DOCUMENT CLUSTERIZATION BASED ON ITS TOPIC-ORIENTED REPRESENTATION

Nikita Voronkov

The Department of Applied Mathematics and Computer Science, Belarusian State University, e-mail: *voronkov@scnsoft.com*

Abstract. The article addresses the problem of document clusterization. The author describes a technology for automatic topic extraction from documents and application of the topics as document representation in clusterization task. The topics are understood as noun phrase-based main themes of documents. The suggested algorithm can also be used to improve the quality of automatic multidocument text summarization.

Introduction

Very often people have to deal not with one or dozens of documents, but with hundreds and thousands of documents of different types. And these documents are not sometimes classified according to certain characteristics. Apparently, it becomes very difficult to find any information within large amount of documents without classifications. In such situation users commonly employ different automatic clusterization and summarization systems which aim at reducing information quantity to be processed.

Generally, the task is to classify the input documents into several parts (clusters) in accordance with their relevance (similarity). The quantity of clusters can be specified by user or calculated by the system itself. User can also define properties of the clusters, so that the system could determine to which clusters the documents belong during document analysis and extraction of properties.

We will consider a clusterization task when a user defines the quantity of clusters for the set of documents to be split into, while the system itself will build properties of the clusters. We define topics (main document themes) as cluster properties. To automatically extract topics from the text we use summarization system which builds topic-oriented summary [6].

General scheme of topic-based clusterization algorithm

A process of topic-based clusters building can be described by the following steps:

1. Pre-formatting.
2. Part of speech sentence tagging, extraction of syntactical and semantic relations from sentences (SAO: subject-action-object).
3. Statistics collection.
4. Topic tree building.
5. Main topics extraction.
6. Clusters building.

There are a lot of document formats, so it is necessary to convert them in one standartizated format to process. The input of pre-formatter module is documents in different formats (txt, rtf, doc, pdf, html etc.). The output of this module is text, which is split on sections, paragraphs, titles, etc. Besides, pre-formatter filters any auxiliary information such as buttons' text, scripts, menus and so on.

Then, sentence splitting, part of speech tagging and different types of relation extraction take place [1,3]. After that, the system extracts topics. Topics are the most important simple noun phrases extracted from the document.

Topics extraction

A process of topics extraction is divided into 2 stages: statistics collection and hierarchical topics tree building.

On the stage of statistics collection system accumulates information about words frequency.

To make our statistical algorithm more effective, a set of informative word tags is extracted. Informative words are nouns, adjectives, verbs, adverbs, proper nouns, etc. Uninformative words have tags such as prepositions, articles, numeral and some other tags. Our system takes into account only statistics of informative words.

We also use a set of additional coefficients to improve quality of statistical algorithm. During the processing of each sentence words get some weight taking into consideration which of the following conditions is met:

- word comes across the sentence;
- word is a part of noun phrase;
- word is a part of main semantic/linguistic word of noun phrase;
- word is a part of SAO;
- word is a part of S and O of SAO and main semantic/linguistic word of S and O;
- word is a part of text title.

Every document section has some coefficient of importance of this section in document. So, we define the most important section in the document where the word came across and then we scale word frequency according to coefficient of this section.

After all, we normalize the weights of all the words so that the weight of the most important word becomes equal to 1.

Each simple noun phrase extracted from sentences is a candidate to become a topic [4, 5]. Preliminarily, system transforms each of them according to the following rules:

- split noun phrase by "or" and "and";
- filter uninformative words from noun phrase by tags (articles, etc.);
- filter uninformative words from noun phrase by dictionary ("said", etc.);
- noun phrase transformation (producers of hazardous materials -> hazardous materials producer);
- canonization of main linguistic word (if it is plural, the system makes it singular);

Then, whole noun phrase filtration takes place. Our system makes it in 2 stages:

1. The system filters uninformative noun phrases by tags. For example, if some noun phrase consists of words with special tags or there are some words with non-letter symbols and there is no informative word in this phrase, so this noun phrase is considered as uninformative.
2. On the second stage system filters noun phrases, which consist of special predefined words. For example, "step + letter". In this case noun phrases such as "step A", "step B" etc. will be filtered.

If some noun phrase was not filtered during these 2 stages, we add it to a list of informative noun phrases. We also store location of this noun phrase in the sentence and in the document.

When all sentences of document are processed, we get the list of all informative phrases extracted from text. Then, each noun phrase from this list gets some statistical weight based on word weights, which are counted by statistical algorithm by this time. Noun phrase weight is equal to the arithmetic mean weight of all the words it contains.

Usually, there are a lot of noun phrases in each document and there are a lot of unique noun phrases, so it makes sense to cut off some noun phrases with low weight. For this purpose, all noun phrases are sorted in decreasing order of the weights assigned. Than, it is possible to apply one of the following algorithms to cut off noun phrases:

- leave not more than a predefined number of the heaviest topics;
- leave noun phrases with the weights greater than some predefined value;
- leave noun phrases with the weights greater than the weight of the heaviest noun phrase multiplied by some predefined coefficient;
- among sorted informative noun phrases such a number of the most difficult ones is being chosen, which does not allow their total weight fall below 80% of the total weight of all noun phrases.

For our system we have chosen the last type of noun phrase set cut off with coverage of 80%.

All noun phrases remaining will be called topics. Then, from the topics obtained a hierarchical tree is built.

