Keyword Extraction for Academic Papers

Andrew Giugliano, Brandon Oyer, Nick Pangori, and Haemin Park
agiuglia@umich.edu, broyer@umich.edu, npangori@umich.edu,
haeminp@umich.edu
University of Michigan

Abstract

Keyword Extraction has become an increasingly important topic in today's world as the amount of data grows exponentially everyday. Reading and summarizing the contents of large entries of text into a small set of topics is time consuming for a human, and as a result, automated systems are used commonly to complete this task. We specificly take interest in the domain of academic papers, where there is a large corpus of semi-structured documents. Effectively leveraging the power of Keyword Extraction would allow the ability to efficiently read the summary of several potentially interesting documents in a short amount of time.

We would like to present an algorithm that can efficiently output the top k keywords of a given academic paper based on their weighting score. The algorithm will be based upon Term Frequency and Document Frequency weighting schemes, and as such will also utilize a corpus of academic papers to extract Document Frequency values. In our research, we examined various different existing weighting schemes to find the most effective one for this domain. We additionally leveraged the general structure of academic papers by creating a weighted Term Freqeuency based upon where the term shows up.

Ultimately, we found that the Term Frequency weighting leads to higher Precision and Recall scores, and that the TF-IDF weighting scheme mixed with high weighting for the Abstract and Conclusion sections lead to the best Precision/Recall scores.

1 Introduction

Reading and summarizing the contents of large entries such as academic papers into a small set of topics is difficult and time consuming for a human. As a result, our solution to this problem is to create an automated system - keyword extraction - that extracts terms from a set of documents that are in some way more relevant than others in characterizing the corpus'domain.

Keyword extraction is a valuable technique for information retrieval and Web search. Keyword extraction is the task of isolating the most relevant words or phrases in a given text. Because a keyword is the smallest unit of measure, which can aptly express meaning of a document or series of documents, numerous applications can suitably take advantage of keyword extraction such as automatic indexing, text summarization, classification, text mining, filtering, cataloging, and topic detection. The applications seem endless, with the driving force between all subfields as reducing time. By systematically extracting essential keywords from an article, a student can easily choose what to read or learn based on personal relevance. Instead of filing through thousands of possible articles, he / she can scan the keyword list and significantly reduce time. This problem however is challenging due to the intricate complexities of natural language, as well as the inherent difficulty

in determining if a word or set of words accurately represent topics present within the text. [1] Thus, developing an algorithm using some heuristic such as already existing term-weighting schemes and locating and defining a set of words that accurately convey themes or describe information contained in the text will be the key of our project.

In our research we focus on the domain of academic papers. This is an important application; as there is an extremely large corpus of academic papers on the internet and it is crucial to effectively and efficiently filter through this corpus to find relevant documents, but it is also very interesting since we know that most academic follow a similar structure (i.e. Abstract, Results, Conclusion, etc.) and introduces an opportunity to create an algorithm that leverages this structure.

2 Keyword Extraction Algorithm

When finished, we would like to create an algorithm that can identify keywords within a single document of text. The purposes of these keywords are to identify the underlining concepts or meanings within the document in fewer words.

Identifying these keywords within a document of text will be our main challenge in the project. We want to be able to identify key concepts within a document; which optimistically will be a unique word on its own, but mentioned often within the document. As described in Section 4, we are using the SemEval dataset, which is made up of academic research papers. In this context, our main focus would be to identify the key concepts discussed in the paper using keywords.

Our initial method to identify keywords is to look for words that occur often in a specific document, but are used sparingly otherwise. To map this concept to our dataset, we would want to look for words that occur many times in one of the documents, but are not used very often in the other documents in our dataset. Because the dataset we are using consists of academic research papers, we would like to focus on distributing different weights for terms in certain sections of the paper. For instance setting higher weights for introduction section than abstract section of the research paper. In order to create such algorithm, we would like to use a term weighting scheme, as described in Section 3, to identify key statistics that we can use to implement this method. Moreover, we would like to analyze how applying certain weighting schemes introduced in Salton and Buckley[8] can affect the accuracy of our algorithm.

3 Related Work

The field of keyword extraction can be generalized to a number of domains, with different algorithms being used.

In Li, et all[2], the authors extend the tfidf algorithm to identify keywords in Chinese newspapers. They also build in features specific to the Chinese language, such as parts of speech and morphology, and features specific to a general newspaper such as the position and frequency of the word.

