Keyword Extraction from Abstracts and Titles

Rekha Bhowmik
Department of Computer Science, Sam Houston State University, Huntsville, TX 77341
rbb002@shsu.edu

Abstract

Keywords are very important for any academic paper. We propose the Perceptron Training Rule for keyword extraction from titles and abstracts. We present a system for generating keywords which relies on weights of words in a sentence. The system generates keywords from academic research articles by selecting the most relevant keywords. We compare the keywords generated by our system and those generated by cluster analysis to the keywords given by the authors and analyze the results based on full-keyword matches, partial-keyword matches and no-keyword matches.

1. Introduction

For any research article people look for titles, abstracts, keywords. This enables them to decide whether to read the entire text. In this paper, we propose a model that applies the *Perceptron Training Rule* for extraction of keywords only from abstracts and titles. Here, we have not studied The *Delta Rule* or the *Back-Propagation Rule* for keywords extraction. Also, the focus is mainly on extraction of keywords from research papers. We consider extraction from Word documents.

Keywords given by the author are very common in research articles and their purpose is to give the reader an idea of the article. Although, in general, the authors do not just collect the important phrases from their articles, they are either extracted from the original or with minor modifications .

The paper is organized as follows: Section 2 presents some of the previous work. Section 3 describes our approach to keywords extraction using Perceptron Training Rule. Section 4 gives our experimental results. In section 5, we include the summary and propose future work.

2. Related Work

Text summaries are becoming essential. Instead of having to go through an entire text, people can understand a text quickly and easily by means of a concise summary. The title, abstract and key words, if provided, can convey the main ideas, but they are not always present in a document.

Summarization is the process where an information is reduced to some keyword [18, 7]. Abstracts can be read quickly. Based on the abstracts, one decides to read the entire text. Some researchers have moved towards learning approaches[21], [23] that take advantage of training data. It describes another machine learning, Support Vector Machine (SVM) based summarization technique. Humans have an incredible ability to condense huge amounts of information and they are known as excellent summarizers. However, creation of summaries manually requires time. Thus, there has been an increase in research in the area of automatic text summarization.

A number of papers in text summarization appear in [7]. Text summarization is also surveyed in [11]. There are various classifications for automatic text summarization. One useful classification is based on the level of processing [7]. Text can be summarized as: (i) surface-level, : information is extracted by finding titles, keywords and phrases, (ii) entity-level, information is extracted by identifying important internal relationships between entities of text, example, similarity, syntactic relationships and logical relationships, and (iii) discourse-level, information is extracted by identifying the global structure of the text. A new keyword extraction algorithm based on a single document is presented in [24]. First frequent terms are extracted and then occurrences in the same sentences is generated. If a term appears with a particular subset of frequent terms, the term is likely to have a meaning.

Keyword extraction is an important technique for text retrieval, web page retrieval, summarization, text mining, text clustering[13]. Information that can be automatically extracted from text collections: probabilistic associations of keywords and prototypical document instances are presented. Natural Language Processing tools necessary for such extractions are also presented.

A keyword is either a single word (unigram), or a grouping of two or more words. For example, keyword extraction represents an important concept, example, 'machine learning', 'natural language processing'. Retrieving collocations from text has been presented in [3]. Techniques based on statistical methods for retrieving and identifying collocations from large textual corpora is described. These techniques produce collocations based on some original filtering methods that allow the production of higher-precision output. These techniques have been implemented and resulted in a lexicographic tool.

Automatic extraction of collocations has been presented in [12].

Many academic conferences require that each paper include paper's title, abstract and a set of keywords. The keywords provide information about the content of the paper and can be seen as a document abstraction. The concept of keyword is also widely used in search engines.

The basic idea for extraction of keywords for a given text is to build a list of words and collocations sorted in descending order according to their frequency. The algorithm filters general terms in the article using a stop-list of closed-class words such as articles, prepositions and pronouns. The most frequent terms are selected as keywords since the assumption is that the author repeats important words as he advances and elaborates. Examples of systems that applied this method for abstract-creation are described in [6] and [10, 11].

In [1] automatic learning of keywords for classification of documents stored in hierarchical classification structure is presented. Their learning method is based on the Bayesian classifier on text data, which is described in [20]. However, their method learns keywords from a set of general pre-defined keywords, which is human constructed and designed for web browsing.

A statistical method to learn visual keywords for image and video content-based retrieval is presented in [8]. His learning method relies on pre-annotated words and/or primitive visual features. He demonstrates his framework in retrieval of natural scene images. The method is not suitable for dealing with Academic papers that usually do not include image and video data.

