Textstar: a Fast and Lightweight Graph-Based Algorithm for Extractive Summarization and Keyphrase Extraction

David Brock
Dallas College
dbrock@dallascollege.edu

Ali Khan
University of North Texas
alikhan@my.unt.edu

Tam Doan
University of North Texas
tamdoan@my.unt.edu

Alicia Lin
University of North Texas
alicia.y.lin@gmail.com

Yifan Guo
University of North Texas
yifan.guo.3517@gmail.com

Paul Tarau
University of North Texas
paul.tarau@unt.edu

Abstract

We introduce Textstar, a graph-based summarization and keyphrase extraction system that builds a document graph using only lemmatization and POS tagging. The document graph aggregates connections between lemma and sentence identifier nodes. Consecutive lemmas in each sentence, as well as consecutive sentences themselves, are connected in rings to form a "ring of rings" representing the document. We iteratively apply a centrality algorithm of our choice to the document graph and trim the lowest ranked nodes at each step. After the desired number of remaining sentences and lemmas is reached, we extract the sentences as the summary, and the remaining lemmas are aggregated into keyphrases using their context. Our algorithm is efficient enough to process large document graphs without any training, and empirical evaluation on several benchmarks indicates that our performance is higher than most other graph-based algorithms.

1 Introduction

Contemporary natural language processing is mostly done through neural networks. However, this is resource intensive and requires large amounts of data. This can be a problem for languages that are not widely spoken, due to insufficient data for training these models. Even for tasks where neural network based models excel, they are often an overkill. State of the art Transformer-based tools such as BERT (Devlin et al., 2019), GPT-3 (Brown et al., 2020), and even Longformer (Beltagy et al., 2020) have size limits and require hierarchical approaches to long documents. However, graph-based approaches are one-shot algorithms that do not require expensive computational resources to train. The need for a fast graph-based summarizer is also justified as a preprocessor to assist these neural models by enabling them to work on salient smaller subsets of a large document.

To address this insufficiency, we propose Textstar, a lightweight graph-based summarization and keyphrase extraction algorithm that outperforms most other graph-based methods. Our model is language-independent, making it suitable for application to languages with insufficient training data. Additionally, it simultaneously supports keyphrase extraction and summarization. This flexibility, along with its short runtime, opens up possibilities for many applications.

We will start with an overview of our system, as presented in Figure 1. After a document is uploaded, it is pre-processed. Then, each sentence is converted into a connected ring of nodes (Figure 2). Additionally, the sentence IDs are also connected into a ring of nodes. Afterwards, the graph is fed to the Textstar algorithm, which gradually trims out word nodes and sentence ID nodes that have low ranking values. When the graph contains only the desired number of sentence ID nodes or keyword nodes, we feed this information to the postprocessing component. The postprocessing component then converts certain keywords into keyphrases and combines the sentences represented by the sentence ID nodes into a summary.

Our contribution is as follows:

(1) We introduce a novel graph-based algorithm that extracts both summaries and keyphrases at the same time.

(2) We construct textgraphs via the ring-of-rings method.

(3) Our Textstar algorithm is an iterative text graph trimming approach for identifying in one pass the most important sentences and keyphrases.

(4) We show that our system improves the state-of-the-art with respect to other similar graph-based algorithms.

The rest of the paper is organized as follows:

Section 2 overviews related work.

Section 3 describes the algorithm and implementation.

Section 4 provides empirical analysis.

Section 5 analyzes the results and discusses the limitations.

Section 6 concludes the paper.

2 Related Work

Graph-based approaches to text summarization and keyphrase extraction are well-established. TextRank (Mihalcea and Tarau, 2004) and its derivatives are popular unsupervised approaches to text summarization and keyphrase extraction. They utilize graph-based centrality algorithms to score sentences or words with the assumption that sentences or words with the highest centrality scores are expected to have the highest importance in a document. TextRank, in particular, uses the PageRank algorithm (Page et al., 1999) as its scoring mechanism.

