Technology Review: Keyword Extraction Toolkits for Medical Literature in English

Introduction

The performance of search engines in terms of accuracy and speed is crucial for an optimal user experience. Given that medicine is a rapidly evolving field, search trends frequently change as new diseases, treatments, and methodologies are discovered, which adds an extra layer of complexity to identifying search trends and caching some of the results for those trends. One way to identify such trends is to crawl recent content in popular health forums, extract keywords, search for those keywords, and finally store the results in the cache. With that said, keyword extraction is a topic of research interest over the last few years. A comparison of keyword extraction toolkits for medical literature is largely unexplored, as such, I will review and compare some of the approaches and toolkits that might be used for this purpose.

Methodology

The first toolkit I tested is Yet Another Keyword Extractor (Yake), which is a python library that utilizes text statistical features from each document for the keyword extraction. As such, it does not need to be trained, it is language and domain independent, and can scale linearly to the size of the document being analyzed. Experimental results from the developers claim that their algorithm outperformed the state-of-the-art unsupervised approaches such as TF-IDF, KP-Miner, Rake, and TextRank when tested with news and computer science literature datasets. However, it has not been tested within the medical context. As such, it is yet to be seen whether Yake's performance will suffer compared to language and domain dependent approaches and toolkits.

The second toolkit I tested is spaCy, which is an open source Python library for natural language processing (NLP) tasks with a focus on production use. One of its main advantages is that it offers pre-trained pipelines for various languages, such as Russian, and various topics, such as scientific biomedical text. As such, I attempted to use the "en_core_sci_md" pipeline, which is optimized for biomedical text, but it was not available for the most recent spaCy versions. Instead, I chose the "en_core_web_trf" pipeline, which is optimized for web content, but has a large size for better accuracy. I also used "en_core_web_sm" and "en_core_web_md" which are smaller than the trf pipeline to study the tradeoffs between accuracy and speed. Finally, I added the Text Rank and the Biased Text Rank algorithms to the pipelines to extract the keywords.

The third toolkit I tested is Python Keyphrase Extraction (PKE), which is a toolkit that provides an end-to-end keyphrase extraction pipeline that can easily be extended or used for benchmarking various approaches. Several supervised learning and unsupervised learning algorithms are implemented, including Text Rank, Yake and several more. Initially, I wanted to explore TopicRank, but the algorithm seemed to be buggy and always ran into a division by zero error. According to my research, this could be as a result of not finding enough keywords, but

there were not many resources available. Other algorithms, such TF-IDF, may require a specific input format in order to generate document frequency data, which is another barrier for this toolkit. It is also worth noting that PKE depends on the “en_core_web_sm” corpus by spaCy for some of its approaches.

The comparison covers performance in accuracy and speed as well as the ease of use. For the benchmark testing, I performed 10 trials on a set of 10 medical research papers along with keywords written by the authors. The accuracy is defined as the cosine similarity between the set of keywords produced by the toolkit and the set of keywords written by the authors while the speed is measured as the seconds a toolkit takes to perform the analysis on the documents set.

Performance Results

Toolkit	Algorithm	Cosine Similarity	Time (seconds)
Yake	Yake	0.23	2.85
spaCy (sm, md, trf)	Text Rank	0.12, 0.11, 0.19	13.84, 14.40, 261.01
spaCy (sm, md, trf)	Biased Text Rank	0.12, 0.11, 0.19	13.84, 14.35, 263.07
PKE	Text Rank	0.01	16.75
PKE	Yake	0.06	19.10

It can be seen from this table that the Yake toolkit is by far the best performer both in terms of accuracy and speed. On the other hand, the PKE toolkit was the worst performer in both Text Rank and Yake algorithms in terms of accuracy. SpaCy’s performance across the various algorithms and models performed in-between the two. It seemed to perform closer to PKE when using the smaller and faster pipelines. On the other hand, it gains some accuracy when using the larger pipeline but at a very significant computational cost. Another note is that the Yake algorithm seems to outperform Text Rank both when comparing the Yake toolkit to spaCy and when comparing Yake and Text Rank algorithms within PKE. Also, it seems that Biased Text Rank performed identically to TextRank within spaCy.

