

Machine Learning Approach for Word Extraction

Ajith K J¹, Ajay Kumar C M¹, Adithya S Prakash¹, Ramesh G²

¹Dept. of Computer Science and Engineering, The National institute of engineering, Mysuru, Karnataka, India

²Assistant professor, Dept. of Computer Science and Engineering, The National institute of engineering, Mysuru, Karnataka, India

Abstract - Keywords and Acute words in a document are used to describe structure within information retrieval system as they are easy to remember, store, define and share. Acute words can be applied across multiple corpora and are independent of any corpus when compared to mathematical signatures. The goal of word extraction is to train the data using the approach of machine learning and extract the words which gives more meaning in support for understanding the document and for summarized text. While in word extraction we rank the words which is free from special characters, numbers and retrieve the words which contains maximum information and give weightage to the document in better understanding.

Key Words: Extraction, co-occurrence, word frequency, classification, training, pre-processing.

1. INTRODUCTION

Paramount words or keywords plays a vital role in retrieving the right information as per user requirement. In today's world of technology information we study lots of journal, newspaper, books, articles, messages which makes the user difficult to go through all the words and sometimes leaves them to be apart from studying those materials, instead a tool of text summarization is required and need for extraction summarized text, keywords, acute words which actually provides the actual contents of the document. As such effective words are necessity. Here the word extraction is the smallest unit which extracts the meaning of entire document, many application can take advantage of it such as automatic indexing, classification, clustering, filtering, web searches etc.

In this paper, we put summarized text in the machine learning setting and introduce a supervised learning approach for word extraction. We break the document into text blocks which is structure-independent and extract the features from the text blocks. We also do semantic analysis based on id and class attributes associated with text blocks using Naïve bayes Algorithm. To improve accuracy in understanding of the document from the extracted words the result of semantic analysis can be incorporated.

2. RELATED REASEARCH

Word extraction have been applied to improve the functionality of information retrieval systems. Jones and paynter describe Phrasier, a system that list document related to the primary documents keyword. Hidayet Takci and Tunga Gungor made to classify the language independent text document using centroid-based classification approach. This experiment resulted to obtain better accuracy than other methods when done on multilingual corpus. Based on semantic hierarchy Xiaogang peng and beng Choi propped automatic classification of documents which actually classifies the document into group of texts which was actually based on the word extractions keeping semantic elements. A Machine learning approach to webpage content extraction by jiawei Yao and Xinhui Zuo where they explained that the webpage contains boilerplate elements such as to be comments, advertisements etc. These are treated as noise in the documents and are removed properly to improve the user's experience in understanding that document. Using the SVM classifier they set relevant features to predict weather the text block is content or non-content. In sentimental analysis which have been focused on classification models for text the approach given in hand-coded rules by neviarouskaya in 2010 have done the classification only for few categories and hence our work attempts to extend this by inferring the specific reactions rather than broad categories.

3. IMPLEMENTATION

There are various implementation method to achieve word extraction by locating and defining the acute words that have been used in the document. Despite the difference, in most of the cases we rank the text blocks in the given document, separate them by frequencies, predict their nature and define the a set of words that accurately describe the information contained in the text and provide maximum satisfaction for the user.

