while the rest of the documents formed the training set. The fine-tuning phase was thus repeated nine times in total, and the results averaged across the runs.

5.1. Experiment 1: Keyphrase Extraction

The first experiment evaluates IndexBERT on a keyphrase extraction task. The main assumption is that satisfactory performance reflects both the dataset quality and the model suitability for domain-specific applications. Results are reported in terms of precision and recall. However, given that the dataset was built from book indexes, we did not compute the metrics on the individual keywords but based on index terms in the GC. Specifically, recall is calculated against those index terms included in the textbook/s being part of the test set. This is because some index terms could be missing from the test textbook/s and therefore be unretrievable. We call this metric *Local Recall*. On the other hand, precision is meant to inform about the correctness of a prediction regardless of the indexes being part of the textbook/s in the test set. The ground truth is therefore a term bank that follows the same structure as the GC but is expanded with additional index terms to minimize the number of mismatches. This extended-term bank, which we name Global Corpus Plus, shortened GC+, counts on two more books with their index terms and all the subheadings not included in the GC. Such metric is referred to as *Global Precision*.

After lemmatizing both the ground truth and the extracted terms that passed the post-filtering phase, Local Recall and General Precision are calculated through an exact match between the model predictions and the synonyms of the index terms in the respective ground truth. Figure 2 shows this through an example.

Baselines We compare the proposed architecture to a series of unsupervised and deep learning models. TF-IDF [13], and LDA [14] are popular methods that rely on statistical features. With LDA, we extract a number of topics equal to the sections in a book. From each of these, only the top two scored keywords are kept. Among the family of graph-based algorithms, TextRank [15] and TopicRank [16] are evaluated. With TextRank and TopicRank we get the 10-highest-scored candidates as keywords for each section and topic, respectively. Finally, the evaluation involves Key-BERT[6], a pre-trained application of the Transformer encoder for AKE, to extract relevant monograms and bigrams. All the baselines except for TFIDF and KeyBERT are implemented using the Python Keyphrase Extraction package[7].

5.2. Experiment 2: Semantic Space Analysis

In the second experiment, we investigate how the reciprocal semantic similarity of word embeddings generated by BERT changes before and after training the model. A study of

[6] https://maartengr.github.io/KeyBERT/
[7] https://boudinfl.github.io/pke/build/html/index.html

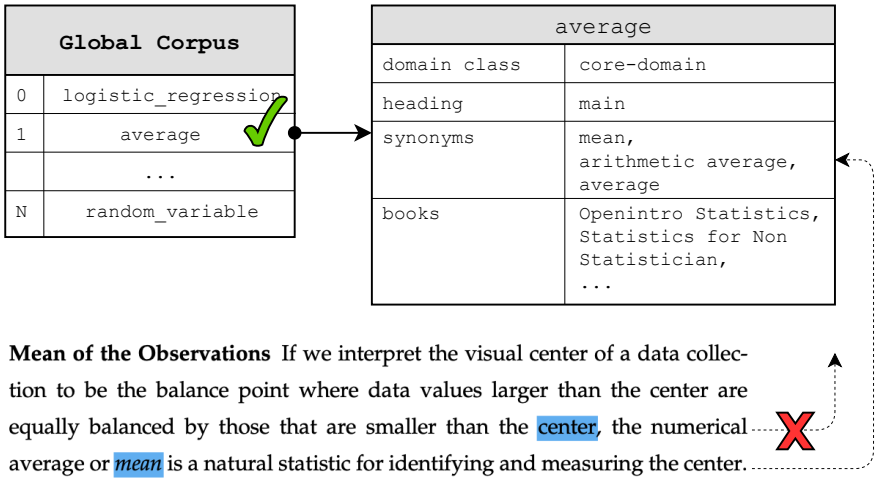

Figure 2: The computation of Local Recall and General Precision. Each term is considered to be retrieved if at least one of the synonyms is identified in the body of the book. In this specific case, the model produces one true positive and one false positive.

Table 1
Performance of various AKE algorithms.

