ECE496KCC Spring 2021 Concluding Report

Liam Xu, jx17, 671630451

May 2021

1 Introduction

Every researcher has their own research interest, which is reflected in their previous research work throughout their career. It is not infrequent that, one may be interested in another person's research interest.

It might be in customer-facing scenarios such as when a Ph.D. candidate is finding a matching group to join, or in institution-facing scenarios such as when a company is conducting preliminary filtering of candidate resumes during its recruiting process.

Therefore, it will be favorable, if an application, that is able to get representations of researcher interest accurately and in a reasonable time, can be developed.

This project proposes a general-purpose researcher keyword [1] extractor that facilitates researcher interest evaluation based on a pre-trained language model [7]. The extractor extracts a researcher's keywords from their publication titles and abstracts in a three-step manner: 1) Filter out all potential phrases that can be key phrases via Part-Of-Speech (POS) analysis and regular expression 2) Find a primary set of high-confidence key phrases, which is a subset of the phrases found in the previous step via similarity analysis based on semantic embedding 3) Increase key phrases quality by finer-grained selection via EmbedRank [1] based on Maximal Marginal Relevance (MMR).

The project is fully open-source and released on GitHub. [2]

2 The Problem

The problem can be summarized as finding a automatic mapping from a collection of texts $\mathcal{D}$ (i.e., researcher publication titles or abstracts) to a probability distribution, p, over a set of manually selected sub-field names of Computer Science, $\mathcal{L}$.

[1] The terminology "Keyword" is abused to also contain meaning of "Key phrases" (a consecutive combination of multiple keywords).

[2] https://github.com/LiamXu-xjl/XUKE

In this project, $\mathcal{D}$ is accessed from Microsoft Academic Graph (MAG) [8] with the user-provided researcher name and researcher affiliation. $\mathcal{D}$ is provided in the Appendix.

2.1 Major Difficulties

The major difficulties met during the development process can be summarized into three categories:

- **MAG interaction:** The abstract data is not directly given in the MAG APIs.
- **Keyword Extraction:** 1) Traditional keyword extraction is not time efficient. 2) There is no quantitative standards for keyword quality
- **Result delivery:** Traditional methods lack ability to allow users to control the result presentation. (e.g., A specific time span, a certain diversity level, or a specific keyword scale)

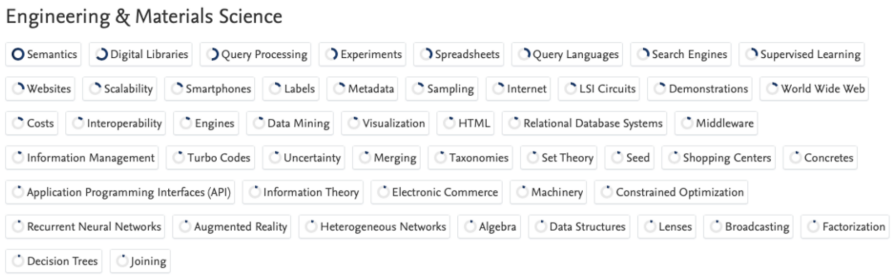

Figure 1: An example of Elsevier output

2.2 Existing approaches

Existing methods like Elsevier Fingerprint Engine [3] suffer from a series of issues brought by poor keyword quality. In figure 1, an example is displayed.

The keyword ("fingerprint") collection extracted by Elsevier contains the following flaws:

- Inaccuracy (e.g., "Augmented Reality"),
- Genericity (e.g., "Experiment"),
- Repetition (e.g., "Query processing" and "Query language")
- Un-relevance (e.g., "Shop Center")

[3]https://www.elsevier.com/solutions/ elsevier-fingerprint-engine

Also the Elsevier Fingerprint Engine lacks the user control freedom, namely the following:

- Only sees a screenshot of the whole research career
- Visualization not easy to view
- Does not support filtering for time, importance and diversity
- Fixed scale of vision fields

3 My Approaches

3.1 MAG Accessing

In the beginning, I had issues with requesting MAG for data. I actively reached out to other team members working on similar topics and received a existing script which was originally for other purposes. I learned about HTTP requests and JSON, with a better understanding of the MAG APIs, I improved the existing script and shared it with other students in the group.

Another major issue I had when accessing MAG was the abstract, the abstract text is not provided by MAG. However, after examination I found a intermediate signal provided by MAG, the IA, which provided words and the index of their appearance in the abstract. I utilized this signal to avoid the limitation by reconstructing the abstracts.

3.2 Keyword Extraction

3.2.1 TextRank

TextRank[4] is a classical graph-based keyword extraction method. It consists of two modules: 1) A graph building module that uses words as nodes and the fixed-window-size co-occurrence relationship as edges 2) A iterating module that runs PageRank[5] on the built graph.

TextRank is originally implemented as a baseline and was later improved by embedding-based methods for two reasons: 1) TextRank takes a long time calculating the graph and PageRank 2) TextRank lacks the ability to compare inter-paper word importance, i.e. TextRank fails to evaluate which is more important on words from different abstracts.