A keyphrase extraction system which consists of a set of parametrized heuristic rules that are tuned to the training documents by a genetic program is proposed in [16]. The system suggests that about 80% of the extracted key phrases are acceptable to readers, much better from the results achieved by C4.5 decision tree induction algorithm as suggested in [9]. Keyphrase extraction based on the naïve Bayes Learning is proposed in [2]. They show that the extracted keyphrases improves significantly when domain-specific information is exploited. They claim that their learning method is quicker than the genetic learning applied by [16]. Good initial weights increase the accuracy and the speed of the learning process [4]. Also, good initial weights have an effect on the learning process[19].

An abstract consists of sentences. A sentence is considered to be a set of words separated by a stop mark (".", "?" or "!"). Two terms in a sentence are considered to occur once. Frequent terms can be obtained by counting term frequencies.

A clustering method based on the automatic extraction of the keywords of a Web page is proposed in [15]. The occurrence of common keywords is exploited to decide when it is appropriate to group pages. The page content as the basic feature used to cluster a Web site is considered. Summary information about a page is obtained by means of keyword extraction. The keyword with the highest score within each cluster is used as cluster label. Some terms occur with each other and clusters of terms are obtained by combining occurring terms. An adjacency matrix is an $N \times N$ matrix, where columns corresponding to frequent terms are extracted for calculation. The columns with low frequency terms occurrence are ignored.

The agglomerative hierarchical algorithm known as Johnson's algorithm is presented in [22]. The algorithm begins with the computation of number of occurrences between keywords. In each step, the two most similar clusters are joined. After $N-1$ steps (with N the number of entities), all entities are grouped into one cluster. Each level shows partition of clusters. To select the resulting clustering, a cut point has to be determined which breaks the single group into clusters of closely associated keywords. Thus, the steps are: i) begin with N clusters for an abstract, and compute distance between the sentences; ii) while there are more than one cluster: find the pair of clusters at minimum distance; merge these clusters into a new cluster; and update the distance measures between each pair of clusters. To update the distance measure between clusters, the *complete linkage rule* [14] is used. The rule states that the distance between the already existing cluster C and a new cluster, formed by joining clusters A and B, is the minimum between *dist*(C, A) and *dist*(C, B).

The procedure for clustering and determining adjacency matrix[17] involves four steps: i) develop the adjacency matrix between pairs of words. ii) find the largest number of occurrence between pairs of words from the adjacency matrix. This is the level to start with the cluster analysis procedure. Here the type of sentence is the level interval for dendogram. The pairs of words which fall in this level form a cluster and is designated by some cluster name for the purpose of identification. Decrease the level by the level interval chosen(which is one). Find the words which fall in this level. We go in for the third and subsequent levels by further reducing by the level interval. In this way, the words falling in a particular level are searched and identified by cluster name. iii) plot the dendogram v) Construct a intergroup adjacency matrix representing the communication costs between clusters. Thus, if cluster C_i is obtained by grouping of keywords belonging to the set I and C_j is another cluster representing the group of keywords belonging to the set J, the element Tij of the adjacency matrix of clusters is:

$$Tij = \sum_{k \varepsilon I} \sum_{l \varepsilon J} t_{kl}$$

The most simple approach to the extraction of the key-words of a text is based on locating the most frequent words in the text. The basic idea underlying this approach is that the most important concepts in the texts are likely to appear repeatedly, or, at least, more frequently than minor concepts.

We are mainly interested in three classes of complex terms: named entities, complex lexical units, and recurrent free phrases. Named entities are terms referring to individuals, locations, organizations, and dates. Complex lexical units are the kind of multi-word expressions that can be found in dictionaries (for instance phrasal verbs such as ”put on” and idiomatic expressions such as ”roller coaster”). Finally, the notion of recurrent free phrase was introduced by [2] to refer to a free combination of words which is recurrently used to refer to a concept. They are characterized by either high frequency in a reference corpus (e.g. ”American government”), high degree of association between words (e.g. ”first time”), or high salience (e.g. ”international summit”).

Titles, abstracts and keywords are the most common summaries in academic papers. Usually, the title, the abstract and the keywords are the first, second, and third paragraphs of an Academic paper, respectively. The title usually describes the main issue discussed in the paper. The abstract presents one with a short description of the background, the study and its results.