%	General Precision	Local Recall	F1-score
TF-IDF	20.59	35.48	26.06
LDA	36.67	31.85	34.09
TextRank	12.99	27.41	17.62
TopicRank	39.54	49.04	43.78
Key-BERT	26.23	52.35	34.95
IndexBERT	**52.88**	**67.19**	**59.18**

BERT's embedding offers the opportunity to uncover patterns that explain its behavior in domain-specific scenarios, thus establishing guarantees that the performance will continue to be consistent when deployed in different applications. We hypothesize that after being fine-tuned on a domain-specific dataset, BERT captures more defined semantic similarities, generating a word space where domain-related terms lie in closer regions than out-of-domain ones.

To give proof of this, we compare the similarity scores between embeddings of domain-related and out-of-domain terms generated by BERT before and after fine-tuning. Since the encoder model is pre-trained and fine-tuned with natural language, feeding it with uncontextualized terms would produce nonoptimal representations [17]. For this reason, aggregate vector representations are created by averaging BERT's embeddings for the single term t in different contexts c_i. More formally: $\boldsymbol{t} = mean(t_{c_i}, ..., t_{c_n})$ where $\boldsymbol{c}_i$ is the sentence where the target occurs, and $\boldsymbol{t}_{c_i}$ is the token embedding t in the context c. Following [18]; we set a minimum sentence length of 4 tokens and a maximum bound at 21 tokens. For each target $\boldsymbol{t}$, up to 100 sentences are selected to produce aggregated representations [19]. Some terms have some contexts inferior to this threshold.

Table 2
Results of two U-test within domain-related terms and between domain-related and out-of-domain terms.

Effect	Cos. Sim. Before	Cos. Sim. After	t	Sig.
pull-in	0.72	0.85	9945	$\ll$.001
push-out	0.63	-0.52	161949	$\ll$.001

6. Results & Discussion

Results of the first experiment are shown in Table 1. The data show that IndexBERT outperformed all the baselines by a large margin. This has two relevant implications. First, the generated dataset can convey the target vocabulary distribution, giving a positive signal to be suitable for the training and evaluation of deep AKE algorithms. Second, it proves the effectiveness of fine-tuning BERT on the specific goal compared to models which are not tailored for any specific domain.

As regards the second experiment, Table 2 reports average similarity scores before and after fine-tuning BERT between word vectors of domain-relevant terms, i.e. belonging to the "core-domain" and "in-domain" categories (405 in total) and between domain-relevant terms and "out-of-domain" terms (29 in total). Running a non-parametric Mann-Whitney U-test, we noticed a significant difference in both cases. Particularly interesting is the shift in similarity found with out-of-domain terms. Although this might seem in contrast with the precision shown in Table 1, we attribute the result to the numerous partial matches found during the error analysis, an error that the model seems to be particularly prone to. Since the geometric distance between word vectors of domain-specific terms decreases and, conversely, increases for word vectors of different semantic domains, we decided to call these phenomena *pull-in effect* and *push-out effect*, respectively. Our conclusion is that the fine-tuned model can be interpreted as what in distributional semantics is known as region model [20]. Meaning that if BERT is fine-tuned on a close-domain task, it tends to identify semantic neighbors in topically similar terms, i.e. terms belonging to the same semantic domain.

7. Related Work

7.1. Existing Datasets and Annotation Paradigms

Traditional approaches to AKE dataset annotation require the cooperation of human experts that read through the text and highlight the most relevant information. This operation is frequently done by the authors of the documents or annotators hired specifically for that case study; such is the case of popular corpora like Inspect [21], SemEval2010 [22] or KP20k [23]. Since manual annotation is very time-consuming, other studies proposed a semi-automatic procedure in which keyphrases are generated with a mix of unsupervised algorithms, heuristics, and human editors.

Traditionally, AKE systems have been using a two-step approach. First, candidate keywords are identified based on their representativeness of the document, and then, whether the approach is supervised or unsupervised, they are respectively classified or ranked using various strategies

and features. Unsupervised approaches include phrase scoring methods based on statistical properties [24, 25], graph-based ranking [15, 26], and topic clustering techniques [16, 27]. Such methods come with the benefit of being completely independent of datasets. However, this configuration has one major flaw: unsupervised algorithms are inherently biased toward terms that describe more prevailing topics, suffering, therefore, a weak topic coverage. Supervised learning relies on manually-defined features derived from external resources (e.g. Knowledge Bases) or generated from the document itself, expressed by statistical, structural, or linguistic properties. Once these features are identified, the task is formulated as a binary classification where a machine learning algorithm, such as Naive Bayes algorithm [28, 29] Support Vector Machine [30], ensemble methods [31] or Artificial Neural Networks [32], is trained to map candidates keyphrases unto two classes, i.e. "key-phrase" and "not-key-phrase". More recent attention has focused on deep neural networks trained on a sequence labeling task to identify keyphrases. Recurrent Neural Networks [23], Bidirectional Long-Short Term Memory (BiLSTM) [33], and Transformers models [34] have been adopted for keyphrase extraction with unequaled results.