3.2.2 EmbedRank

The underlying framework was later changed to EmbedRank[1]. EmbedRank consists of three components:

- A pre-processing module, which contains a lemmatizer and a combinational tokenizer. The lemmatizer reduces inflectional form of words into

its original base via morphological analysis. (e.g. Studying → study, happiest → happy). The combinational tokenizer splits the document into possibly meaningful phrases with regular expressions of the form "zero or more adjectives followed by one or more nouns" (JJ*NN+) via regular expression

- A pre-trained language model that maps both a document and separate words into the same semantic space. In this project, SentenceTransformer [6] is chosen as the pre-trained framework. Specifically, the model bert-base-nli-mean-tokens[4] is used.
- A keyword diversification module that improves keyword quality.

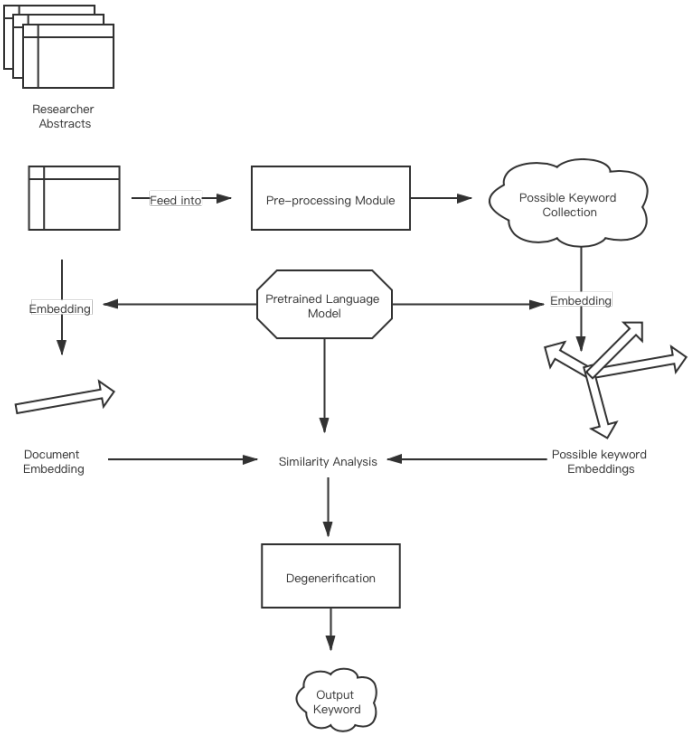

Figure 2: The EmbedRank Flow

As displayed in figure 2, The researcher abstracts $\mathcal{D}$ are first passed through the pre-processing module, which gives a collection of possible keywords. Then the document and the possible keywords are embedded separately with the pre-trained language model. They are further conducted similarity analysis,

[4]https://huggingface.co/sentence-transformers/bert-base-nli-mean-tokens

after which a de-generification module is used to de-generificate the keywords extracted and outputs the final keyword results.

3.3 Sub-field Mapping

After a keyword collection of high quality is obtained, the embedding vector of the keywords are projected into semantic space constituted by the pre-set manually selected sub-field names of Computer Science, $\mathcal{L}$.

Here, it is worth noting that both the cosine similarity and Euclidean distances between $\mathcal{L}$ and the keywords are taken into consideration.

The degenerification module also comes into play in this phase. A possible keyword without personality (i.e., the similarity does not lean towards some sub-fields in $\mathcal{L}$) will be regarded as generic and therefore eliminated.

3.4 Visualization

Visualization is conducted with WordCloud, and the HTML/CSS and JavaScript version is under developing.

4 Results

4.1 Input

The application requires only two inputs: researcher name and researcher affiliation. It will then automatically conduct the data accessing, keyword extraction and the sub-field evaluation.

4.2 Controlling features

The application allows the user to specify

- The time span of interest of the researcher's career (e.g. the last third)
- The keyword ratio to keep in the abstracts (e.g. the most important 10%)
- The diversity factor of the keyword extraction between 0 an 1 (e.g. 0.9 implies a highly diverse keyword population)

4.3 Example

One example of the application is shown in figure 3 in the Appendix. It can be seen that the sub-fields are correctly mapped and the keywords extracted are of favorable quality.

4.4 Assessment

The application achieved the goal set mainly on the following two major aspects:

4.4.1 Better Keywords

The application improves keyword quality on

- Accuracy
- Non-genericity
- Uniqueness
- Relevance

4.4.2 Increased User Control freedom

The application gives user freedom on

- Specifying interested time span
- Utilizing easier wordcloud presentation
- Quantifying researcher's profile

5 Future Plans

This work can be formulated as a variation of the task of Weakly Supervised Text Classification, with the texts to be classified being $\mathcal{D}$ and the sub-field names being $\mathcal{L}$. In my opinion, it's worth exploring existing classification methods such as [9], [3] and [2].

The idea of using keywords to facilitate the classification task is also an interesting one. An equivalent of the concept of scale-invariant feature transform (SIFT) in the realm of Computer Vision might be introduced as an intermediate phase.