Discussion

Overall, spaCy was relatively easy to install and use. It has a large community support, which is shown by the number of languages it supports and the frequent updates. However, in this test, spaCy did not perform particularly well. This can be down to the graph-based approach that Text Rank uses, which adds significant computational overhead. Furthermore, spaCy does not allow the user to specify the n-gram as a parameter, which may be computationally costly as keywords might need to be split while removing duplicates depending on the application. With that being said, I am wondering if utilizing other algorithms by using another toolkit in combination with

the spaCy pipeline will improve its performance since Text Rank was consistently underperforming.

The Yake toolkit was the easiest to use and install among the tested toolkits. It was also the most accurate and the fastest. One of the contributors to it's high performance is that it only depends on the document analyzed, so it does not need to do collection-level computations. In fact, it's computationally light enough for use on mobile devices, where it is critical to consider not only computational cost but also energy cost. Furthermore, it is available for deployment within a container to the cloud as an application programming interface, thus eliminating any computational costs on the user's application. However, a potential disadvantage is that the performance might be worse compared to other algorithms and toolkits depending on the application, but this was not the case in this test.

Finally, the PKE toolkit was relatively easy to install, but was not as easy to use compared to the other toolkits. Unfortunately, it does not seem to have a strong community to support it, which could be a contributing factor to its lacking performance as there might be implementation-level differences. Also, its dependence on the "en_core_web_sm" corpus might hinder the performance depending on the application. However, it might be a perfect toolkit for those who would like to filter out the various keyword extraction algorithms, as their performance can be application dependent and can be extended in the future to include more algorithms.

Conclusion

In conclusion, the Yake toolkit seems to be the clear choice to use for keyword extraction from medical literature in English. Its accuracy, speed, ease of use, and independence of languages and models makes it very convenient for use. As such, I will be utilizing Yake as the keyword extraction toolkit for medical search trend identification purposes in my course project.

References

1. Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, Adam Jatowt (2019). YAKE! Keyword extraction from single documents using multiple local features, *Information Sciences*, Volume 509, 2020, Pages 257-289, ISSN 0020-0255, https://doi.org/10.1016/j.ins.2019.09.013.
2. Yake Github https://github.com/LIAAD/yake
3. Boudin, F. (2016). pke: an open source python-based keyphrase extraction toolkit. *COLING*. https://aclanthology.org/C16-2015.pdf
4. PKE Github https://github.com/boudinfl/pke

5. SpaCy Official Website
https://spacy.io/

Appendix

List of Medical Documents used for the experiment and their corresponding keywords

#	Article Name	Keywords
1	A systematic review of asymptomatic infections with COVID-19	SARS-CoV-2; COVID-19; Asymptomatic infections; Epidemiological characteristic; Outcome
2	Diagnosis of COVID-19 for controlling the pandemic: A review of the state-of-the-art	SARS-CoV-2; COVID-19; Virus detection; Diagnostics; Pneumonia; Serology tests
3	Remdesivir in COVID-19: A critical review of pharmacology, pre-clinical and clinical studies	Remdesivir; COVID-19; SARS-Co-V-2; Clinical outcome; Mortality
4	Covid19, beyond just the lungs: A review of multisystemic involvement by Covid19	COVID19; Multi-systemic; Lungs; Diffuse alveolar damage
5	A new drug formula for pneumonia and severe seasonal flu; a promising drug for eradicate COVID19	COVID19; Pneumonia; Seasonal flu; The GC/MS data; Pneumonia and severe seasonalful; Health Organization (WHO)
6	Differences in responsiveness of intratympanic steroid injection for intractable vertigo in Meniere's disease	Meniere's disease; Intratympanic steroids; Vestibular evoked myogenic potential; Saccule; Endolymphatic hydrops
7	Meniere's disease: Medical management, rationale for vestibular preservation and suggested protocol in medical failure	Meniere's disease; Intratympanic steroid; Intratympanic gentamicin; Endolymphatic hydrops; Vestibular migraine; Treatment of Meniere's disease; Endolymphatic sac shunt
8	The relationship between nutrition and Ménière's disease	Ménière's disease; Endolymphatic hydrops; Low sodium intake; Gluten free diet; Allergy
9	An infrequent type of nystagmus during a vertigo crisis in Meniére's disease	Downbeat nystagmus; Meniere disease; Vertigo crisis
10	Comparison of inner ear MRI enhancement in patients with Meniere’s disease after intravenous injection of gadobutrol, gadoterate meglumine, or gadodiamide	Magnetic resonance imaging; Inner ear; Intravenous injection; Perilymph