8. Conclusion and Future Work

In this research, we have described how textbooks can be used as a source of label data that can be used in multiple applications. Specifically, we have presented an end-to-end pipeline for creating domain-specific AKE corpora and a thorough evaluation of a BERT-based model trained on the generated dataset. Experimental results showed the benefit of the fine-tuned model over general-domain approaches. This outcome raises an important point: given the amount and variety of textbooks, it would possible to generate ad-hoc datasets with relative ease, opening the possibility of implementing AKE models in a larger number of domains where before was not possible due to lack of data. The second experiment demonstrated the semantic coherence in BERT's embedding space. This finding, while preliminary, suggests that BERT can be used in domain-oriented tasks where semantic reasoning is required.

Future research will extend this work by exploring a broader array of scenarios and applications where the rich content of textbooks can serve as a potential source of labeled data, thereby unlocking new possibilities for data-driven analysis and modeling. Additionally, regarding the current AKE approach, we are interested in improving the term filtering step to have a process free from human supervision. Also, given the variance in vocabulary distribution between academic fields, it would be interesting to verify the effectiveness of the AKE pipeline in different scenarios and study if and why the outcome differs.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[2] Y. Gu, R. Tinn, H. Cheng, M. Lucas, N. Usuyama, X. Liu, T. Naumann, J. Gao, H. Poon,

Domain-specific language model pretraining for biomedical natural language processing, ACM Transactions on Computing for Healthcare (HEALTH) 3 (2021) 1–23.
[3] Z. Zheng, X.-Z. Lu, K.-Y. Chen, Y.-C. Zhou, J.-R. Lin, Pretrained domain-specific language model for general information retrieval tasks in the aec domain, arXiv preprint arXiv:2203.04729 (2022).
[4] D. Araci, Finbert: Financial sentiment analysis with pre-trained language models, arXiv preprint arXiv:1908.10063 (2019).
[5] S. Zhou, N. Wang, L. Wang, H. Liu, R. Zhang, Cancerbert: a cancer domain-specific language model for extracting breast cancer phenotypes from electronic health records, Journal of the American Medical Informatics Association (2022).
[6] I. Alpizar-Chacon, S. Sosnovsky, Expanding the web of knowledge: one textbook at a time, in: the 30th ACM Conference on Hypertext and Social Media, HT '19, 2019.
[7] I. Alpizar-Chacon, S. Sosnovsky, Order out of chaos: Construction of knowledge models from pdf textbooks, in: Proceedings of the ACM Symposium on Document Engineering 2020, 2020, pp. 1–10.
[8] I. Alpizar-Chacon, S. Sosnovsky, Knowledge models from pdf textbooks, New Review of Hypermedia and Multimedia 27 (2021) 128–176.
[9] I. Alpizar-Chacon, S. Sosnovsky, What's in an index: Extracting domain-specific knowledge graphs from textbooks, in: ACM Web Conference, 2022.
[10] I. Alpizar-Chacon, J. Barria-Pineda, K. Akhuseyinoglu, S. Sosnovsky, P. Brusilovsky, Integrating textbooks with smart interactive content for learning programming, in: Proceedings of the Third Workshop on Intelligent Textbooks, volume 2895, CEUR WS, 2021, pp. 4–18.
[11] N. C. Mulvany, Indexing books, University of Chicago Press, 2009.
[12] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., Google's neural machine translation system: Bridging the gap between human and machine translation, arXiv preprint arXiv:1609.08144 (2016).
[13] G. Salton, M. J. McGill, Introduction to modern information retrieval, mcgraw-hill, 1983.
[14] D. M. Blei, A. Y. Ng, M. I. Jordan, Latent dirichlet allocation, Journal of machine Learning research 3 (2003) 993–1022.
[15] R. Mihalcea, P. Tarau, Textrank: Bringing order into text, in: Proceedings of the 2004 conference on empirical methods in natural language processing, 2004, pp. 404–411.
[16] A. Bougouin, F. Boudin, B. Daille, Topicrank: Graph-based topic ranking for keyphrase extraction, in: International joint conference on natural language processing (IJCNLP), 2013, pp. 543–551.
[17] G. Chronis, K. Erk, When is a bishop not like a rook? when it's like a rabbi! multi-prototype bert embeddings for estimating semantic relationships, in: Proceedings of the 24th Conference on Computational Natural Language Learning, 2020, pp. 227–244.
[18] A. Lenci, M. Sahlgren, P. Jeuniaux, A. Cuba Gyllensten, M. Miliani, A comparative evaluation and analysis of three generations of distributional semantic models, Language Resources and Evaluation (2022) 1–45.
[19] I. Vulić, E. M. Ponti, R. Litschko, G. Glavaš, A. Korhonen, Probing pretrained language models for lexical semantics, arXiv preprint arXiv:2010.05731 (2020).
[20] A. Lenci, Distributional models of word meaning, Annual review of Linguistics 4 (2018)

