KEY-PHRASE EXTRACTION FOR CLASSIFICATION

Nikitas N. Karanikolas* and Christos Skourlas**

*Areteion University Hospital, Systems' Head, Athens, Greece
** TEI of Athens/Dept. of Informatics, Professor, Aegaleo, Greece

nnk@aretaieio.uoa.gr

Abstract: In this paper we consider the problem of extracting key-phrases from a bilingual texts collection and using them for text classification. A key-phrase could be defined as a sequence of words of a given size in a given partial order that occur within a sentence. We describe an algorithm for the discovery of key-phrases. Then, a framework of handling multilingual texts / documents is described which combines the use of the traditional vector space model with a new similarity measure which is based on the key-phrases. This framework is used for finding the most similar documents of a training set with any new document and selecting the classes of the similar documents as the most plausible ones for the new document. Some experimental results are also presented.

Introduction

Phrase extraction is the subject of interesting research accompanied by various experimental and operational tools. It is worth mentioning here that these tools are usually oriented towards the extraction of information from Web applications. As an example we can mention the case of the KEA system [1] which implements in Java a rather simple algorithm for extracting phrases from English text.

In the case of the Greek language there is a rich syntactic and inflectional (grammatical) system that implies further difficulties in the extraction process. Hence, the use of a stop-words list, some morphological analysis, stemming, etc are prerequisites for handling Greek-Latin text. It is also interesting to see the problem of extracting phrases from texts written in other languages with rich inflectional system [2].

We must also stress the importance of using these extracted phrases as terms characterizing the document and their store and organization as a basis for effective free-text searching.

Key-phrase extraction

Most machine learning and text mining techniques are adapted towards the analysis of text collections. Texts are composed of words or phrases and have an inherent sequential structure. Such a text can be viewed as a sequence of words, stop-words, punctuation marks, parentheses, key-phrases, where each key-phrase has an associated frequency e.g. the number of occurrences. Let see a bilingual portion of a discharge letter covering past history and presentation:
"71 years old male patient with a history of a AAA (Abdominal Aortic Aneurysm) repair 13 months ago presented with RUQ (Right Upper Quandrum) pain and a palpable mass of two months duration".

Stop-words could be words as the following ones: with, a, of, and that have no implication in retrieving texts. Key-phrases could be sequences of words as the following ones:

AAA Abdominal Aortic Aneurysm
RUQ Right Upper Quandrum pain

The basic problem in analyzing such a text collection is to find key-phrases, i.e., sequences of words occurring frequently close to each other. For example, a phrase "A (followed by) B (followed by) C (followed by) D", where A, B, C, D are four distinct words, must occurs a specific number of times in order to be considered as a key-phrase. Note that in the sentence there can be other words occurring between these ones. The user must define how close is close enough by giving the width of the word window within which the key-phrase must occur. The user also must specify how often a key-phrase has to occur to be considered frequent.

Discovery of frequent key-phrases

Mannila et. al. [6] describe an algorithm for discovering frequent episodes in a telecommunication network alarm database. The essential points of their method are the following:

a. Input data is a flat sequence of ordered episodes (faults) and is not organized into other structural levels. This fact implies that the situation in our case is different. We have documents / texts collections where words and phrases compose sentences, sentences compose documents and documents compose the collection.
b. Their main idea is that for every frequent sequence of events all the subsequences are at least equal frequent. Therefore, the construction of candidate sequences of n+1 width (C_{n+1}) can be based on the frequent sequences of n width (L_n). The implication is that the search space could be reduced.

c. The episodes of a sequence are successive but in the sequence can be other events occurring between the episodes.

We think that the choice of *sequences of events* which is based on high frequencies is a reliable method for forecasting in general. An adaptation of such a method in a text processing environment can be helpful and especially in the creation of a *type ahead wizard.*

However, if *key-phrases (sequences of words)* are used as indexing terms for information retrieval it is better to choose phrases that exist in a few texts (if the candidate phrases exist in many texts then they are useless for retrieval purposes) and are quite frequent within these texts. Therefore, it is better to use appropriate measures that prefer / choose such phrases.