Several methods have been proposed that improve the base TextRank algorithm by changing the scoring metric (Barrios et al., 2016) or by changing the construction of the textgraph using salient information about the text or by use of word embeddings.

Bougouin et al. (2013) discover and categorize candidate keyphrases to topics by applying the Hierarchical Agglomerative Clustering algorithm. Then, a weighted complete and undirected graph is generated where nodes represent the topics and weighted edges show the semantic relations of the topics. Keyphrases that best represent each topic are chosen with three criteria: appearing first in the document, appearing most frequently in the document, and being the most similar to other keyphrases in the topic. Then, each topic is ranked by the Textrank algorithm. Choosing the topics with the N highest scores and selecting the most significant keyphrase per topic generates the final set of keyphrases.

Florescu and Caragea (2017) retrieve nouns and adjectives and construct an undirected word graph in which each node is a unique word and the weight of an edge is calculated from the number of bigram co-occurrences in the document. The biased PageRank score of each word is counted by considering both its position and its frequency. The sum of scores of words in each keyphrase generates the keyphrase's score.

Boudin (2018) selects keyphrase candidates and classifies word stems to topics in a manner similar to Bougouin et al. (2013). Then, a complete directed k-partite graph is constructed where each node is a keyphrase, an edge connects 2 different topics, the weight of an edge shows a distance between 2 nodes in the document, and k is the number of topics. In addition, the incoming weight of the first node of each topic is adjusted. Then the TextRank algorithm gives the score for each node, and the N top scoring keyphrases are extracted.

LexRank (Erkan and Radev, 2004) showcases improved summarization by introducing the idea of computed eigenvector centrality. This method constructs a weighted undirected cosine similarity graph cluster from the given multiple documents, where nodes denote sentences and a weighted edge signifies the idf-modified-cosine of 2 nodes. Then, the graph is transferred to an undirected graph which focuses on the salient similar sentences by setting a threshold. The LexRank score of each node is calculated based on eigenvector centrality. The summary is N top scoring sentences.

Most graph-based methods perform either extractive text summarization or automatic keyphrase extraction, but not both. Neural methods have recently been shown to be effective at multi-task natural language processing. However, like graph-based methods, there is little work on neural methods that perform both extractive summarization and keyphrase extraction.

Our approach implements a multi-task approach to summarization and keyphrase extraction by creating a textgraph sharing both sentences and word nodes. We also introduce a different topology for building a textgraph using a ring-of-rings construction for connecting both words in a sentence and sentences among them. At the same time, a new method is used to compute rankings by successive trimming of unimportant nodes until the required number of sentences and keyphrases is reached.

3 Method

Our overall method is shown in Figure 1. The first step of processing a document is to remove words and sentences that are unlikely to contain relevant information. A text graph with a ring-of-rings structure is then constructed using the remaining words and sentences. Using the graph, the core

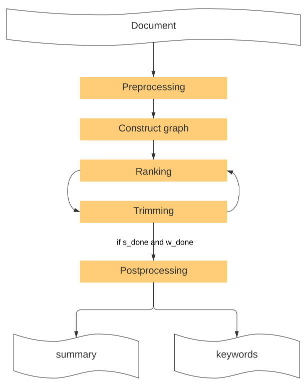

Figure 1: Overview Method

Textstar algorithm works by repeatedly computing a centrality metric and then removing low ranked nodes. This process continues until the desired number of summary sentences and key words is reached. Finally, the word and sentence nodes left in the graph are post-processed to create the final summary and keyphrases.

3.1 Text Preprocessing

Using the NLTK Python package[1] (Bird et al., 2009), we first split the text into sentences (sentence tokenization) and the sentences into words (word tokenization). The words and sentences are filtered to remove those that are unlikely to contain useful information. Sentences are removed if they are too long and/or noisy after the pdftotext translation. We also perform stopword removal. Finally, we also use NLTK to lemmatize the words and apply a basic POS tagging.

