International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Keyword Extraction using Clustering and Semantic Analysis

Dr. Mohamed H. Haggag[1], Dr.Amal Abutabl[2], Ahmed Basil[3]

[1, 2, 3]Computer Science Department, Faculty of Computers and Information, Helwan University

Abstract: ***Keywords are list of significant words or terms that best present the document context in brief and relate to the textual context. Extraction models are categorized into either statistical, linguistic, machine learning or a combination of these approaches. This paper introduces a model for extracting keywords by making words pairs and clustering these pairs based on the Semantic similarity that will be provided by using lesk algorithm and (WordNet), a lexical database for the English language. The model also used a statistical method to ensure clusters cohesion and provide more reliable result, because the final keywords will be selected from these clusters. This paper also show three other basic approaches to extract keywords, these approaches will be used to measure the efficient of the main approach. The proposed model showed enhanced over the three other approaches in both precision and recall.***

Keywords: keywords extraction, semantic analysis, wordnet

1. Introduction

Automatic keyword extraction is the task to identify a small set of words, key phrases, keywords, or key segments from a document that can describe the meaning of the document [1].

However, a large number of documents do not have keywords. At the same time, manual assignment of high quality keywords is costly and time-consuming, and error prone. Therefore, most algorithms and systems to help people perform automatic keywords extraction have been proposed.

Many text mining applications can take advantage of automatic keyword extraction, e.g. automatic indexing, automatic summarization, automatic classification, automatic clustering, automatic filtering, topic detection and tracking, information visualization, etc. Therefore, keywords extraction can be considered as the core technology of all automatic processing for documents.

There are two existing approaches to automatic keyword indexing: keyword extraction and keyword assignment. In keyword extraction, words occurred in the document are analyzed to identify apparently significant ones, on the basis of properties such as frequency and length. In keyword assignment, keywords are chosen from a controlled vocabulary of terms, and documents are classified according to their content into classes that correspond to elements of the vocabulary [2]. Existing methods about automatic keyword extraction can be divided into four categories, i.e. simple statistics, linguistics, machine learning and other approaches.

The simple statistics approach methods are simple and do not need the training data. The statistics information of the words can be used to identify the keywords in the document. Cohen uses N-Gram statistical information to automatic index the document [3]. N-Gram is language and domain-independent. Other statistics methods include word frequency [4], TF*IDF [5], word co-occurrences [6], and PAT-tree [7], etc.

The linguistics approaches use the linguistics feature of the words mainly, sentences and document. The linguistics approach includes the lexical analysis [8], syntactic analysis [1], discourse analysis [9] [10] and so on.

Keyword extraction also can be seen as supervised learning from the examples. Machine learning approach employs the extracted keywords from training documents to learn a model and applies the model to find keywords from new documents. This approach includes Naïve Bayes [11], SVM [12], Bagging [1], etc. Some keyword extraction tools, e.g. KEA [13], GenEx [14], have been developed.

Other approaches about keyword extraction my combines the methods mentioned above or use some heuristic knowledge in the task of keyword extraction, such as the position, length, layout feature of words, html tags around of the words, etc [15].

In this paper we produce a keyword extraction model that extracts keywords from single documents depending on the semantic similarity. The model removes the stop words and does stemming using porter stemming algorithm, then the document will be divided in to sentences. After that the model will take every word and find its best sense by comparing its senses that will be fetched from the WordNet with the other words senses to solve the word sense disembogued problem (WSD). Words pairs will be generated from the words of each sentence alone, so the sentence with (n) words will generate (n-1) words pairs for every word.

By using adaptive lesk algorithm the model will calculate the word-to-word similarity for all words pairs and find the best sense for the two words in each pair along with their similarity score. The pairs then will be clustered depends on its similarity score and also an average similarity score for each and every cluster will be calculated. Then a reverse similarity recursion is provided to validate the importance of words (one by one) to the cluster similarity score. The cluster average weighted similarity is re-calculated for n-1 words by dropping one word in each word score evaluation. This step will be repeated for every cluster to ensure clusters cohesion.

2. Semantic Analysis

Semantic similarity is the score of confidence which shows the two terms meanings relation semantically. High accuracy in semantic similarity evaluation is difficult to reach because the perfect semantic senses are only understood in a specific condition [16].

Semantic similarity depends on only one relation between two concepts which measures the similarity degree between terms in the hierarchy. Some pairs of words are closer in meaning than others, for example car–tire are strongly related while car–tree are not. WordNet [17] is used to measure the similarity; it supports the similarity between noun pairs (e.g. cat and dog) and verb pairs (e.g., run and walk).