A simple measure in this category is the following:

$$freq(P,D) \times \frac{N}{docfreq(P)}$$

where freq(P,D) is the frequency of phrase P in document D, docfreq(P) is the number of documents in the collection that include phrase P and N is the number of documents in the collection.

An alternation of the above measure is used in the KEA system [3] to build a prediction model based on a training set of documents. The following equation describes this measure:

$$TFxIDF = \frac{freq(P,D)}{size(D)} \times -\log_2 \frac{docfreq(P)}{N}$$

If key-phrases are used as features for texts/documents classification [4, 5] then the frequent key-phrases are inappopriate for such a task.

If the choice of key-phrases for text classification is based on measures, as the one used by the KEA system, then there are some potential problems:

1. A candidate key-phrase that exists in many documents of only one class (and not in another class) could be erroneously rejected if the number of the documents of this class is greater than the number of documents of other classes.
2. A candidate key-phrase could be erroneously chosen in case that it exists in a small subset of texts of a numerous / dense class and all these texts are dedicated on a specific subtopic of the topic of class.
3. A candidate key-phrase is chosen because it exists only in few texts in the collection and it is quite frequent within these texts but these few texts that contain the candidate key-phrase are spread within a lot of classes.

As a first conclusion, the choice of the key-phrases must not be based on frequent (within the whole text collection) candidate phrases or measures like the one used in KEA system.

Instead of using these frequent, in the whole collection, phrases for text classification we can use key-phrases which are frequent within the documents of one or few classes but do not be frequent in the documents of the rest classes in the training set. We also estimate that the selection of key-phrases based on some syntactic structures poses some extra complexity (use of morphological part of speech taggers and syntactic analyzers) and works restrictively in the selection of key-phrases for classification.

The above discussion combined with the analysis of the method in [6] have influnced us in the construction of a new algorithm for key-phrase extraction for classification.

We formalize the problem of phrase extraction for classification in the following way. Given a collection of documents subdivided into classes, a window width and a frequency threshold, find all key-phrases that occur frequently enough in one or few classes but do not occur frequently enough in other classes. We describe an algorithm for solving this task. The algorithm has two alternating phases: building new candidate key-phrases, and evaluating how often these ones occur in a class of the collection.

The idea of building candidate patterns from smaller ones is incorporated to the algorithm. Such an idea has been profitably used in the discovery of association rules etc. and occurs also in other contexts [6, 7].

Adapting the main ideas discussed in [6] we can claim that the efficiency of our algorithm is based on the fact: *Potentially, a very large number of candidate key-phrases has to be checked. Hence, we can reduce the search space by building larger key-phrases from smaller ones. In other words, it is only necessary to test the occurrences of key-phrases whose all sub key-phrases are frequent.*

Algorithm

We give an algorithm for finding key-phrases that occur frequently enough in one or few classes but do not occur frequently enough in many classes:

```
For every class of the training set do
   For every document of the class do
      Stemming
      stopword removal
   End {For every document of the class}
   Choose the most frequent stems of the
   class (P0 - parameter)
   Form the candidate double word phrases
   (C2) from the frequent stems (L1)
   Choose the most frequent double word
   phrases (L2) (W1 and P1 - parameters)
   For x=3,4 do
      Form the candidate x - width word
      phrases (Cx) from the frequent (x-1) -
      width word phrases (Lx-1)
      Choose the most frequent x - width
      word phrases (Lx) (P2 and W2,
      P3 and W3 - parameters)
   End {For x=3,4 do}
   Compose an integrated list by joining Lx
   (for x=2,3,4). This join, forms the
   frequent word phrases of class (LFC)
End {For every class of the training set}
Integrate / Join the lists of frequent word
phrases of all classes of the training set
Reject the frequent word phrases that exist
in many classes (Pt - parameter). The rest
of the frequent word phrases form the
set of key-phrases or «Authority list»
Form the dictionary of «Terms». It is the
list of stems that are components of the
key-phrases of the «Authority list».
```

Parameters:

P0 percentage of texts of the class that must contain a stem,
W1 width of window that covers 2-word phrases,
P1 percentage of texts of the class that must contain a 2-words phrase,
W2 width of window that covers 3-word phrases,
P2 percentage of texts of the class that must contain a 3-words phrase,
W3 width of window that covers 4-word phrases,
P3 percentage of texts of the class that must contain a 4-words phrase,
Pt percentage of classes that can contain a key-phrase.