3.2 The Ring of Rings Textgraph Construction

We construct the textgraph of the document as a ring of rings meta-structure, in which each sentence is a ring and each sentence is connected to a node in the central ring. This structure allows for the natural encapsulation of information from the document, including word and sentence position in a directed graph. Moreover, the ring structures of words and sentences allow both words and sentences to be connected back to front; this is important because later words/sentences refer to earlier introductions.

[1]https://www.nltk.org/api/nltk.html

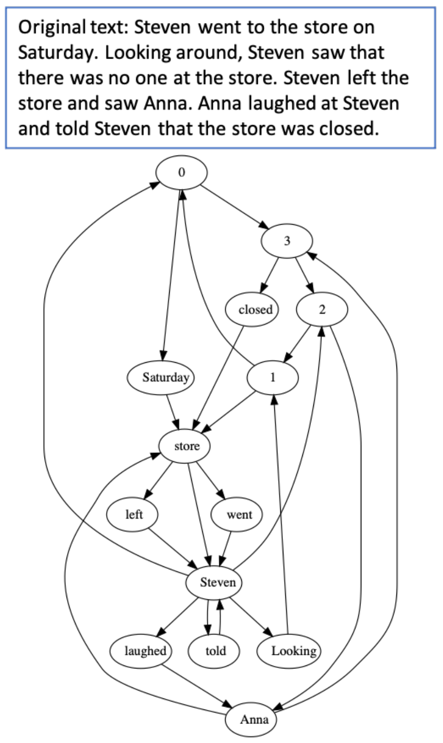
Original text: Steven went to the store on Saturday. Looking around, Steven saw that there was no one at the store. Steven left the store and saw Anna. Anna laughed at Steven and told Steven that the store was closed.

Figure 2: Example Graph

The graph is created from the cleaned and lemmatized text. To facilitate both summarization and keyphrase extraction, the graph's nodes represent both words and sentences. Strings represent words, and sentences are represented by an integer ID.

The cleaned and lemmatized words of a sentence are connected in reverse order, and the sentence's id is connected between the first and last word of the sentence to form a ring. The sentence nodes are also connected in reverse order, and the last sentence node is adjacent to the first sentence node to form the central ring. As a result, a ring of rings with a structure similar to that shown in Figure 3 is formed. An example of the resulting graph of an extremely short text is shown in Figure 2. The ring of sentence ids is shown as the nodes labeled 0 to 3. At the same time, the ring-of-rings torus topology is distorted by the shared occurrences of words such as 'Steven' and 'store' that originate from multiple word rings. Note that some nodes are shared by multiple rings, since some words (e.g. store) are shared by multiple sentences. We also add edges between compounds.

3.3 The Trimming Algorithm

After the graph (containing both word and sentence nodes) is generated, it is passed to the Textstar algorithm. The algorithm first ranks the nodes in the graph using a ranking function. From our tests, the degree centrality ranking function performs best, although Pagerank also works well. The nodes are sorted based on rank and only the highest X percent are kept, where X is a parameter that can be tuned. For summarization, a value of X around 70-80 percent works best, and for keyphrase extraction X can be a bit lower.

This process is then repeated, with the graph being re-ranked and then trimmed. When the number of remaining sentence nodes and word nodes drops below the desired number of summary sentences and key words, respectively, iteration stops.

Algorithm 1: The Textstar Algorithm

```
Input: g: Textgraph of the document
       ranker: ranking algorithm
       sumsize: final number of sentences,
       kwsize: final number of keyphrases,
       trim: percent of lowest ranked
       nodes to remove per step
Result: final_sids, final_kwds
while true do
    ranks ← Ranker(g);
    sids ← $\forall x \in$ ranks, if x is a sentence id;
    kwds ← $\forall x \in$ ranks, if x is a lemma;
    s_done ← length of sids $\leq$ sumsize;
    w_done ← length of kwds $\leq$ kwsize;
    n ← number of nodes in g;
    if not s_done then
        final_sids ← sids;
    end
    if not w_done then
        final_kwds ← kwds;
    end
    if s_done and w_done then
        break;
    end
    split ← trim * n // 100;
    for $i$ = split...n do
        g.remove(ranks[i])
    end
end
```

