Keyword Extraction Using Naive Bayes

Yasin Uzun

Bilkent University, Department of Computer Science, Turkey, yasinu@cs.bilkent.edu.tr

Abstract

As the internet grows, amount of electronic text increases rapidly. This brings the advantage of reaching the information sources in a cheap and quick way. Keywords are useful tools as they give the shortest summary of the document. But they are rarely included in the texts. There are proposed methods for automated keyword extraction. This paper also introduces such a method, which identifies the keywords with their frequencies and positions in the training set. It uses Naïve Bayesian Classifier with supervised learning.

INTRODUCTION

Keywords are set of significant words in a document that give high-level description of the content for investigating readers and are useful tools for many purposes. They are used in academic articles to give an insight about the article to be presented. In a magazine, they give clue about the main idea about the article so that the readers can determine whether the article is in their area of interest. In a textbook they are useful for the readers to identify the main points in their mind about a particular section. They can also be used for search engines in order to return more precise results in shorter time. Since keywords describe the main points of a text, they can be used as a measure of similarity for text categorization. In summary, keywords are useful tools for scanning large amount of documents in short time.

Despite the usefulness of the keywords, very few of the current documents include them. In fact many authors are not intended to extract keywords and do not denote them unless they are not explicitly instructed to do so [1]. Extracting keywords manually is an extremely difficult and time consuming process, therefore it is almost impossible to extract keywords manually even for the articles published in a single conference. Therefore there is a need for automated process that extracts keywords from documents. Microsoft Word implements such a function for its users. Verity also uses a keyword extractor for Search97. Tetranet also uses one [1].

There are two approaches for the described problem.[2] In *keyword assignment* approach, which is also known as text categorization, there is a prior set of vocabulary and the aim is to match them to the texts in a set. In the other approach, namely *keyword extraction,* the aim is to extract keywords with respect to their relevance in the text without prior vocabulary. Kea [2] and GenEx [1] are two proposed solutions for the second problem stated. While GenEx uses a genetic algorithm as a solution, Kea uses Naïve Bayesian Decision rule with two features, which are *TFxIDF* the distance of the word to the beginning of the text.

In this paper we will investigate the problem from the keyword extraction approach, but the solution can also be applied to the other with little modification. We use Bayes method, which is one of the machine learning methods, to determine the distinguishing features of keywords in a text and extract keywords from text using this information. We follow supervised learning assuming that there is a set of training documents with extracted keywords to teach the learner the identifying features of the keywords.

KEYWORD EXTRACTION USING NAIVE BAYES

Machine Learning techniques consider the keyword extraction as a classification problem. There are words (examples) in a document and the purpose is to identify whether a word belong to the class of *keywords* or *ordinary words*. As with other machine learning methods, we assume that there is a training set that can be used to learn how to identify keywords and using the knowledge gained from the training set, the unlabeled examples, which are the new documents in our case.

Bayesian Decision Theory is a fundamental statistical approach based on the tradeoffs between the classification decisions using probability and the costs that accompany those decisions. It assumes that the problem is given in probabilistic terms and the necessary values are already given [3] Then it decides on the best class that gives the minimum error with the given example. In cases where there is no distinction is made in terms of cost between classes for classification errors, it chooses the class that is the most likely (with the highest probability).

Preprocessing the Documents

It is clear that all the words do not have the same prior chance to be a keyword in a document. Firstly, the all tokens in the document should be identified by using delimiters such as spaces, tabs, new lines, dots, etc. Even though propositions may appear many times in a text, they should not be labeled as a keyword. The second term in the TFxIDF score seems a barrier for such words, but may not be

adequate. Therefore it can be a solution to eliminate by assigning prior probability of zero. Moreover non-alphanumeric characters and numbers should be eliminated.

Constructing the Model

In order to decide whether to label a word as a key, the words in the document must be distinguished by using features and the properties of keywords have to be identified. The first possible feature that comes into mind is the frequency, which is the number of times a keyword appears in the text. It is obvious that the more important phrases will be more used in a text. We can expect the term "communication" or "battery" in a research paper about sensor networks much more than the others. But this may cause some problems. Usually prepositions such as "the, that, this, etc." appear much more than any other words (even those which are keys) even though they have no value as a keyword. Therefore, we have to extend the concept of a keyword. It can be said that other than the frequency, what makes a word a key is the specialty of the word to the document that the word exists in. If a word appears much more frequently in a document than with respect to other documents, this can be another distinguishing feature of that word on deciding whether it is a keyword or not. Combining these two properties, we obtain the metric *TFxIDF* (standing for Term Frequency x Inverse Document Frequency) score, which is the standard metric used in Information extraction [2], and for a word W in document D, is defined as

TFxIDF (P, D) = P(word in D is W) x [- log P(W in a document)].