3. The Model

3.1. The Basic Algorithm

The system selects the *n* most highly weighted words as keywords. The value of *n* has been set at 9 because: (i) the number 9 is accepted as the maximum number of items that a person is able to remember, according to the cognitive rule called “magical number 7, plus or minus two”. This means that a person is capable of remembering approximately 5 to 9 information words over a relatively short term [5]. ii) also, we selected some of the abstracts with 9 keywords.

The system selects only keywords which are unigrams, bigrams or trigrams. This is because our database contains 2% keywords with four-keywords. Some statistics about the abstracts, type of keywords, is given in appendix.

The main steps of the model are:

For each abstract:

1. Take an article with headline, abstract. Build a word weight matrix for all unigrams, bigrams, and trigrams excluding high frequency closed class words via a stop list file.

The stop list consists of common function words such as determiners(a, the, this), prepositions(in, from, into, of, on), conjunctions(and, but, after, since, as), coordination (and, or).

To reduce the size of the word weight matrix, we transform each word to lowercase. At this step, the set of words of the title and abstract are filtered through this list of fluff words.

2. Compute the weights of all unigrams, bigrams, trigrams by counting full-keyword matches and partial-keyword matches.

Weights are calculated by counting full-keyword matches and partial-keyword matches. A full-keyword match is one which is exactly identical to authors’ keyword including singular/plural/abbreviations, first letter keyword as lowercase/ uppercase. In partial-keyword match, the original keyword and the system generated keyword have identical five-letter prefix. All the other pairs of words are considered no-keyword match. For example, the 8 words are regarded as partial-keyword matches because they have the same five-letter prefix “crypt”: “cryptanalysis”, “cryptanalyst”, “cryptographer”, “cryptology”, “cryptography”, “cryptanalytic” , “cryptosystems”, “cryptographically”.

3. Sort the types of words in descending order. Merge them and select the n highest weighted groups of words as the proposed keywords.

4. Compare the keywords generated by our system with the keywords defined by the author in terms of full-keyword matches, partial-keyword matches and no-keyword matches..

Here error-analysis and learning capabilities are checked. We identify words in the abstract that were chosen by the author as keywords but not by our system. That is, partial-keyword matches and no-keyword matches are identified and counted.

Figure 1. The basic algorithm

The basic idea in automatic extraction of keywords is that the most frequent terms are selected as keywords since the assumption is that the authors’ repeat important keywords. An additional criterion which is taken into account in many summarizing programs is the importance of the sentence from which the keyword is taken.

There are various methods to identify the importance of sentences, example: (i) position of sentence , (ii) sentences which include important words and phrases like *problem*, *our investigation* , *conclusions* (iii) analysis of text based on logical relationships, and (iv) structure and format text

(a) find the pair of clusters at least distance;

(b) merge these clusters into a new cluster;

outline.

We use frequency, importance of sentences, keyword length for extracting and learning keywords. A statistical analysis of the author keyword length as shown in appendix shows that bigrams are the most frequent, then unigrams, then trigrams, and fnally quadgrams .

Our first assumption is that the importance of a sentence is determined by: (i) the title of the paper, (ii) the first sentence of the abstract, (iii) the third sentence of the abstract, and (iv) any other sentence of the abstract. Our second assumption is based on the fact that most of the keywords should be bigrams, then unigrams and then trigrams. Therefore, their weights should be set accordingly.

3.2. The Learning Algorithm

The learning of the weights has been performed using Perceptron Training Rule as presented in [17]. Initially, we try with equal weights (value = 1) for all w[i][k] for $i = 1,2,3,4$ are the different types of sentences; $i = 1$ represents the title of the paper, $i = 2$ represents the first sentence of the abstract, $i = 3$ represents the third sentence of the abstract, and $i = 4$ represents any other sentence of the abstract, and $k=1,2,3$ is the keyword length.