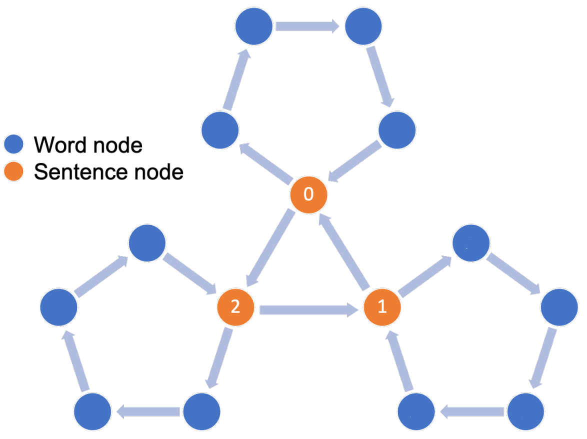

Figure 3: Ring of Rings Structure

3.4 Postprocessing

The summary is generated by taking the remaining sentence nodes from the graph. These nodes have the highest ranks, and the associated sentences from the original text are extracted. To make the summary more readable, the summary sentences are sorted according to the order they appear in the original text.

Similarly, the keyphrases are extracted from the word nodes in the final graph. Only unique word nodes with the highest ranks are taken.

4 Evaluation

We have used the Degree Centrality ranking algorithm, with the summary size and keyphrase size as 6 and the trim percentage set to 80%.

Table 1 provides extractive summarization results on the *arXiv* and *PubMed* datasets. The PubMed dataset, which has 133K scientific documents, is divided into a training set (125,020 documents, 94%); a validation set (3,990 documents, 3 %); and a test set (3,990 documents, 3%). The ArXiv dataset, which contains 215K scientific documents, is distributed between a training set (193,500 documents, 90%); a validation set (10,750 documents, 5 %); and a test set (10,750 documents, 5%).The gold summary of each document in both these datasets is the abstract of the document.(Cohan et al., 2018).

We compare Textstar against well-known extractive graphical algorithms: *LSA* (Steinberger et al., 2004) , *SumBasic* (Vanderwende et al., 2007), and *LexRank* (Erkan and Radev, 2004). We use the results of these algorithms found in Cohan et al. (2018).

Table 2 provides the results for automatic keyhprase extraction. We evaluate on the following well-known datasets:

- *Inspec*: This dataset contains 2,000 short English texts, which are collected from the Inspec database from between 1998 and 2002. Each piece consists of an abstract, a title, and keyphrases. (Hulth, 2003).

- *SemEval*: This dataset consists of 284 English scientific articles from the ACM Digital Library in four topics: Distributed Systems, Information Search and Retrieval, Distributed Artificial Intelligence, and Behavioral Sciences - Economics. The distribution of each topic is equal. The gold keyphrases were cautiously selected by both authors and readers. (Kim et al., 2010).

Our model is evaluated by f-mesure on the top K keypharses (F1@K). Textstar is compared against the following graph-based keyphrase extraction algorithms: *TextRank* (Mihalcea and Tarau, 2004), *SingleRank* (Wan and Xiao, 2008), *TopicRank* (Bougouin et al., 2013), *PositionRank* (Florescu and Caragea, 2017), and *MultipartiteRank* (Boudin, 2018). The results of *Inspec* and *SemEval2010* for the baseline algorithms are obtained from Liang et al. (2021). Textstar is also comparable to *CopyRNN* (Meng et al., 2017) and outperforms *RNN* (Meng et al., 2017) deep learning model on both the Inspec and SemEval dataset. These results are gained from Meng et al. (2017)

To evaluate these algorithms, we use a Python implementation of the ROUGE (Lin, 2004) metric[2]. Our tests show that we outperform other graph-based algorithms for text summarization on arXiv, and are competitive to LexRank on the PubMed dataset. For keyphrase extraction, we outperform all other graph-based algorithms on all datasets with the exception of the Inspec dataset.