Lesk Algorithm, the Micheal Lesk algorithm [18], is used to disambiguate words in the sentence context. The adaptive Lesk algorithm enhanced the original algorithm by finding overlaps in WordNet glosses and Synsets of words to measure semantic relatedness rather than glosses found in traditional dictionaries.

3. Proposed Model

There is three main steps the proposed system we will discuss to give a good understanding to our approach.

3.1 Preprocessing

In this level the system will reduce the number of the words in the document by removing all the words that's not important to make the processing more efficient.The preprocessing level has four steps:

3.1.1. Text Filtering

The system will load the article and divide it in to split lines delimiter by "." and convert all the words in to lower case letters. That will make the processing fast and easier to find the words similarity.

3.1.2 Stop-Word Remover

Preprocessing starts with the removal of stop words, which are considered non-information bearing words like (about, because, can, during, …). And this will do to reduce noisy data from the text.
The procedure done here is to create a filter for those words that remove them from the text; a list of 571 stop words has been used in this step.

3.1.3 Stemming

Removing suffixes by automatic means is an operation which is especially useful in the field of keyword extraction and information retrieval. The proposed system employs the porter stemming algorithm to improve system accuracy by reducing the large number of words morphological variants [19].

3.2 Words Sense Disembogued problem

Word-sense disambiguation (WSD) is an open problem of natural language processing, which governs the process of identifying which sense of a word (i.e. meaning) is used in a sentence, when the word has multiple meanings. To solve this problem the system will make pairs from the target word and all the words that appear in same sentence with it. After that the system will fetch all the words senses from the WordNet and looking if there is any similarity between the senses.

And the sense that in common between the target word and the other words will be the correct sense for the target word.

For example the following sentence:
("The bass line of the song is too weak").
The word “bass” can have the following meanings:

- *S: (n) bass (the lowest part of the musical range.)*
- *S: (n) sea bass, bass (the lean flesh of a saltwater fish of the family Serranidae)*
- *S: (n) freshwater bass, bass (any of various North American freshwater fish with lean flesh (especially of the genus Micropterus))*

So we will make pairs of the word ("bass") with the next words only if they are nouns, adjectives, adverbs and verbs because we already removed all the stop-words:
(Bass line - bass song - bass weak)

The software will look for example at the definitions of (song):

- *S: (n) song, vocal (a short musical composition with words) "a successful musical must have at least three good songs"*
- *S: (n) song (a distinctive or characteristic sound) "the song of bullets was in the air"; "the song of the wind"; "the wheels sang their song as the train rocketed ahead"*
- *S: (n) song, strain (the act of singing) "with a shout and a song they marched up to the gates"*
- *S: (n) birdcall, call, birdsong, song (the characteristic sound produced by a bird) "a bird will not learn its song unless it hears it at an early age"*
- *S: (n) song (a very small sum) "he bought it for a song"*
- *S: (n) Sung, Sung dynasty, Song, Song dynasty (the imperial dynasty of China from 960 to 1279; noted for art and literature and philosophy)*

And it will see that “musical” is found both in the meanings of the word “bass” and the word “song” so it will conclude that bass is “the lowest part of the musical range” and not “freshwater fish”.

3.3 Generating and clustering Keywords

Now we will generate the candidate keywords and clustering them and the final keywords will be selected from these clusters.

3.3.1. Generating initial Keyword list

All possible combinations of every sentence words (n) are topologically organized in pairs. Each word of sentence appears in (n-1) pairs along with other words in the same sentence, so the pairs will made for each sentence alone, with no intersection between words from different sentence

3.3.2. Clustering

Pair's similarity distance

The word clustering algorithms that we use is depends on the similarity between every words pairs senses.

By using adaptive lesk algorithm the model will calculate the similarity between the first word in (pair 1) with the first word in (pair 2) and also the similarity between the second word in (pair 1) with the second word in (pair 2). The summation of the two scores will be consider as a *pairs similarity distance* and saved in to a two dimensions array. This step will be repeated for every words pair with each and every word pair in the document.

In other words for every pair the *pair's similarity distance* will be calculated for
(Pair i, pair1), (pair i, pair2), (pair i, pair3),.. , (pair I, pair n-1)
So for every word pair we will calculate n-1 *pair's similarity distance*.

Clustering

The two pairs with the biggest *pair's similarity distance* score will be the first in the clustering process.

First we will check the *pair's similarity distance* of them with a minimum similarity threshold and if it's more than the threshold the two pairs will group together in one cluster, and the *pair's similarity distance* that will be set to the cluster will depend on the pair that has the highest *pair's similarity distance* in the cluster according to the pairs the not clustered yet.

The process will continue with all the other and it will stop only when there is no *pair's similarity distance* more than the threshold.

