N-Grams TextRank : A Novel Domain Keyword Extraction Technique

Saransh Rajput **Akshat Gahoi** **Manvith Reddy** **Dipti Mishra Sharma**

Language Technologies Research Center
International Institute of Information Technology, Hyderabad, India

{saransh.rajput,akshat.gahoi}@research.iiit.ac.in
manvith.reddy@students.iiit.ac.in
dipti@iiit.ac.in

Abstract

The rapid growth of the internet has given us a wealth of information and data spread across the web. However, as the data begins to grow we simultaneously face the grave problem of an *Information Explosion*. An abundance of data can lead to large scale data management problems as well as the loss of the true meaning of the data. In this paper, we present an advanced domain specific keyword extraction algorithm in order to tackle this problem of paramount importance. Our algorithm is based on a modified version of TextRank(Mihalcea and Tarau, 2004) algorithm - an algorithm based on PageRank(Page et al., 1998) to successfully determine the keywords from a domain specific document. Furthermore, this paper proposes a modification to the traditional TextRank algorithm that takes into account bigrams and trigrams and returns results with an extremely high precision.

We observe how the precision and f1-score of this model outperforms other models in many domains and the recall can be easily increased by increasing the number of results without affecting the precision. We also discuss about the future work of extending the same algorithm to Indian languages.

1 Introduction

Graph based ranking algorithms have proved to be useful for tasks which involve ranking or ordering. This includes important tasks like citation analysis and ranking webpage results. Graph based ranking Algorithms are used in many key areas even today. PageRank, an algorithm developed by the founders of Google, was the primary algorithm used to rank webpage searches until 2018.

The fundamental idea behind any graph based ranking algorithm is make use of global knowledge for making local decisions. To determine the importance of a node in a graph, we recursively look at other nodes to gain more information.

More recently, the applications of graph based algorithms have extended to other domains as well, including Natural Language Processing. This includes the use of Textrank algorithms for summarisation, word sense disambiguation(Mihalcea et al., 2004) and keyword extraction tasks. Knowledge extracted from the whole text is considered while making local decisions.

In this paper we introduce and evaluate an unsupervised approach for the task of domain terminology extraction. We employ the Textrank algorithm for this task with a few modifications. Taking into account that domain terms are often multi-worded expressions, we consider bigrams and trigrams as nodes in the graph with suitable additional weight to these nodes. Furthermore, terms are filtered based on their POS(Manning, 2011) tags in order to remove excessive domain-less words.

2 Pre-processing and Data

For our study we used a dataset that contained a collection of over 800 domain specific documents. The dataset featured documents from 4 distinct domains namely : **Bio-Chemistry, Communication, Computer Science and Law**.

Before passing a document through the model, it was crucial to carry out fundamental preprocessing in order to achieve a high standard of results. We first removed non-essential punctuations and tokenized the the document. In addition to the elementary NLTK(Bird, 2006)/Spacy Stop word list, we curated an additional specific list of common words that we observed added no meaning to our algorithm. POS Tagging(Brants, 2002) was a critical part of our model and was based on the powerful assumption that if a term is domain specific then it is often a **Noun** or a **Verb**, which we made after

Proceedings of the 17th International Conference on Natural Language Processing: TermTraction 2020 Shared Task, pages 9–12
Patna, India, December 18 - 21, 2020.

analyzing the data meticulously. The addition of a POS tagger gave us a significant increase in the f1-score.

3 TextRank

TextRank is a graph-based ranking model(like HITS(Kleinberg, 1999)) for text processing which can be used in order to find the most relevant keywords in a text. TextRank is an algorithm based on PageRank which we will explain briefly.

PageRank is an algorithm used for computing a ranking for every web page based on the graph of the web and helps in measuring the relative importance of specific web pages. We can take all web pages to be directed graph(Georgiadis et al., 2014). In this graph, a node is a webpage. If webpage A has the link to web page B, it can be represented as a directed edge from A to B. After we construct the whole graph, we can assign weights for web pages by the following formula.

$$S(V_i) = (1-d) + d * \sum_{j \in In(v_i)} \frac{S(V_j)}{|Out(V_j)|} \quad (1)$$

where S represents the weight of a webpage, d represents a damping factor, $In(V_i)$ represents the set of nodes having an edge directed to node i and $Out(V_i)$ represents the set of nodes which have an incoming edge from node i.

TextRank is conceptually the same algorithm as PageRank with the difference being that nodes in the graph are words rather than webpages. In order to find relevant keywords, the TextRank algorithm constructs this word graph. The graph is constructed by looking which words follow one another. An edge is set up between two words if they are located within a window of a size of our choice, the link gets a higher weight if these 2 words occur more frequently next to each other in the text. As we can see, preprocessing plays a huge part in the TextRank algorithm without which the results would easily be skewed towards common stop words and punctuations.

4 Our Implementation

In our model(Code can be found here [1]), we began the process by passing a document through our preprocessing pipeline. An input document was first split into sentences on the basis of end of sentence punctuation marks and then further tokenized using the SpaCy tokenizer. Stop Words were filtered out using the SpaCy stop words list along with our extensive custom list of domain-less terminology. Furthermore, we use the powerful tool of POS tagging in order to filter out irrelevant words and make the computation of our model much quicker. After extensive research we determined that **Nouns** and **Verbs** contained most of the important domain related terminology.