5 Discussion

The experiments show that the algorithm is competitive on benchmarks of both extractive summarization and automatic keyphrase extraction. Whereas separate sentence and word text graphs lose information from the original text, reducing the effectiveness of either task, our multi-task approach takes advantage of the synergies between summarization and keyphrase extraction, allowing for better results than either individually. The important words in a document are strongly correlated to the important sentences. We make use of the relationship between the words and the structure of the document explicitly.

5.1 Limitations

The Textstar algorithm shares its limitations with the larger graph-based family of extractive summarization and keyphrase extraction algorithms:

- performance is usually worse than state-of-the-art of deep learning algorithms

- textgraphs generally do not rely on deeper syntactic and semantic information

- textgraph-based algorithms do not make use of domain knowledge

- extracted summaries are not natural to human readers

- textgraphs generally do not perform well on very short documents

Some of these limitations can be alleviated by bringing in richer syntactic information (e.g., dependency trees) and semantic relations extracted from the text or from knowledge graphs specific to the domain of the document along the lines of Tarau and Blanco (2021).

6 Conclusions

We introduced *Textstar*, a multi-task graph-based extractive text summarization and automatic keyphrase extraction algorithm. By iteratively simplifying the text graph while eliminating the lowest ranked scores as determined by a centrality algorithm, we efficiently determine the most salient sentences and keyphrases. Moreover, by building the textgraphs from both sentence and word nodes, we extract in one pass both summaries and keyphrases.

By aggregating information about word subsequences occurring in a sentence and sentence subsequences occurring in a document, we show that we outperform most other graph-based methods.

While like most other graph-based methods, Textstar's performance does not match that of state-of-the-art deep learning frameworks, Textstar can act as a useful preprocessor to them to

[2]https://github.com/Diego999/py-rouge

Algorithm	arXiv			PubMed		
	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-1	ROUGE-2	ROUGE-L
LSA	29.91	7.42	25.67	33.89	9.93	29.70
SumBasic	29.47	6.95	26.30	37.15	11.36	33.43
LexRank	33.85	10.73	28.99	**39.19**	13.89	**34.59**
Textstar	**38.8**	**12.8**	**32.1**	38.6	**13.9**	32.0

Table 1: Summarization results on PubMed and arXiv dataset.

Algorithm	Inspec		SemEval 2010	
	F1@5	F1@10	F1@5	F1@10
Graph_based Models				
TextRank	27.04	25.08	3.80	5.38
SingleRank	27.79	34.46	5.90	9.02
TopicRank	25.38	28.46	12.12	12.90
PositionRank	**28.12**	32.87	9.84	13.34
Textstar	24.70	**34.70**	**15.20**	**22.80**
Deep Learning Models				
RNN	8.50	6.40	15.70	12.40
CopyRNN	**27.80**	**34.20**	**29.30**	**30.40**

Table 2: Result of keyphrase extraction with metrics F1@5 and F1@10 in Inspec and SemEval 2010 datasets.

accommodate the input size limitations in various neural systems.

Future work is planned to evaluate the use of Textstar as a preprocessor for transformer-based systems that have input-size limitations as well for enhancing the text graph with similarity links between sentence and word embeddings.

References

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

S. Bird, E. Klein, and E. Loper. 2009. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly Media.

Florian Boudin. 2018. Unsupervised keyphrase extraction with multipartite graphs. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 667–672, New Orleans, Louisiana. Association for Computational Linguistics.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. TopicRank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 543–551, Nagoya, Japan. Asian Federation of Natural Language Processing.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the*

Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

G. Erkan and D.R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

Corina Florescu and Cornelia Caragea. 2017. PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1115, Vancouver, Canada. Association for Computational Linguistics.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, EMNLP '03, page 216–223, USA. Association for Computational Linguistics.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26.