Topics tree building

To accomplish this, all topics are sorted in increasing order of their lengths (considering informative words only) and within the group of words of the same length topics are sorted in increasing order of their weights. Then, moving from the last list element towards the beginning of the list, the most suitable parent is sought for every topic. The criteria for selection the best parent is the following:

- it is contained in the child;
- it length and weight are the highest.

Checking whether a topic is contained in another topic requires O(l1 + l2) operations, where l1 and l2 – are lengths of the topics in informative words. Finding the best parents requires O(n*n) operations, where n - is a quantity of topics.

However, we can propose some modifications to this algorithm, which show the same complexity, but in practice works 5 to 10 times faster in average.

1. All topics are sorted in increasing order of their lengths (considering informative words only) and within each group of words of the same length topics are sorted in increasing order of their weights;
2. Taking into account that all words in text have unique identifiers from 0 to M - 1, where M is the number of unique canonized wordforms in the text, for each word there is built a list of IDs of noun phrases where the word was found;
3. Moving from the last list element towards the beginning of the list, there is built a list of noun phrase IDs where each ID corresponds to a noun phrase where all informative words of each topic occurred;
4. The list of noun phrase IDs is sorted in decreasing order;
5. The best parent is sought in the order of the above list of possible candidates among those of them that have the ID smaller than the ID of the current noun phrase.

Then, root vertices and all children of each vertex are sorted in decreasing order of the weights. Thus, a hierarchic tree of topics is obtained.

Then, from every tree branch the main topic is selected which is the one that shows the major number of occurrences in the text.

Clusterization

We will focus on the type of clusterization with user-defined number of clusters in which all documents submitted for processing must be grouped.

By the stage of grouping in clusters, each document is presented in the form of a list of topics and weights that represent the degree of importance of each topic with respect to the source document.

Let n be the number of documents submitted for processing and m is the number of clusters requested to build. If $m \geq n$, the task is considered solved. Therefore, we assume that $m < n$.

First, from the list of documents there are chosen m documents that will be the "centers" of future clusters. For this purpose, the following matrix is built:

	Document 1	...	Document n
Document 1	w_{11}		w_{1n}
...	...		
Document n	w_{n1}	...	w_{nn}

where w_{ij} is the degree of proximity of a set of documents which is computed as:

$w_{ij} = \sum \frac{wt_{ik} + wt_{jk}}{2}, i \neq j$ - the sum by common topics of i-th and j-th documents, wt_{ik} - is the weight of k-th common topic in the document i, wt_{jk} - is the weight of k-th common topic in the document j.

$w_{ii} = 0, i = \overline{1,n}$

Then, according to the formula $D_i = \sum_{i=1}^{n} w_{ii}$ D-value is calculated for each document where D is understood as the degree of proximity of all document being processed with respect to the current one.

In the beginning, all documents are marked as non-main. Then, the following steps are performed:

Step 1. The list is sorted in decreasing order of the weights D_i. From the list thus ordered the first document that is not marked as main is chosen. Let M be its index. The document is marked as main and will serve as the center for a cluster. Then the following procedures are applied.

Step 2.

to all documents

if document i is not main and $w_{iM} = \max_{j=\overline{1,n}} w_{ij}$, that is, the most proximate to the current document is the one found as main at the previous step,

then $D_j = D_j - w_{ij}, j = \overline{1,n}, j \neq i$ и $D_i = 0$

The steps 1 and 2 are repeatedly applied m times until requested cluster centers are found.

Then, clusters are built by grouping the documents that have not been marked as main with the main documents to which target non-main document show maximal proximity.

Then, so called cluster centroids are built. Cluster centroid is understood as a set of common topics that are present in all documents of a cluster, or a number of noun groups that have maximal total weight in a cluster.

Conclusions

Proposed clusterization algorithm is not speed efficient but shows rather high quality of cluster building and does not depend on the order the documents are submitted for processing. Moreover, there is a possibility to assign a name to each resulting cluster automatically in form of a set of topics that constitute cluster centroid (topics common to all documents that form a single cluster) or in form of a set of topics most relevant to the cluster. Our method of clusters construction is domain independent so it is possible to process documents of different styles. Proposed clusterization algorithm can also be used in multidocument summarization if the number of documents is relatively small when prior grouping of the documents by their relevance to each other provides for better quality of resulting summary.

References

[1] Advances in Automatic Text Summarization. *The MIT Press*, 1999.

[2] Sovpel I.V. "Issues on the implementation of natural language text analysis" *doctoral thesis*, Minsk, 1980.

[3] Sovpel I.V. "Linguistical and technical principles, methods and algorithms of automatic text analysis", *Vishejshaja Shkola*, Minsk, 1991.

[4] Salton, Gerard, Amit Singhal, Chris Buckley and Mandar Mitra, "Automatic Text Decomposition Using Text Segments And Text Themes", *Proceedings of the Seventh ACM Conference on Hypertext*, Washington D.C., 1996.

[5] Salton, Gerard and Amit Singhal, "Automatic Text Theme Generation and the Analysis of Text Structure", *Cornell Computer Science Technical Report* 94-1438, July 1994.

[6] Voronkov N.V., Sovpel I.V., "Automatic topic-oriented summarization", *Text Processing and Cognitive Technologies. Paper Collection.* №7. Kazan, 2002, 94-102.