The perceptron training rule is presented in Figure 2. Here, *w(i)'* is the new weight of sentence *i* , *w(i)* is the old weight, α represents a small constant between 0.0 and 1.0, δ represents the training value, *y* denotes the output value, and x_i represents the input value for the i^{th} iteration

$$w(i)' = w(i) + \alpha \cdot (\delta - y) \cdot x(i)$$

Figure 2. The Perceptron Training rule

The learning algorithm is presented in Figure 3. This is based on the basic algorithm presented in Figure 1.

```
for an abstract
for keyword $j$ and keyword length $k$ included by author
   for SentenceType $i$ keyword $j$ found in abstract
       AND keyword $j$ not selected by our model
            error[i][k] = error[i][k] + 1
```

Here, *j*= serial number of the keyword in the author's list of keywords
i = type of sentence
k = keyword length
error[i][k] = counter for errors (partial-keyword matches and no-keyword matches)

Figure 3. Partial-keyword and no-keyword matches

Figure 4 presents the learning rule which is based on the Perceptron Training Rule.

```
      for each value of $i$
            for each value of $k$
                 w[i][k] = w[i][k] + $\varepsilon \cdot$ error[i][k]
```

where, *w[i][k]* = weight of sentence
i = type of sentence
k =length of keyword
and, ε equals 0.07

Figure 4. The learning rule

4. Experiments

We tested our model for 90 academic papers. Each of these documents included title of the paper, abstract, and a list of keywords as given by the author. We have included some of the statistics in Appendix.

Initially we set the weight as *1*, for all values of *w[i][k]* where, *i* is the type of sentence, and *k* is the keyword length. This is performed for all combinations of $i = 1,2,3,4$ and $k = 1,2,3$. Table 2 presents the results of our basic algorithm with weight = 1.

4.1. Data Set

There exist large volumes of text data on the net, and it can be subjected to automatic summarization. We selected the research papers from i) Informatica An International Journal of Computing and Informatics (http://www.informatica.si/) and, ii) Dynamical & Evolutionary Machine Organization(DEMO) Demonstra: (http://www.demo.cs.brandeis.edu/papers/year.html). For each data source, we manually generated a model summary to be used in training and for evaluation.

For the test set, we selected 90 academic papers. Each document included the title of the paper, abstract, and a list of keywords included by the author.

4.2 Example

We demonstrate running our system without learning. Below we present two abstracts. The abstract, authors' keywords and the keywords proposed by are system is presented in Tables 1a and 1b.

Title: A Semantic Kernel to Classify Texts with Very Few Training Examples
Abstract: Advanced techniques to access the information distributed on the Web often exploit automatic text categorization to filter out irrelevant data before activating specific searching procedures. The drawback of such approach is the need of a large number of training documents to train the target classifiers. One way to reduce such number relates to the use of more effective document similarities based on prior knowledge. Unfortunately, previous work has shown that such information (e.g. WordNet) causes the decrease of retrieval accuracy. In this paper, we propose kernel functions to use prior knowledge in learning algorithms for document classification. Such kernels implement balanced and statistically coherent document similarities in a vector space by means of the term similarity based on the WordNet hierarchy. Cross-validation results show the benefit of the approach for Support Vector Machines when few training examples are available.

Table 1a: Keywords

	keywords
author keywords	kernel methods, similarity measures, support vector machines, WordNet
our system	automatic text categorization, classifiers, WordNet, kernel, document similarities, learning algorithms, Support Vector Machines

our clustering method	Training, automatic text categorization, WordNet, document similarities, learning algorithms, Support Vector Machines, term similarity, document similarities

	keywords	frequency
Full-keyword match	support vector machines, WordNet	2
Partial-keyword match	kernel methods, similarity measures	2
No-keyword match	automatic text categorization, classifiers, learning algorithms	3

Title: A Gradient Descent Method for a Neural Fractal Memory. ***Abstract:*** It has been demonstrated that higher order recurrent neural networks exhibit an underlying fractal attractor as an artifact of their dynamics. These fractal attractors offer a very efficient mechanism to encode visual memories in a neural substrate, since even a simple twelve weight network can encode a very large set of different images. The main problem in this memory model, which so far has remained unaddressed, is how to train the networks to learn these different attractors. Following other neural training methods this paper proposes a Gradient Descent method to learn the attractors. The method is based on an error function which examines the effects of the current network transform on the desired fractal attractor. It is tested across a bank of different target fractal attractors and at different noise levels. The results show positive performance across three error measures.

Table 1b: Keywords

	keywords
author keywords	neural networks, fractals, learning rules, gradient descent, dynamical systems, iterated function systems, recurrent neural networks.
our system	gradient descent, recurrent neural networks, attractors, error function, substrate, performance.
Text clustering method	gradient descent, recurrent neural networks, fractal, neural networks, attractors, error, method, encode.

	keywords	frequency
Full-keyword match	neural networks, recurrent neural networks, fractal, gradient descent	4
Partial-keyword match	error function	1
No-keyword match	attractors, substrate, performance	3

Table 2 shows that 57% were full-keyword matches and 14% were partial-keyword matches and 43% were no-keyword matches. This is because our algorithm proposes keywords only from title and abstract. Also, it depends on the abstract being selected. It has been found that only 40-50% keywords appear in abstracts.