Xinnian Liang, Shuangzhi Wu, Mu Li, and Zhoujun Li. 2021. Unsupervised keyphrase extraction by jointly modeling local and global context. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 155–164, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592, Vancouver, Canada. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Josef Steinberger, Karel Jezek, et al. 2004. Using latent semantic analysis in text summarization and summary evaluation. *Proc. ISIM*, 4(93-100):8.

Paul Tarau and Eduardo Blanco. 2021. Interactive Text Graph Mining with a Prolog-Based Dialog Engine. *Theory Pract. Log. Program.*, 21(2):244–263.

Lucy Vanderwende, Hisami Suzuki, Chris Brockett, and Ani Nenkova. 2007. Beyond sumbasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing & Management*, 43(6):1606–1618.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge.

Appendix A. Examples of Textstar Output

Below are the results of the Textstar algorithm on a near final version of this paper.

> **Summary:** Our contribution is as follows: (1) We introduce a novel graph-based algorithm that extracts both summaries and keyphrases at the same time. The words and sentences are filtered to remove those that are unlikely to contain useful information. Sentences are removed if they are too long or too noisy after the pdftotext translation. This structure allows for the natural encapsulation of information from the document, including word and sentence position in a directed graph. Moreover, the ring structures of words and sentences allow both words and sentences to be connected back to front; this is important because later words/sentences refer to earlier introductions. The algorithm first ranks the nodes in the graph using a ranking function. Table 1 provides extractive summarization results on the arXiv and PubMed datasets. Moreover, by building the textgraphs from both sentence and word nodes, we extract in one pass both summaries and keyphrases.
>
> **Keyphrases:** *'word', 'node', 'sentence', 'ring', 'document', 'connected', 'score'*

Below are the results of running Textstar on the following few paragraphs from a news article about the *Bloom* deep learning-based language model [3].

> Now there is a true open-source alternative to GPT-3, BigScience Bloom, which is freely available for research and enterprise purposes. Bloom was trained over 117 days at the supercomputing center of the French National Center for Scientific Research and is 176 billion parameters in size. The development involved over 1000 volunteer researchers, organized in the BigScience project, coordinated by Hugging Face, and co-funded by the French government. Bloom can be downloaded for free on Hugging Face and is said to be on par with GPT-3 for accuracy ? and also toxicity. A key difference from GPT-3 is a stronger focus on languages away from the otherwise dominant English language. Bloom can process 46 different languages, including French, Vietnamese, Mandarin, Indonesian, Catalan, 13 Indian languages (such as Hindi) and 20 African languages. BigScience collected numerous new datasets for this and is publishing full details on datasets, development and training of Bloom. The release falls under the Responsible AI License developed by BigScience, which prohibits the use of Bloom in areas such as law enforcement, healthcare, or deception. However, unlike OpenAI, for example, BigScience has no way to effectively prevent misuse because the
> model is available directly and not through an interface. Bloom is now expected to serve as the foundation for numerous applications and, more importantly, research projects that create alternative AI applications away from the big tech companies.

[3] `https://mixed-news.com/en/bloom-is-a-real-open-source-alternative-to-gpt-3/`

The summary and keyphrases generated by Textstar. The resulting textgraph for this article contains 153 nodes and a fragment of it is shown in Figure 4.

> **Summary:** BigScience Bloom is open science and open source. Bloom was trained over 117 days at the supercomputing center of the French National Center for Scientific Research and is 176 billion parameters in size. The development involved over 1000 volunteer researchers, organized in the BigScience project, coordinated by Hugging Face, and co-funded by the French government. Bloom is now expected to serve as the foundation for numerous applications and, more importantly, research projects that create alternative AI applications away from the big tech companies.
>
> **Keyphrases:** *'National Center', 'Center Scientific', 'Scientific Research', 'Now true alternative', 'volunteer researchers', 'BigScience project'*