For example lets the threshold = 150 consider that (p1, p2, p3, p4) are pairs of words.

3.3.3. Word pairs similarity

By using adaptive lesk algorithm the model will calculate the word-to-word similarity for all words pairs. The best sense for the two words in each pair along with their similarity score will be the output of this step[20]. An example of the output will be in the following format:

profit#n#1 sale#n#5 78
profit#n#1 revenue#n#1 19
profit#n#1 quarter#n#2 58
profit#n#1 internet#n#1 25

3.3.4. Similarity Score Normalization, word-to-word

The similarity score of each word pair (between words wi & wj) is normalized according to the following formula:
Weighted similarity (wi, wj) = (Similarity (wi, wj) – min similarity)/ (max similarity-min similarity) … (1)

3.3.5. Average Similarity (word wi) & Average Similarity (clusters words), word-to-whole

The average weighted similarity score of each word (w*i*) is calculated using the weighted scores of equation (1) for all pairs (w*i*, w*1*), (w*i*, w*2*), …….. (w*i*, w*n*).

The cluster average weighted similarity score is then calculated for all the cluster words w*1*, w*2*, …. w*n*.
This step will be repeated for all the clusters.

3.3.6. Words Similarity versus Cohesion

At this step, reverse similarity recursion is provided to validate the importance of words (one by one) to the cluster similarity. The cluster average weighted similarity is re-calculated for n-1 words by dropping one word in each word (wi) score evaluation. In other words, weighted scores of equation (1) is calculated for the pairs (wi, w1), (wi, w2), …….. (wi, wn-1).

The effect of the dropped word is then evaluated by calculating the difference between the cluster similarity on the basis of n and n-1 words similarity, as the following:

Cluster *Similarity*= Cluster *Similarity (n-1 pairs) – Overall similarity (n pairs) (2)*

Check Cluster Similarity for the following options:

- Cluster *Similarity* < *0*, fix to the words list is required.
- *0* < Cluster *Similarity* <= *threshold (0.005*), the word is kept for further validation in the coming iterations.
- *0* < Cluster *Similarity*> *thresholds (0.005)*, the word are removed from the keywords list.

Negative difference in the first option means that the updated contribution, by removing on of the keywords list, will negatively affect the cohesion of the keywords list. Accordingly, this word need not to be dropped and another word should be selected.

Positive similarity difference is then required. However, in the second option, a small difference below threshold gives the impression that this word has no significant effect to the keywords list cohesion. The decision is to be postponed to later step after the words list will be changed. Positive similarity differences higher than the threshold ensures that the selected word is effectively increased the keywords cohesion. This step will be repeated for every cluster to ensure the keywords cohesion in all the clusters that we have.

3.3.7. Selecting Final Keywords

Every cluster will refer to specific senses, so the size of the cluster will reflects the important of the senses that belong to that cluster. The final keywords number that will be selecting from every cluster depends on the size of that cluster. So the cluster with highest number of pairs is more likely to be the cluster with highest number of keywords. And so on with the other clusters. So from every cluster, depending on the cluster size we will select a number of pairs with highest frequency as final Keywords.

4. Evaluation and Discussion

The system will be evaluated by comparing the proposed system result with the results of three other systems that use three different methods to extract keywords.

These methods are:

Term Frequency method; keywords are defined according to their term frequency. Most frequent terms are considered highly expressing the text.

Position Weight method; keywords are extracted according to their positions within the sentence and paragraph [21].

Semantic Term Frequency method; the best sense of each word, resulted from TF method, is evaluated using Adaptive Lesk algorithm to the first occurrence of keyword in the document. A context window of size value 3 is used. Getting the best senses from all documents, the vector space model is constructed that includes semantic keywords.

Experimental results are conducted and analyzed using articles with predefined keywords that extracted manually. These articles are selected from wikipedia website randomly in different fields.

Evaluation metrics considered are the Precision and Recall, which are the standard metrics for retrieval effectiveness in information retrieval. Basically calculated as follows:

Precision = TP / (TP + FP)
Recall = TP / (TP + FN)
Where:
TP= keywords extracted keywords by the algorithm and already found in document's predefined keywords.
FP= keywords extracted keywords by the algorithm and doesn't found in document's predefined keywords.
FN= document's predefined keywords that are not extracted by the algorithm.

The experimental result of the proposed system show enhance in both recall and precision over the result of the other three methods

Table 1: Precision and Recall results

	Recall	*Precision*
Term Frequency	77.2	68.5
Position Weight	80.3	73.0
Semantic Term Frequency	83.5	76.3
Proposed system	86.4	78.5

Table 1 show the precision and recall results for term keywords extraction based on term frequency method, position weight method and the proposed model. Proposed model results in enhanced precision and recall evaluation over other methods.