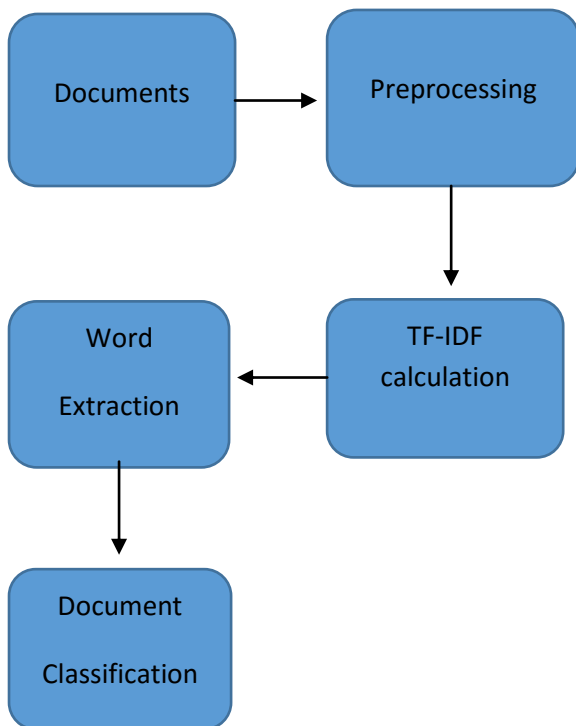


FIG - 1: Word extraction based classification

3.1 Text Classification

In any document, for the word extraction the text classification is one of the main application in machine learning. Here the task is to assign the unlabeled new text document to predefined category. In classification we face two main problems, firstly in the training phase extraction of the feature terms that actually become the actual acute words and secondly actual classification of the document using these feature term in the test phase. In classification process of preprocessing is done in which the words are stemmed and the stop words are removed. Fig-1 depicts the idea of how actually the process of extraction is done based on classification. The next process of term frequency calculation is carried out and TF-IDF is calculated.

3.2 Word Frequency analysis

Most of the work concerned the frequency of term usage in the text, focused on defining words in relation to a document. This technique is called as term frequency- inverse document frequency (tf-idf). This technique is a numerical statistic which reveals that a

word is how important to a document in a collection. It is actually a weighting factor in information retrieval. The word is extracted based on the weighting the term positivity and negativity for the number of times the term occurs within the specific document. Let us consider the term t and document d belongs to D , where t appears in n of N documents in D . the function is of the form

$$TFIDF(t,d,n,N) = TF(t,d) \times IDF(n,N)$$

There are many functions related to TF and IDF functions. Any functions of TF and IDF can be used and such TF and IDF functions include:

$$TF(t,d) = \begin{cases} 1, & t \in d \\ 0, & \text{else} \end{cases}$$

$$TF(t,d) = \sum_{word \in d}^0 1 \text{ if } word = t$$

$$IDF(n,N) = \log(N/n)$$

Finally the TF-IDF function by combing will be:

$$TFIDF(t,d,n,N) = (\sum_{word \in d}^0 1 \text{ if } word = t) \times \log(N-n/n)$$

Thus by this function all the words in the document that are extracted can be ranked in the given document. The higher TF-IDF score indicates that a word is both important to the document and uncommon in the document. TF-IDF provides a good method for heuristic for determining the frequent words. Till today this method is well and good because of its effectiveness and simplicity.

3.3 Text Rank

Now another important algorithm which plays a important role in our research is Text rank which actually helps to depict a Graph-based ranking algorithm. This type of graphs actually the essential way of deciding the importance of those text blocks. "Ranking" and "Recommendation" are the methods to implement in the graph-based algorithm. Higher the number of votes that are cast for particular set of data, higher will be the importance for that set of data. Here

the importance of the casting the vote determines how important the vote itself is.

Text rank in word extraction is set of words that are representative for a given natural language text. The units that are going to be ranked are the sequence of the units that are extracted from the documents and these represent the vertices that are added to the text graph. We here use co – occurrence relation which are controlled by the distance between word occurrences. After the occurrence in post-processing all lexical units selected as vital words or keywords by text rank algorithm are marked and set of adjacent key-words are joined into a multi-word keyword. Based on the text rank algorithm for a sample text consist of more than 100 words the important words that are vital in that sample text is given in the FIG-2.

```

C:\
C:\Users\User 1\Desktop>python k.py yes.txt
extraction monolingual ripper statistical terms
C:\Users\User 1\Desktop>
    
```

FIG-2: Word extraction for sample text.

3.4 Naive Bayes

The text classification is our next task where we use naive bayes classifier by observing bayes rule,

$$P(c|d) = P(c)P(d|c) / p(d)$$

where $p(d)$ plays no role in selecting. To estimate the term $P(d|c)$, Navie bayes decomposes it by assuming the f_i 's are independent d 's class:

$$P_{NB}(c|d) = P(c) \left(\prod_{i=1}^m P(f_i|c)^{n(d)} \right) / P(d)$$

Using add-one smoothing the training method will actually consists of relative-frequency estimation of $P(c)$. In additional with bayes class to identify the classification of text we encounter the method called Support Vector Machine (SVM) as a final classifier to make predictions for both task. Navie bayes tend to classify the text blocks of the documents as the most possible category. In this one possible feature model is

to select the topic from document and find the vote distribution for that particular topic and hence the probability of voter selecting category can be estimated based on the topic occurrence. To estimate the presence of any topics, whether it be the similar of a topic in each document, or the proportions of various topics within each document. We run Latent Dirihlet Allocation (DLA) method on our documents after distribution to get the outline of the document and new set of text blocks. First we ran this results varying the number of topics that are assumed to be in our set of data. We use heuristic in next process in each scenario. Our resulting test accuracies are shown in the fig-3.