4.3 Experimental Results

Table 2. Basic algorithm with equal weights

	tested keywords	full-keyword matches	partial-keyword matches	no-keyword matches
keywords	312	91	112	109
%	100	29.16	35.89	34.95

Table 2 shows that full-keyword matches 29.16%. Full-keyword and partial-keyword matches were found at 65.04%. Table 3a presents the learned weights with Perceptron Training Rule.

Table 3a. Learned weights

	unigrams	2-grams	3-grams
title-sentence	1.21	1.51	1.19
first sentence	1.40	2.14	1.40
third sentence	1.19	1.69	1.15
any other sentence	2.10	2.50	1.40

Table 3b presents the results based on our algorithm using the weights in Table 3a.

Table 3b. Results with learned weights

	tested keywords	full-keyword matches	partial-keyword matches	no-keyword matches
keywords	312	81	122	109
%	100	25.96	39.10	34.94

Table 3b shows that full-keyword matches were 25.96%. Partial-keyword matches and full-keyword matches were 65.06%.

We have selected reasonable weights as initial weights. These values are presented in Table 4a and are based on the hypothesis that the most important sentence for extraction of keywords is the title of the paper, then the first sentence of the abstract, the third sentence, and then any other sentence. Table 4b presents the results of our basic algorithm using the weights in Table 4a.

Table 4a. Initial weights

	unigrams	bigrams	trigrams
title-sentence	3	3	3
first- sentence	2	2	2
third sentence	2	2	2
any other sentence	1	1	1

Table 4b. Results with initial weights and without learning

	tested keywords	full – keyword matches	partial-keyword matches	no-keyword matches
keywords	312	99	122	91
%	100	31.73	39.10	29.17

Table 4b shows that full-keyword matches were 31.73%. Full-keyword and partial-keyword matches were 70.83%. These results are better than the results obtained with initial equal weights.

However, with Perceptron Training Rule with initial weights as given in Table 4a, successful results were achieved. The learned weights are shown in Table 5a and the results of the algorithm are shown in Table 5b.

Table 5a. Learned weights

	unigrams	bigrams	trigrams
title-sentence	3.08	3.57	3.29
first sentence	2.45	2.94	2.45
third sentence	2.17	2.87	2.17
any other sentence	2.31	2.80	1.47

Table 5b. Results with learned weights

	tested keywords	full-keyword matches	partial-keyword matches	no-keyword matches
keywords	312	97	112	103
%	100	31.09	35.89	33.02

Table 5b shows that full-keyword matches were 31.09%. Full-keyword and partial-keyword matches were 66.98%. These results are good when compared with the results obtained with equal initial weights as indicated in Table 2.

5. Conclusions

The results obtained from our model is good. These results are based on the fact that 64.44% of the abstracts do not include at least one of their own keywords, and (b) 57.37% of the authors' keywords do not appear in their own abstracts.

To obtain best results, extraction of keywords and learning of keywords should be performed on the paper rather than just the titles and abstracts.

Results indicate that the title of the paper is the most important sentence. Also, the initial weight of sentence is an important factor for the results obtained.

In future we would be interested in selecting an optimal initial weights, testing different learning techniques, and also extend our model for extracting and learning keywords from the entire research paper.

6. References

[1] D. Mladenic, and M. Grobelnik, "Assigning Keywords to Documents using Machine Learning", *Proceedings of the 10th International Conference on Information and Intelligent Systems*, 1999.

[2] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C.G. Nevill-Manning, "Domain-Specific Key-Phrase Extraction", *Proc. International Joint Conference on Artificial Intelligence*, Stockholm, Sweden: Morgan Kaufmann, 1999, pp. 668-673.

[3] F. Smadja, "Retrieving Collocations from Text" *Computational Linguistics,* 19(1), 1993, pp. 143-177.

[4] F. Van den Bergh , A. P. Engelbrecht, "Cooperative Learning in Neural Networks using Particle Swarm Optimizers", *South African Journal* , 26, 2000, pp. 84-90.

[5] G. A. Miller: The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity of Information. Psychological Science, 63, 1956, pp. 81-97.

[6] H. P. Luhn, "The Automatic Creation of Literature Abstracts", *IBM Journal of Research and development.* Reprinted in *Advances in automatic text summarization*, Cambridge, MA, MIT Press, 1999, pp. 159-165.