6 Reflection

In the process of building this project, I have learned a lot. I developed my skill-set and became more confident for future works in the alike fields. I have also learned the importance of teamwork and good atmosphere. It is very important to communicate with others working on similar topics, to share opinions and methodologies, and to sync with each other's every now and then.

7 Appendix

7.1 The sub-field names of Computer Science

Subfield Names
Computer Architecture
Compilers
Parallel Computing
Artificial Intelligence
Machine Learning
Bioinformatics
Computational Biology
Databases
Data Mining
Information Retrieval
Interactive Computing
Programming Languages
Formal Methods
Software Engineering
Scientific Computing
Computer Security and Privacy

7.2 An example

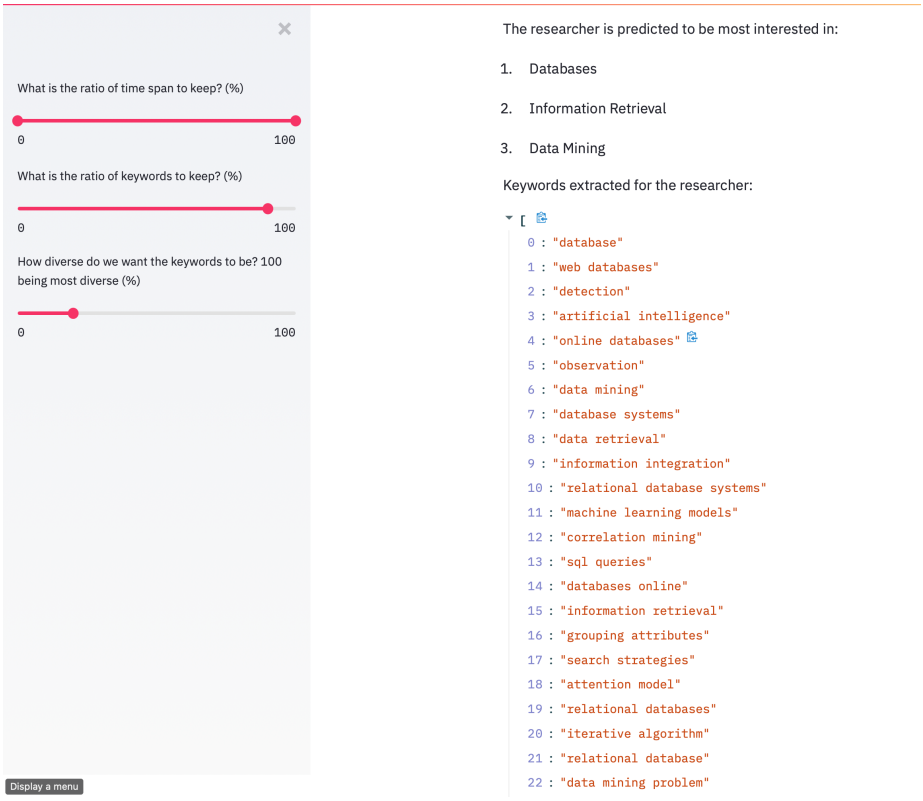

Figure 3: An example of the application

References

[1] Kamil Bennani-Smires et al. "EmbedRank: Unsupervised Keyphrase Extraction using Sentence Embeddings". In: *CoRR* abs/1801.04470 (2018). arXiv: 1801.04470. URL: http://arxiv.org/abs/1801.04470.

[2] Yu Meng et al. "Text Classification Using Label Names Only: A Language Model Self-Training Approach". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. 2020.

[3] Yu Meng et al. "Weakly-Supervised Neural Text Classification". In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. CIKM '18. Torino, Italy: Association for Computing Machinery, 2018, pp. 983–992. ISBN: 9781450360142. DOI: 10.1145/3269206.3271737. URL: https://doi.org/10.1145/3269206.3271737.

[4] Rada Mihalcea and Paul Tarau. "TextRank: Bringing Order into Text". In: *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Barcelona, Spain: Association for Computational Linguistics, July 2004, pp. 404–411. URL: https://www.aclweb.org/anthology/W04-3252.

[5] Lawrence Page et al. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical Report 1999-66. Previous number = SIDL-WP-1999-0120. Stanford InfoLab, Nov. 1999. URL: http://ilpubs.stanford.edu:8090/422/.

[6] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: https://arxiv.org/abs/1908.10084.

[7] Victor Sanh et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *CoRR* abs/1910.01108 (2019). arXiv: 1910.01108. URL: http://arxiv.org/abs/1910.01108.

[8] Kuansan Wang et al. "Microsoft Academic Graph: When experts are not enough". In: *Quant. Sci. Stud.* 1.1 (2020), pp. 396–413. DOI: 10.1162/qss_a_00021. URL: https://doi.org/10.1162/qss%5C_a%5C_00021.

[9] Zihan Wang, Dheeraj Mekala, and Jingbo Shang. *X-Class: Text Classification with Extremely Weak Supervision*. 2020. arXiv: 2010.12794 [cs.CL].