In Peng, et all[3], the author specifically researches keyword extraction within the context of research papers. Instead of using tf-idf, they used a machine learning algorithm, specifically built with conditional random fields.

In Hulth[4], the author uses a supervised machine learning algorithm, and uses a combination of both document statistics (such as term frequency and n-grams) and linguistic features. Hulth found that the combination of the two results in a better accuracy.

In Lee, et all[5], the author focuses on keyword extraction in online news articles. The author chooses to use tfidf as an initial algorithm to identify keywords, however throughout the paper also evaluates variants to tf-idf. The author finds that a variant name Table Term Frequency, which relies on computing individual tf-idf scores as well as evaluating the top n documents as a whole, was the most accurate algorithm in extracting keywords from news articles.

In Liliana Medina in Politics[6], the author focuses on keyword extraction from presidential speeches. She uses the toolkit JATE, which of-

fers a number of key extraction algorithms implemented in Java and chooses C-Value to extract keywords from a set of speech transcripts by 12 presidents of the United States. The important of C-value that the author states is that it does not need a refernce corpus and can extract multi-world keywords. It is a method that combines a term-frequency based approach with an inspection of frequencies of a term used as part of a larger term.

Lastly, in Ohsawa, et al, the author attacks the problem from a different angle. While most keyword extraction methods rely on statistical information gathered from term occurenc frequency in the document, he uses this method, called *KeyGraph*, which relies on clustering of related items into graphs to determine which words in a document are representative of the document's content [7].

4 Dataset

We are using the SemEval dataset, a corpus of technical, academic documents. You can find the dataset 1. We believe that SemEval is a good dataset to test our algorithm against, because these scientific documents have crystal clear meanings and summarizations, rather than a stream of social media observations, for example.

4.1 Dataset Makeup

SemEval is made up of 244 total documents, all of them being academic papers. It is split (by the creator) into 3 different sets; test, train and trial. There are no immediate differences in the 3 subsets, however we will likely take advantage of this split when training our algorithm, then testing it out on unseen trial data.

All documents are already annotated (by paper authors and readers), and we will use these when training out algorithm. Further along, we will also use this to judge the accuracy of our algorithm on trial data

5 Method Description

5.1 Overview

The goal of any keyword extraction algorithm is to highlight the most significant words in a given text. We define a texts most significant words as a set of words that, together, can give the reader a very good idea of the entire texts inherent meaning at a glance.

The effectiveness of a certain keyword extraction algorithm is dependent on the implementation of the term-weighting schemes as well as the nature of the document collection that the algorithm is run against. For example, a keyword extraction algorithm using a certain term-weighting combination will react differently (in terms of efficiency) to a dataset comprised of aerospace engineering manuscripts than to a collection of posts gathered from social media.

We will evaluate our algorithm using the SemEval dataset, a corpus of highly technical documents. Our aim is to derive an algorithm that maximizes precision and accuracy in the SemEval dataset; we are not interested in the efficiency of the algorithm in other domains.

We will first train our algorithm with the training set, in which the algorithm will add all potential keywords (i.e. not stopwords) to an inverted index and keep track of necessary statistics, such as the collection frequency and overall raw frequency. This training process is crucial; we learn which words are common in the collection and can be weighted to a lesser degree, whereas less frequently occurring terms will be given a higher weight.

After the extraction algorithm is trained from the training documents, we can test the algorithm, applying it to the test documents. In this set of testing documents, we will highlight the words that have been assigned the highest weight according to our implementation. With some tinkering, we will set a threshold weight at which all words above that weight are deemed keywords.

All documents have been annotated beforehand; meaning the author has already set aside the words that give the paper inherent meaning, the keywords. This allows us to, after training

and testing, judge our algorithms precision and recall based upon the fraction of words that have been returned by the algorithm over those in the annotated answer key. Precision and recall will be calculated accordingly.

We will use a number of different term-weighting schemes and choose among them the ones that yield the maximum precision and recall. Trial and error will be a large part of our methodology.

Our algorithm will then output the terms with the heighest weighting scores. For example, if we were using TF-IDF, then the algorithm will print out the top k keywords with the heightest k TF-IDF scores. The following is a description of the general pipeline of our algorithm:

1. A document (academic paper) is given as input
2. The document is then removed of stopwords, stemmed (using the Porter Stemmer) and tokenized
3. Given a pre-indexed corpus, the algorithm will use both the document and corpus to compute the potential keywords
4. Finally, the output will be the keywords with top k weighting scores

5.2 Review of Salton and Buckley

After reviewing Salton and Buckleys [8] paper thoroughly, as a point of reference for various weighting schemes, we were able to narrow down several schemes which are extremely accurate across a range of dataset types including aerospace engineering, computer engineering, and medicine. We were also able to find some promising schemes that, while not immediately yielding the best results, can be tweaked and tested beyond what Salton and Buckley did.