The first term in this formula is calculated by counting the number of times the word occurs in the document and dividing it to the total number of words in it. The second term is calculated by counting the number of documents in the training set that the word occurs in except D and dividing it by the total number of documents in the training set. A problem with using this metric is that if the training set consists of documents with similar subjects, the second term will approximate to zero causing scores of the actual keywords approach to zero if they appear in most of them. Therefore the training set must be selected by choosing texts with varying subjects.

Another possible feature can be the position of the word in the document. There are several choices for characterizing the position. Obviously the first property is the distance of the word to the beginning of the text. Usually the keywords are concentrated in the beginning and end of the text with some more spread on the body. Whatever the distribution is, if we can learn it from the training set, it can be useful to predict the keywords in the text.

The position of the word according to the paragraph that it exists in can be another identifying feature of the keywords since we also expect them to be in the beginning and end of the paragraph. Furthermore, the position of the word in the sentence may be an identifying feature. For instance, while more important terms are found in the beginning or end of the sentence in English, they are placed before the last word in Turkish. Therefore, if we apply learning method for extracting keywords, we can use it independent of the structure or type of the language.

Naive Bayes makes the assumption that the feature values are independent. With this assumption, we can compute the probability that a word is a key given its *TFxIDF* score(T), the distance to the beginning of the paragraph (D), the relative position of the word with respect to the whole text (PT)and the sentence that it exists in (PS) by using Bayes Theorem [5]:

$$P(key \mid T, D, PT, PS) = \frac{P(T \mid key) \times P(D \mid key) \times P(PT \mid key) \times P(PS \mid key)}{P(T, D, PT, PS)}$$

where P(key) denotes the prior probability that a word is a key (assumed to be equal for all words in our problem), P(T | key) denotes the probability of having *TFxIDF* score T given the word is a key, P(D | key) denotes the probability of having neighbor distance D to the previous occurrence of the same word given the word is a key, P(PT | key) denotes the probability of having relative distance PT to the previous occurrence of the same word given the word is a key, P(PS | key) denotes the probability of having relative distance D to the beginning of the paragraph given the word is a key and P(T, D, PT, PS) denotes the probability that a word having *TFxIDF* score T, neighbour distance D, position in the text PT and position in the sentence PS.

The Classification

Having the training set associated with feature values mentioned earlier, we can use the naïve Bayesian classifier to extract keywords. Although the independency assumption about the features we used can be criticized, it has been shown that Bayesian classification performs very well even if the independency assumption fails.[6] Therefore we can assume that the features are independent and use naïve Bayesian classifier to extract keywords from the documents.

By using the formula above and the given training set, we can build the classification model and then processes the input documents to extract the keywords and sorts the possible keywords according to

their ranks (probabilities). Tie condition may be a problem here. Two words can have the same rank. In this case, *TFxIDF* score can be used as a tie-breaker. If tie condition continues (which is a rare case), the distance and the position of the word in the text can be used.

EXPERIMENTAL RESULTS

We are still working on the implementation to obtain experimental result to evaluate the performance of the method mentioned. We hope they will be ready to present for the final copy.

CONCLUSION

Keyword Extraction is necessary for many purposes. They are used in many areas varying from search engines to text categorization. There are proposed methods for automatic keyword extraction from documents. Our method uses naïve Bayesian classifier, which uses the features *TFxIDF* score, distance of the word to the beginning of the text, paragraph and the sentence to identify keywords in the text. We assume the keyword features are normally distributed and independent. Our method follows supervised learning by using documents in the training set and extract keywords from the test set with the gained knowledge.

REFERENCES

[1] P. D. Turney, Learning Algorithms for Keyphrase Extraction, *Information Retrieval*, 1999

[2] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning. *Domain-specific keyphrase extraction*. In IJCAI, pages 668--673, 1999.

[3] R. O. Duda, P. E. Hart, D. G. Stork, *Pattern Classification*, page 20, Wiley-Interscience, 2000.

[4] G. Salton and M. J McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[5] Thomas Bayes. An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society (London).* 53:370-418, 1763

[6] P. Domingos and M. Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. Machine Learning, 29(2/3):103-130, 1997.