Adopting the idea proposed in Mannila et. al. our algorithm works iteratively, alternating between building (steps 7 and 10) and recognition phases (steps 8 and 11). In the building phase of an iteration *i,* a collection C_i of new *candidate key-phrases* of distinct words is built, using the information available from smaller frequent key-phrases. Then, these candidate key-phrases are recognized in the documents in the class and their frequencies are computed. The collection L_i consists of frequent key-phrases in C_i. In the next iteration *i+1*, candidate key-phrases in C_{i+1} are built using the information about the frequent key-phrases in L_i. The algorithm starts by constructing C_1 to contain all key-phrases consisting of single words. At the end of each step, the list of frequent key-phrases of the processed class is being built (step 13). At the end, the algorithm composes the Authority list (steps 15 and 16).

Steps 7 and 10 are based on the second (b) essential points of Mannila's paper [6]. The set of candidates 2-word phrases (C_2) must contain key-phrases of length 2 (key-phrases including two stems of distinct words). To construct this set, we form the cartesian product and then remove all the tuples that have the same elements. Figure 2 illustrates an example of the application of step 10 of the algorithm. In this case the construction of the sixth class of key-phrases C_6 is based on the set L_5.

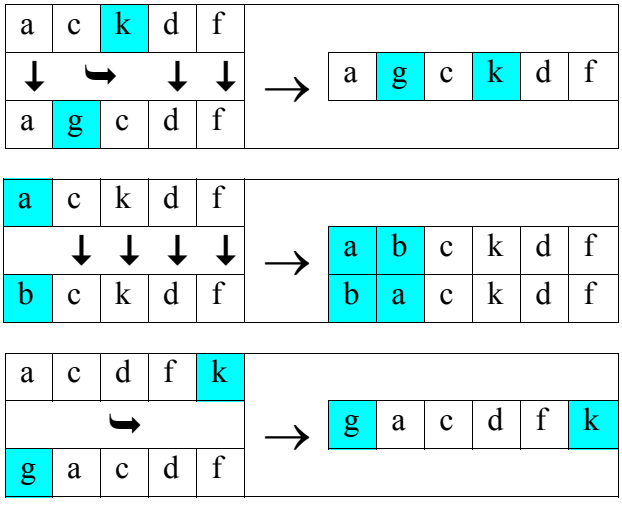

Figure 1: construction of C_6 based on L_5.

Similarity based classification

Karanikolas and Skourlas [5] presented the idea that the classification of new medical documents can be based to their similarity to existing documents (of a "training" set). Such an Instance based learning method assumes that similar documents must be classified in the same category or (in other words) must share the same classification code (e.g. the same ICD code). According to this approach the text collection is divided into a number of classes and each document of each class is characterized by a number of key-phrases. For each document in the collection the existing key-phrases in the document can have a frequency, etc.

The vector space model

In the popular vector space model a data set of n unique terms is specified, called the index terms of the document collection, and every text can be represented by a vector of weights of the terms in the document. In our case we use a set of m key-phrases instead of simple terms and the vector representation of each document can be:

$(kp_1, kp_2, \ldots, kp_m)$

where kp_j=1, if the key-phrase j is present in the text, and 0 otherwise.