Because the tfc-nfx schemes results arent tied to document length, it will be a good point of comparison as we begin to develop a system designed specifically for SemEval. We hope to use it to find a good middle ground between normalizing on length, normalizing on lengths specific

Table 4. Performance results for eight term-weighting methods averaged over 5 collections

Term-weighting methods	Rank of method and ave. precision	CACM 3204 docs 64 queries	CISI 1460 docs 112 queries	CRAN 1397 docs 225 queries	INSPEC 12,684 docs 84 queries	MED 1033 docs 30 queries	Averages for 5 collections
1. Best fully weighted (*tfc-nfx*)	Rank	1	14	19	3	19	11.2
	P	0.3630	0.2189	0.3841	0.2626	0.5628	
2. Weighted with inverse frequency *f* not used for docs (*txc-nfx*)	Rank	25	14	7	4	32	16.4
	P	0.3252	0.2189	0.3950	0.2626	0.5542	
3. Classical tf × idf No normalization (*tfx-tfx*)	Rank	29	22	219	45	132	84.4
	P	0.3248	0.2166	0.2991	0.2365	0.5177	
4. Best weighted probabilistic (*nxx-bpx*)	Rank	55	208	11	97	60	86.2
	P	0.3090	0.1441	0.3899	0.2093	0.5449	
5. Classical idf without normalization (*bfx-bfx*)	Rank	143	247	183	160	178	182
	P	0.2535	0.1410	0.3184	0.1781	0.5062	
6. Binary independence probabilistic (*bxx-bpx*)	Rank	166	262	154	195	147	159
	P	0.2376	0.1233	0.3266	0.1563	0.5116	
7. Standard weights cosine normalization (original Smart) (*txc-txx*)	Rank	178	173	137	187	246	184
	P	0.2102	0.1539	0.3408	0.1620	0.4641	
8. Coordination level binary vectors (*bxx-bxx*)	Rank	196	284	280	258	281	260
	P	0.1848	0.1033	0.2414	0.0944	0.4132	

Table 5. Performance results for NPL collection (11429 docs, 100 queries)

Evaluation	Best fully weighted *tfc-nfx*	Weighted restricted *f* *txc-nfx*	Classical tf × idf *tfx-tfx*	Best probabilistic *nxx-bpx*	Classical idf system *bfx-bfx*	Binary independence *bxx-bpx*	Standard weight *txc-txx*	Coordination level *bxx-bxx*
Rank	116	62	149	2	23	8	172	83
Average precision	0.1933	0.2170	0.1846	0.2752	0.2406	0.2596	0.1750	

Figure 1: Results from the Salton and Buckley paper detailing the performance of each weighting scheme

to SemEval, and not normalizing on length at all.

A slight variation on tfc-nfx also performed well. The txc-nfx is very similar to tfc-nfx, the only difference being that document terms are not weighted with inverse document frequency. This will be a good data point to look to when determining specific term weights to use, as compared to tfc-nfx as well as tf-idf (the scheme from which both of the referenced schemes are loosely derived).

One scheme which showed potential promise only for certain datasets is the weighted probabilistic method nxx-bpx. While inferior throughout Salton-Buckley as compared to all other schemes, it primarily failed on the CISI and INSPEC datasets, which are characterized by long query vectors and a need for term discrimination (given by query term weighting).

Nxx-bpx will be an exciting scheme to at-

tempt to tweak very specifically for our dataset. It performed extremely well on the CRAN dataset, which is comprised of highly technical documents and meaningful terms, just as SemEval is.

Ultimately, for our research, we decided to test the tfidf, txctxx, nxxbpx and bxxbpx weighting schemes.