We are now left with tokenized sentences of each document. We further extract Unigrams, Bigrams and Trigrams from these documents and take them as seperate nodes in our TextRank Graph. We initialize our TextRank graph with a window-size of 4, which means that 4 words around every n-gram will be considered eligible to have an edge with the n-gram. After extensive trial and error, we observed that using a larger window size significantly increased the execution time of the model without much improvement in results. In some cases, an edge was added between two totally unrelated nodes due to large window size. Inspecting equation will reveal that we need to set a damping factor to assign how much relative importance to give the score calculated by the graph. After multiple runs, we achieved the greatest results with a damping factor of 0.85 which gives great importance to the score churned out by the TextRank Graph. We further set our convergence threshold at 10e-5 for our termination condition.

Once built, this graph is then used to calculate weights for each node. The weight of a node essentially represents its contribution to the document. We observed that although multi word domain terms are important to the document and contribute significantly, they are usually less frequent. Often they can get replaced by pronoun terms as well. In order to counter this neglection of the n-gram nodes, we introduce a novel weighting system to the traditional TextRank Algorithm. The weights of Bigram and Trigram nodes are taken as a parameter to our model and our multiplied to the final score returned by the traditional TextRank Algorithm. This imporves our results astronomically with many key bigrams and trigrams showing up as a result. These weights are useful in the hands of a

[1] https://github.com/akshatgui/Domain_Teminology_Extraction

Domain	Run	Precision	Recall	F-1
Law	$Run1$	**0.4**	**0.32**	**0.355**
	$Run2$	0.266	0.32	0.29
	$Run3$	0.133	0.285	0.181
$Communication$	$Run1$	**0.25**	0.208	0.227
	$Run2$	0.233	**0.291**	**0.259**
	$Run3$	0.1	0.125	0.111
$ComputerScience$	$Run1$	0.251	0.13	0.174
	$Run2$	0.3	0.134	0.185
	$Run3$	**0.466**	**0.152**	**0.229**
$Bio-Chemistry$	$Run1$	**0.501**	0.131	0.208
	$Run2$	0.3	0.173	0.219
	$Run3$	0.466	**0.184**	**0.264**

Table 1: Results for the task of Domain Extraction for different Bigram and Trigram Weights

domain expert who would be able to determine the right weights for each domain in order to get the best results.

Across the many domains, our Law Domain results were of extremely high quality and showed that both high precision and recall can be attained by our model.

5 Results and Evaluation

The model was tested on 10 domain specific documents of **Bio-Chemistry, Communication,Computer Science and Law**. We ran three different runs of the model, each with varying bigram and trigram weights. Run 1 featured a bigram weight of '1.8' and trigram weight of '1.5'. Run 2 featured a bigram weight of '1.8' and trigram weight of '2.5'. Run 3 featured a bigram weight of '1.8' and trigram weight of '2.5' along with lemmatization.

The precision we got for each domain is very high. Although we attained a much lower recall score, this was mostly due to the fact that we returned only 20 results for each document. Increasing the number of terms our model would return would drastically increase Recall. We further noticed that Recall is often document specific. Low Recall was observed in Computer Science and Bio-Chemistry which both featured much longer documents. We came to a conclusion that adjusting the number of results produced by our model to be a function of the length of the document would be a great idea.

Across the many domains, our Law Domain results were of extremely high quality and showed that both high precision and recall can be attained by our model.

Observing our results for various runs, we can see varied results for each domain. We notice that weights for bigrams and trigrams can be changed according to the domain in order to attain the best results. Domain Knowledge will help in particular, as a domain specialist can identify whether a specific domain will be having more bigrams or trigrams than unigrams so that there weights can be increased.

One particularly interesting thing to note was that the longer papers did much better when they were preprocessed with a lemmatizer. In domains like law and communication where we observed that words with same roots are used in different places differently, we see a negative impact of lemmatisation.

6 Future Work

We have evaluated the model for English, but the model can be made to work with pretty much any language with very minor modifications. Since we are considering the words as nodes and using a graph based approach to assign weights to these nodes, we are not concerned with what the word means or represents. We're only interested in its node weight. This allows our model to work with pretty much any language, given that tools for preprocessing text in that language are available. Restrictions can be made on number of letters for a specific language. Furthermore, transliteration can help to improve our scores. Transliterating the documents can help the results if a script of a language is unknown to the system. It will help to clearly distinguish between similar looking words. Further study of bigrams and trigrams weight in a specific language will also help the model. Ideally, the model should be able to learn the factor by which the bigram and trigram node weights need to be bumped. These bumping factors will not be same

across different domains or even different documents, since it is not a reasonable assumptions that different documents will have a similar distribution of multi word terms.

References

Steven Bird. 2006. NLTK: The Natural Language Toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 69–72, Sydney, Australia. Association for Computational Linguistics.

Thorsten Brants. 2002. Tnt: A statistical part-of-speech tagger. *ANLP*.

Loukas Georgiadis, Giuseppe Italiano, Luigi Laura, and Nikos Parotsidis. 2014. 2-vertex connectivity in directed graphs.

Jon M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632.

Christopher D. Manning. 2011.

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, Barcelona, Spain. Association for Computational Linguistics.

Rada Mihalcea, Paul Tarau, and Elizabeth Figa. 2004. PageRank on semantic networks, with application to word sense disambiguation. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 1126–1132, Geneva, Switzerland. COLING.

Larry Page, Sergey Brin, R. Motwani, and T. Winograd. 1998. The pagerank citation ranking: Bringing order to the web.