[7] I. Mani, and M. T. Maybury, *Introduction, Advances in Automatic Text Summarization,* Cambridge, MA, MIT Press, 1999.

[8] J-H Lim, "Learning Visual Keywords for Content-Based Retrieval", *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, 2, 1998, pp. 169-173.

[9] J. R. Quinlan, *C4.5: Programs for Machine Learning*", Morgan Kaufmann . 1993.

[10] K. A. Zechner, "Fast Generation of Abstracts from General Domain Text Corpora by Extracting Relevant Sentences", *Proceedings of the 16th International Conference on Computational Linguistics* , 1996, pp. 986-989.

[11] K. A. Zechner, "Literature Survey on Information Extraction and Text Summarization", Term Paper, Carnegie Mellon University. 1997.

[12] K. Kita, Y. Kato, T. Omoto, and Y. Yano, "Automatically extracting collocations from corpora for language learning", *Proceedings of the International*

Conference on Teaching and Language Corpora, 1994, pp. 53-64.

[13] M. Rajman, and R. Besancon, “Text mining – knowledge extraction from unstructured textual word”, *Proceedings of the 6th Conference of International Federation of Classification Societies,* Chapman and Hall, 1997.

[14] N. Anquetil and T. C. Lethbridge, “Experiments with Clustering as a Software Remodularization method”, *Proceedings of the Sixth Working Conference on Reverse Engineering*, Atlanta, IEEE Computer Society, October 1999, pp. 235–255.

[15] P. Tonella, F. Ricca, E. Pianta, and C. Girardi, “Using Keyword Extraction for Web Site Clustering”, *Proceedings of the Fifth IEEE International Workshop on Web Site Evolution*, 2003.

[16] P. Turney, “Learning Algorithms for Keyphrase Extraction”, *Information Retrieval Journal*, 2(4), 2000, pp.303-336.

[17] R. Bhowmik, “Allocating Clusters using Hierarchical Clustering Technique”, *Hawaii International Conference on Computer Sciences,* Hawaii, 2004.

[18] R. Alterman, “Text Summarization”, *Encyclopedia of Artificial Intelligence*, John Wiley, New York, 1992, pp.1579-1587.

[19] S. C. Kremer, “Lessons from Language learning”, *Recurrent Neural Networks: Design and Applications,* The CRC Press International Series on Computational Intelligence, 2000, pp. 179-204.

[20] T. Mitchell, *Machine Learning*, New York, McGraw Hill, 1997.

[21] T. Hirao, K. Takeuchi, H. Isozaki, Y. Sasaki, and E. Maeda, “NTT/NAIST’s Text Summarization Systems for TSC-2”, *Proceedings of the Third NTCIR Workshop,* 2003.

[22] T. Wiggerts, “Using clustering algorithms in legacy systems remodularization”, *Proceedings of the Fourth Working Conference on Reverse Engineering*, IEEE Computer Society, 1997, pp. 33-43.

[23] W. T. Chuang, and J. Yang, “Extracting Sentence Segments for Text Summarization Systems: A Machine Learning Approach”, *ACM SIGIR Special Interest Group on Information Retrieval*, 2000, pp. 152-159.

[24] Y. Matsuo, and M. Ishizuka, “Keyword Extraction from a Single Document using Word Co-occurrence Statistical information”, *American Association for Artificial Intelligence*, 2003.

Appendix

Tables A.1, A.2, A.3, A.4, and A.5 present the statistics for 90 research papers selected for testing our model

Table A.1

number of abstracts	number of keywords
0	1
12	2
19	3
22	4
16	5
10	6
5	7
4	8
2	9
abstracts=90	keywords=45
average number of keywords per abstract =3.46	

Table A.2

number of keywords	number of words
104	1
158	2
44	3
6	4
keywords=312	words = 10
average keyword length =1.79	

Table A.3

number of abstracts	number of sentences
0	1
0	2
3	3
10	5
13	11
10	9
12	8
13	6
15	7
13	6
abstracts=90	sentences= 55
average length of sentences =6.11	

Table A.4

number of abstracts	author's keywords not included in their own abstract
21	1
20	2
22	4
15	6
9	4
2	5
1	6
abstracts=90	Total =27
57.37% of the author's keywords do not appear in their own abstracts	

Table A.5

abstracts	keyword length
28	1
58	2
26	3
7	4
58	0
64.44% of abstracts do not include at least one of their own keywords	