5.3 Weighted Term Frequency

In our research, we would like to introduce a techinique that will leverage the structure of academic papers. We wanted to maintain using the general form of the existing weighting schemes discussed in the previous section, but alter the formula in a slight way that would allow us to make the algorithm specific for our domain. Therefore, we created a slightly differentiated version of Term Frequency that will weight the TF for a given term differently when it appears in different sections of the paper. Generally, the formula for TF will look like this, given section s:

$$TF = \sum w_s * TF_s$$

We specifically divided each paper into the three different subsections: Abstract, Body and Conclusion. Abstract and Conclusion maps to their respective in an academic paper, while the Body contains all text between the Abstract and Conclusion. Other text such as References were not considered. Therefore, given weight α for the Abstract section a, β for the Body section b, and γ for the Conclusion section c, we used the following formula in our research:

$$TF = \alpha * TF_a + \beta * TF_b + \gamma * TF_c$$

For example, if we found the word "retrieval" 2 times in the abstract, 2 times in the body and 1 times in the conclusion, and given $\alpha = 3, \beta = 1, \gamma = 2$, the TF would be computed as $TF = 3 * 2 + 2 * 1 + 1 * 3 = 11$.

6 Experiments and Discussion

6.1 Evaluation Methodology

To evaluate and find the best weighting scheme, we ran our algorithm on the documents in SemEval test folder using an approach where all but the given document would be used to construct the corpus, and the algorithm would extract the top k keywords from the document. In our research, we fixed k to be $k = 10$. We computed Precision and Recall at rank 10 between the extracted keywords from the algorithm and the annotated keywords in given in the SemEval datatset. Then, for a given weighting scheme, we macro-average the Precision and Recall amongst all the documents for which we run the algorithm on.

6.2 Results

α, β, γ	TFIDF	TXCTXX	NXXBPX	BXXBPX
1,1,1	0.320/0.127	0.391/0.156	0.391/0.156	0.029/0.011
2,1,1	0.398/0.157	0.413/0.163	0.420/0.166	0.123/0.046
3,1,1	0.427/0.169	0.417/0.165	0.420/0.166	0.123/0.046

Table 1: Precision/Recall scores for different α weights

α, β, γ	TFIDF	TXCTXX	NXXBPX	BXXBPX
1,1,1	0.320/0.127	0.391/0.156	0.391/0.156	0.029/0.011
1,2,1	0.320/0.127	0.391/0.156	0.391/0.156	0.032/0.012
1,3,1	0.320/0.127	0.391/0.156	0.391/0.156	0.032/0.012

Table 2: Precision/Recall scores for different β weights

α, β, γ	TFIDF	TXCTXX	NXXBPX	BXXBPX
1,1,1	0.320/0.127	0.391/0.156	0.391/0.156	0.029/0.011
1,1,2	0.382/0.152	0.400/0.159	0.407/0.161	0.096/0.036
1,1,3	0.401/0.159	0.403/0.160	0.407/0.161	0.096/0.036

Table 3: Precision/Recall scores for different γ weights

6.3 Term Frequency Weighting Effects

In Table 1, we keep β and γ fixed while manipulating α. We can see that weighting the Abstract higher increases or maintains precision and recall scores across all weighting schemes.

In Table 2, we keep α and γ fixed while manipulating β. We can see that weighting the Body higher maintains the precision and recall scores across all weighting schemes.

α, β, γ	TFIDF	TXCTXX	NXXBPX	BXXBPX
1,1,1	0.320/0.127	0.391/0.156	0.391/0.156	0.029/0.011
2,2,1	0.398/0.157	0.413/0.163	0.420/0.166	0.123/0.046
3,2,1	0.427/0.169	0.417/0.165	0.420/0.166	0.123/0.046
3,2,2	0.463/0.182	0.428/0.169	0.431/0.170	0.221/0.086
2,3,1	0.398/0.157	0.413/0.163	0.420/0.166	0.123/0.046
2,3,2	0.437/0.172	0.423/0.168	0.431/0.170	0.221/0.086
3,3,1	0.427/0.169	0.417/0.165	0.420/0.166	0.123/0.046
3,3,2	0.463/0.182	0.428/0.169	0.431/0.170	0.221/0.086
1,2,2	0.382/0.152	0.400/0.159	0.407/0.161	0.110/0.041
1,2,3	0.401/0.159	0.403/0.160	0.407/0.161	0.110/0.041
2,2,3	0.447/0.176	0.427/0.169	0.431/0.170	0.221/0.086
1,3,2	0.382/0.152	0.400/0.159	0.407/0.161	0.110/0.041
1,3,3	0.401/0.159	0.403/0.160	0.407/0.161	0.110/0.041
2,3,3	0.447/0.176	0.447/0.176	0.431/0.170	0.221/0.086
2,1,2	0.437/0.172	0.430/0.168	0.431/0.170	0.221/0.086
2,1,3	0.447/0.176	0.427/0.169	0.431/0.170	0.221/0.086
3,1,2	0.463/0.182	0.428/0.169	0.431/0.170	0.221/0.086
3,1,3	0.472/0.185	0.431/0.170	0.431/0.170	0.221/0.086
3,2,3	0.472/0.185	0.431/0.170	0.431/0.170	0.221/0.086

Table 4: Precision/Recall scores, manipulating all 3 weights

In Table 3, we keep α and β fixed while manipulating γ. We can see that weighting the Conclusion higher increases or maintains precision and recall scores across all weighting schemes, however the effect is not as extreme as that of inceased Abstract as in Table 1.