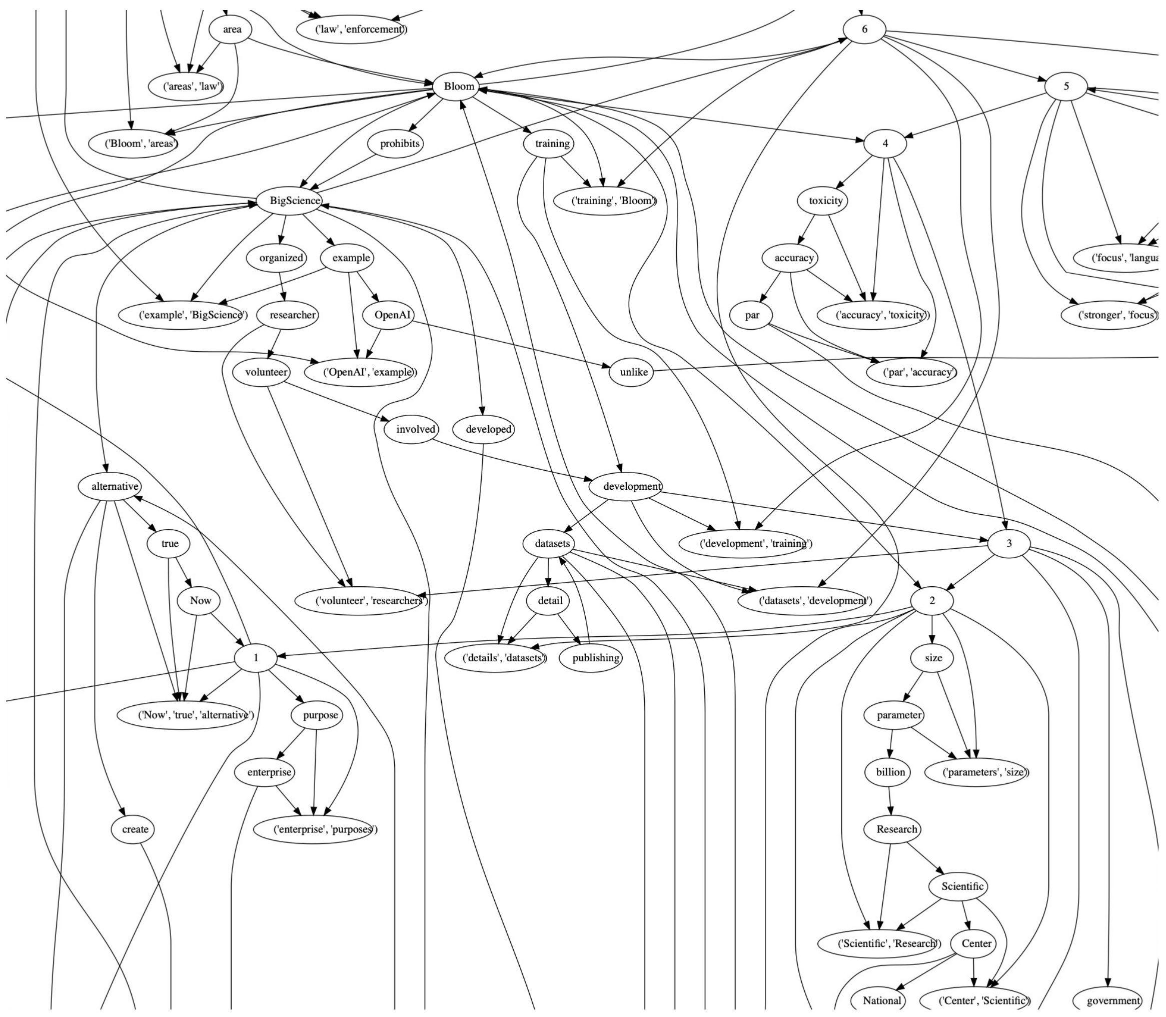

Figure 4: Fragment of the News Article Textgraph