151–171.
[21] A. Hulth, Improved automatic keyword extraction given more linguistic knowledge, in: Proceedings of the 2003 conference on Empirical methods in natural language processing, 2003, pp. 216–223.
[22] S. N. Kim, O. Medelyan, M.-Y. Kan, T. Baldwin, Automatic keyphrase extraction from scientific articles, Language resources and evaluation 47 (2013) 723–742.
[23] R. Meng, S. Zhao, S. Han, D. He, P. Brusilovsky, Y. Chi, Deep keyphrase generation, arXiv preprint arXiv:1704.06879 (2017).
[24] Z. Liu, P. Li, Y. Zheng, M. Sun, Clustering to find exemplar terms for keyphrase extraction, in: Proceedings of the 2009 conference on empirical methods in natural language processing, 2009, pp. 257–266.
[25] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, A. Jatowt, Yake! keyword extraction from single documents using multiple local features, Information Sciences 509 (2020) 257–289.
[26] X. Wan, J. Xiao, Single document keyphrase extraction using neighborhood knowledge, in: AAAI, volume 8, 2008, pp. 855–860.
[27] Z. Liu, W. Huang, Y. Zheng, M. Sun, Automatic keyphrase extraction via topic decomposition, in: Proceedings of the 2010 conference on empirical methods in natural language processing, 2010, pp. 366–376.
[28] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, C. G. Nevill-Manning, Kea: Practical automatic keyphrase extraction, in: Proceedings of the fourth ACM conference on Digital libraries, 1999, pp. 254–255.
[29] C. Caragea, F. Bulgarov, A. Godea, S. D. Gollapalli, Citation-enhanced keyphrase extraction from research papers: A supervised approach, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1435–1446.
[30] K. Zhang, H. Xu, J. Tang, J. Li, Keyword extraction using support vector machine, in: international conference on web-age information management, Springer, 2006, pp. 85–96.
[31] A. Onan, S. Korukoğlu, H. Bulut, Ensemble of keyword extraction methods and classifiers in text classification, Expert Systems with Applications 57 (2016) 232–247.
[32] A. Azcarraga, M. D. Liu, R. Setiono, Keyword extraction using backpropagation neural networks and rule extraction, in: The 2012 international joint conference on neural networks (IJCNN), IEEE, 2012, pp. 1–7.
[33] D. Sahrawat, D. Mahata, H. Zhang, M. Kulkarni, A. Sharma, R. Gosangi, A. Stent, Y. Kumar, R. R. Shah, R. Zimmermann, Keyphrase extraction as sequence labeling using contextualized embeddings, in: European Conference on Information Retrieval, Springer, 2020, pp. 328–335.
[34] N. Nikzad-Khasmakhi, M.-R. Feizi-Derakhshi, M. Asgari-Chenaghlu, M.-A. Balafar, A.-R. Feizi-Derakhshi, T. Rahkar-Farshi, M. Ramezani, Z. Jahanbakhsh-Nagadeh, E. Zafarani-Moattar, M. Ranjbar-Khadivi, Phraseformer: Multimodal key-phrase extraction using transformer and graph embedding, arXiv preprint arXiv:2106.04939 (2021).