In Table 4, we manipulate α, β and γ to get a mix of weighting. In general the highest performering weighting had high Abstract and Conclusion weights, while keeping Body low.

In general, we see that higher weighting for Abstract and Conclusion combined with lower weighting for the Body leads to the best Precision and Recall scores. We see that (3,1,3) and (3,2,3) worked the best.

6.4 Weighting Scheme Effects

Overall, we see that TFIDF and NXXBPX work the best. We see that NXXBPX has the edge when considering an unweighted TF scheme (or in other words, all sections are weighed the same), however with the optimal TF weights we find that TF-IDF works better for both Precision and Recall. Besides this, in general, all five weighting schemes reacted similarily to changes in TF weights.

Therefore, we put forth that within our research, TFIDF is the most effective weighting scheme when considering Precision and Recall. In fact, TF-IDF had the two highest Precision/Recall scores with weights (2,1,2) and (3,1,3).

7 Conclusion

Overall, we can see that our main contribution of Term Frequency weighting leads to better Precision and Recall scores across all weighting schemes considered here. We specifically see that higher weights for the Abstract and Conclusion, while keeping the weight for Body low, leads to the best Precision and Recall scores.

Another main part of our research here has been finding the optimal weighting scheme from the different variants suggested in Salton and Buckley[8]. After considering 5 variants from the paper (TFIDF, TXCTXX, NXXBPX, and BXXBPX), we find that TF-IDF works the best with our Term Frequency weighting.

7.1 Future Work

There are several different directions we could go in within the context of this project.

Since our research here was based on experimenting with the paramters of the different weighting schemes and different sections for weighted TF, we could delve deeper and experiment with more weighting schemes, alternative weights (such as 0 to completely unconsider a given section, or -1 to negatively associate a given section) or we could even utilize alternative sections to weight (we could break apart Body into it's different subsections).

In the context of academic papers, we might consider alternative weighting schemes, such as those referred to in Section 3, or even possibly leveraging machine learning algorithms on domain-specific knowledge to get a more effective algorithm.

Considering Weighted Term Frequency, we might consider leveraging this on other domains besides academic papers; specifically those that might be semi-structured in their nature.

8 Contributions

- Andrew Giugliano worked on the checkpoints and the final paper.
- Brandon Oyer worked on the checkpoints and implementing the algorithm.

- Nick Pangori worked on the checkpoints and implementing the algorithm.
- Haemin Park worked on the checkpoints and the final paper.

9 References

[1] Brian Lott. Survey of Keyword Extraction Techniques. *University of New Mexico*, 2012.

[2] LI Juanzi, F. Q. Keyword Extraction Based on tf/idf for Chinese News Document. *Wuhan University Journal of Natural Sciences ,* 2007.

[3] Peng, F. Accurate Information Extraction from Research Papers using Conditional Random Fields. *Information Processing & Management ,* 2006.

[4] Hulth, A. Improved Automatic Keyword Extraction Given More Linguistic Knowledge. *EMNLP '03 Proceedings of the 2003 conference on Empirical methods in natural language processing.* Association for Computational Linguistics, 2003.

[5] Sungjick Lee, H.-j. K. News Keyword Extraction for Topic Tracking . *Fourth International Conference on Networked Computing and Advanced Information Management,* 2008, IEEE Computer Society.

[6] Liliana Meedina. Automatic Keyword Extraction from Presidential Speeches.*Liliana Medina in Politics.Automatic Keyword Extraction from Presidential Speeches,* 2015.

[7] Yukio Ohsawa, Nels E. Benson, and Masahiko Yachida. Keygraph: Automatic indexing by co-occurrence graph based on building construction metaphor. In *Proceedings of the Advances in Digital Libraries Conference*, ADL 98, pages 12, Washington, DC, USA, 1998. IEEE Computer Society.

[8] Gerard Salton and Christopher Buckley (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management.* Volume 24, Issue 5 (August 1988), pages 513-523.