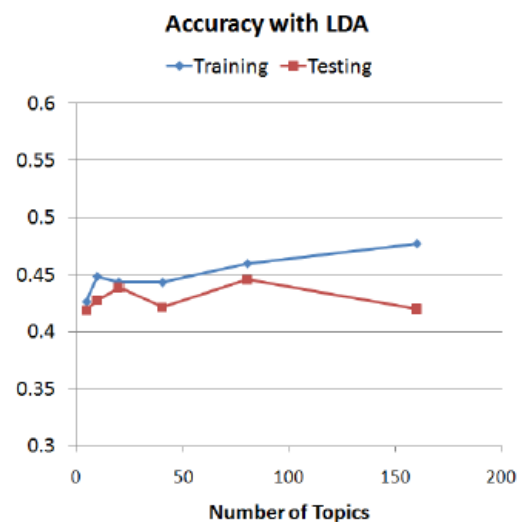
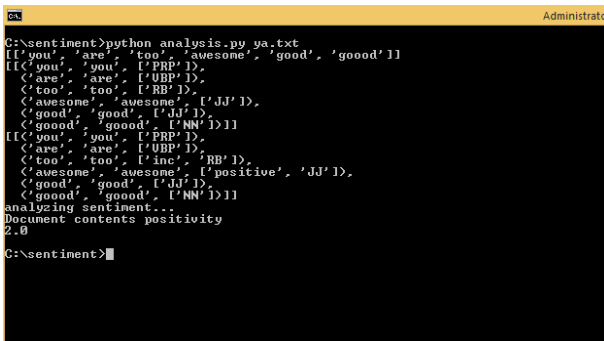


FIG- 3: Accuracy using LDA-derived features

We divide the category into set of negative, neutral and positive entities where the number of possible category will be calculated and based on the percentage the document is classified and results are computed as it is positive or negative semantics. The output of the sample document is shown in the FIG-4 which depicts the positivity of that sample document which is given as an input.



```
C:\sentiment>python analysis.py ya.txt
[['you', 'are', 'too', 'awesome', 'good', 'good']]
[[['you', 'you', ['PRP']],
  ['are', 'are', ['VBP']],
  ['too', 'too', ['RB']],
  ['awesome', 'awesome', ['JJ']],
  ['good', 'good', ['NN']]
  [['you', 'you', ['PRP']],
  ['are', 'are', ['VBP']],
  ['too', 'too', ['RB']],
  ['awesome', 'awesome', ['positive', 'JJ']],
  ['good', 'good', ['JJ']],
  ['good', 'good', ['NN']]
  analyzing sentiment...
  Document contents positivity
  2.0
C:\sentiment>
```

FIG-3: sentiment analysis for sample text

4. CONCLUSION

The most commonly used keyword extraction algorithm when a document is given will be TF-IDF. Along with this the Text rank algorithm will be supportive and helpful to depict the graph of actually what and how the extraction is going on. As early mentioned that the methods in extraction the words are the supportive tool for the text summarization and will be very helpful in better understanding of the document. Here nearly all algorithms for word extraction depends on a weighed function which actually balances some measures of term within the document.

Overall, from many examples with these collection of approaches gave us the overall picture of the challenges of the analysis on the project and was helpful in getting the sentiment of the document. First we note the use of classification and by implementing the algorithms we faced many challenges in extracting keyword such as eliminating the foreign keys, ranking the text blocks etc.

REFERENCES

- [1] Kamal Nigam and Andrew, "Text classification from labeled and unlabeled document". Machine learning vol.4 Kluwer academic publication
- [2] "A Novel concept for extraction based text summarization", by Dipti Pawar, S.H.Patil. Vol. 5(3), 2014, 3301-3304.
- [3] Harry Zhang, "The optimality of naive bayes", University of New Brunswick, Canada
- [4] Mitsuru Ishizuka, "Keyword extraction from a single document using word co-occurrence statistical information".
- [5] "Learning to rank from structures in Hierarchical Text classification" by Richard Johansson and Alessandro Moschitti, University of Gothenburg, Sweden.