A query is a new (unclassified) document (text) and can be represented in the same manner. The text and query vectors can be envisioned as an n-dimensional vector space. A vector matching operation, based on the cosine correlation used to measure the cosine of the angle between vectors can be used to compute the similarity. Hence, the following equation (adapted from Lucarella D., 1988, [8]) gives us a well-known method to measure the similarity of a text D_i of the training set against a new text D_{new} (or query Q):

$$S(D_i, D_{new}) = \frac{\sum_{j=1}^{m} q_j kp_{ij}}{\sqrt{\sum_{j=1}^{m} q_j^2 \cdot \sum_{j=1}^{m} kp_{ij}^2}} = \frac{\sum_{j=1}^{m} q_j kp_{ij}}{L_{D_{new}} \cdot L_{D_i}}$$

where m is the number of key-phrases used in the collection, kp_{ij} is equal to 1 if the key-phrase j exists in document D_i (of the training set), otherwise is equal to 0 and q_j is the weight of key-phrase j in the new document.

The following equation can be used to measure the term q_j:

$$q_j = \log_2 \left(\frac{ClassCount}{ClassFreq_j} \right)$$

where ClassCount is the number of classes of the training set, and $ClassFreq_j$ is the number of classes that include the key-phrase j.

Experimental results

In the next table (table 1) we depict classes of texts of the training set classified by ICD 9 – codes.

Table 1: Training Set

Class number	ICD9 code that characterizes the class	Number of documents (discharge letters) in class
1	0010	4
2	122.8	4
3	151	4
4	153	4
5	153.3	5
6	154.1	4
7	155.0	4

First, we applied our algorithm to the training set (29 discharge letters) and the «Set of the key-phrases» / «Authority list» was constructed. Then, every text of the training set was submitted as a new text for classification (for assigning the appropriate ICD-9 code). The similarity of the «new» text with all the texts of the training set was calculated using the proposed measure of similarity. In the next table (table 2) we present the number of documents of the same class with the «new» document. More precisely, we present the number of retrieved documents of the same class with the «new» document that belong to the first five more similar ones and the first ten more similar ones, respectively. It seems that we have promissing / encouraging results.

Table 2: Results

Number of most similar documents of the same class	In the five best similar	In the ten best similar
1	6	3
2	14	6
3	7	10
4	2	10

Funds

The work presented in this paper is co-funded by 75% from E.E. and 25% from the Greek Government under the framework of the Education and Initial Vocational Training Program – Archimedes.

References

[1] Ian Witten, Eibe Frank. Data Mining: Practical Machine Learning tools and Techniques with Java implementation. Morgan Kaufmann, 1999, ISBN: 1-55860-552-5.

[2] Helena Ahonen et. al. Mining in the phrasal frontier. Principles of Knowledge Discovery in Databases Conference, Trondheim, Norway, June 1997. Lecture Notes in Computer Science, Springer Verlag, 1997.

[3] Eibe Frank, et. al. Domain-Specific Keyphrase Extraction. International Joint Conference of Artificial Intelligence, 1999.

[4] N. Karanikolas and C. Skourlas. Automatic Diagnosis Classification of patient discharge letters. MIE 2002: XVIIth International Congress of the European Federation for Medical Informatics, August 25-29, 2002, Budapest, Hungary.

[5] N. Karanikolas, C. Skourlas, A. Christopoulou and T. Alevizos. Medical Text Classification based on Text Retrieval techniques. MEDINF 2003. 1st International Conference on Medical Informatics & Engineering, October 9 - 11, 2003, Craiova, Romania.

[6] Heikki Mannila, Hannu Toivonen and A. Inkeri Verkamo. Discovering frequent episodes in sequences. KDD-95: First International Conference on Knowledge Discovery and Data Mining, August 20-21, 1995, Montreal, Canada.

[7] Heikki Mannila and Hannu Toivonen. Discovering generalized episodes using minimal occurrences. KDD-96: Second International Conference on Knowledge Discovery and Data Mining, August, 1996, Portland, Oregon. AAAI Press.

[8] Lucarella, D., 1988, A document retrieval system based on nearest neighbour searching, Journal of Information Science, 14, 25-33.