5. Conclusion

Keywords Extraction is widely used in text refinement. It is important in text processes such as clustering, classification or information retrieval. Keywords should express the core content of the document. This paper proposed a keywords extraction system that work based on sense similarity and also use the clustering and statistical method to extract the best keywords for the document, Results ensure enhanced precision and recall compared to traditional statistical approaches.

References

[1] Hulth. Improved Automatic Keyword Extraction Given More Linguistic Knowledge. In: Proceedings of the 2003 Conference on Emprical Methods in Natural Language Processing, Sapporo, Japan, 2003: 216-223.
[2] [2] O. Medelyan, I. H Witten. Thesaurus Based Automatic Keyphrase Indexing. In: Proceedings of the Joint Conference on Digital Libraries 2006, Chapel Hill, NC, USA, 2006: 296-297.
[3] J. D. Cohen. Highlights: Language and Domain-independent Automatic Indexing Terms for Abstracting. Journal of the American Society for Information Science, 1995, 46(3): 162-174.
[4] H. P. Luhn. A Statistical Approach to Mechanized Encoding and Searching of Literary Information. IBM Journal of Research and Development, 1957, 1(4): 309-317.
[5] G. Salton, C. S. Yang, C. T. Yu. A Theory of Term Importance in Automatic Text Analysis, Journal of the American society for Information Science, 1975, 26(1): 33-44.
[6] Y. Matsuo, M. Ishizuka. Keyword Extraction from a Single Document Using Word Co-ocuurrence Statistical Information. International Journal on Artificial Intelligence Tools, 2004, 13(1): 157-169.
[7] L F. Chien. PAT-tree-based Keyword Extraction for Chinese Information Retrieval. In: Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR1997), Philadelphia, PA, USA, 1997: 50-59.
[8] G. Ercan, I. Cicekli. Using Lexical Chains for Keyword Extraction. Information Processing and Management, 2007, 43(6): 1705-1714.
[9] S. F. Dennis. The Design and Testing of a Fully Automatic Indexing-searching System for Documents Consisting of Expository Text. In: G. Schecter eds. Information Retrieval: a Critical Review, Washington D. C.: Thompson Book Company, 1967: 67-94.
[10] G. Salton, C. Buckley. Automatic Text Structuring and Retrieval –Experiments in Automatic Encyclopaedia Searching. In: Proceedings of the Fourteenth SIGIR Conference, New York: ACM, 1991: 21-30.
[11] E. Frank, G. W. Paynter, I. H. Witten. Domain-Specific Keyphrase Extraction. In: Proceedings of the 16th International Joint Conference on Aritifcal Intelligence, Stockholm, Sweden, Morgan Kaufmann, 1999: 668-673.
[12] K. Zhang, H. Xu, J. Tang, J. Z. Li. Keyword Extraction Using Support Vector Machine. In: Proceedings of the Seventh International Conference on Web-Age Information Management (WAIM2006), Hong Kong, China, 2006: 85-96.
[13] H. Witten, G. W. Paynte, E. Frank, C. Gutwin, C. G. Nevill-Manning. KEA: Practical Automatic Keyphrase Extraction. In: Proceedings of the 4th ACM Conference on Digital Library (DL'99) , Berkeley, CA, USA, 1999: 254-26.
[14] P. D. Turney. Learning to Extract Keyphrases from Text. NRC Technical Report ERB-1057, National Research Council, Canada. 1999: 1-43.

[15] J. B. Keith Humphreys. Phraserate: An Html Keyphrase Extractor. Technical Report, University of California, Riverside, 2002: 1-16.
[16] Junsheng Zhang, Yunchuan Sun, Huilin Wang, and Yanqing He. “Calculating Statistical Similarity between Sentences”. Journal of Convergence Information Technology, Volume 6, Number 2. February 2011
[17] C. Fellbaum (Ed.), WordNet: An electronic lexical database, MIT Press, 1998.
[18] M. Lesk, - Automatic sense disambiguation using machine readable dictionaries: how to tell a pine code from an ice cream cone - in: Proceedings of the 5th annual international conference on Systems documentation, ACM Press, 1986.
[19] Lovins, Julie Beth (1968). "Development of a Stemming Algorithm".Mechanical Translation and Computational Linguistics 11: 22–31.
[20] Mohamed H. Haggag. "Keyword Extraction using Semantic Analysis". International Journal of Computer Applications (0975 – 8887) Volume 61– No.1, January 2013.
[21] Xinghua Hu; Bin Wu - Automatic Keyword Extraction Using Linguistic Features,- Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference, Page(s):19 - 23 Dec. (2006)