Searching for Topics in a Large Collection of Texts

Martin Holub **Jiří Semecký** **Jiří Diviš**
Center for Computational Linguistics
Charles University, Prague
{holub|semecky}@ufal.mff.cuni.cz
jiri.divis@atlas.cz

Abstract

We describe an original method that automatically finds specific topics in a large collection of texts. Each topic is first identified as a specific cluster of texts and then represented as a virtual concept, which is a weighted mixture of words. Our intention is to employ these virtual concepts in document indexing.

In this paper we show some preliminary experimental results and discuss directions of future work.

1 Introduction

In the field of information retrieval (for a detailed survey see e.g. (Baeza-Yates and Ribeiro-Neto, 1999)), document indexing and representing documents as vectors belongs among the most successful techniques. Within the framework of the well known vector model, the indexed elements are usually individual words, which leads to high dimensional vectors. However, there are several approaches that try to reduce the high dimensionality of the vectors in order to improve the effectivity of retrieving. The most famous is probably the method called Latent Semantic Indexing (LSI), introduced by Deerwester et al. (1990), which employs a specific linear transformation of original word-based vectors using a system of "latent semantic concepts". Other two approaches which inspired us, namely (Dhillon and Modha, 2001) and (Torkkola, 2002), are similar to LSI but different in the way how they project the vectors of documents into a space of a lower dimension.

Our idea is to establish a system of "virtual concepts", which are linear functions represented by vectors, extracted from automatically discovered "concept-formative clusters" of documents. Shortly speaking, concept-formative clusters are semantically coherent and specific sets of documents, which represent specific topics. This idea was originally proposed by Holub (2003), who hypothesizes that concept-oriented vector models of documents based on indexing virtual concepts could improve the effectiveness of both automatic comparison of documents and their matching with queries.

The paper is organized as follows. In section 2 we formalize the notion of concept-formative clusters and give a heuristic method of finding them. Section 3 first introduces virtual concepts in a formal way and shows an algorithm to construct them. Then, some experiments are shown. In sections 4 we compare our model with another approach and give a brief survey of some open questions. Finally, a short summary is given in section 5.

2 Concept-formative clusters

2.1 Graph of a text collection

Let $\mathcal{C} = \{d_1, d_2, \ldots, d_N\}$ be a collection of text documents; N is the size of the collection. Now suppose that we have a function $\mathrm{sim}(d_i, d_j) = \mathrm{sim}(d_j, d_i) \in \langle 0, 1 \rangle$, which gives a degree of *document similarity* for each pair of documents. Then we represent the collection as a graph.

Definition: A labeled graph $\mathbf{G}$ is called *graph of collection* $\mathcal{C}$ if $\mathbf{G} = (\mathcal{C}, E)$ where

$$E = \{\, \{d_i, d_j\} \mid i \neq j \,\wedge\, \mathrm{sim}(d_i, d_j) > \mathrm{w}_t\}$$

and each edge $e = \{d_i, d_j\} \in E$ is labeled by number $\mathrm{w}(e) = \mathrm{sim}(d_i, d_j)$, called *weight* of e; $\mathrm{w}_t \geq 0$ is a given document similarity threshold (i.e. a threshold weight of edge).

Now we introduce some terminology and necessary notation. Let $\mathbf{G} = (\mathcal{C}, E)$ be a graph of collection $\mathcal{C}$. Each subset $X \subset \mathcal{C}$ is called a *cut* of $\mathbf{G}$; $\bar{X}$ stands for the complement $\mathcal{C} \setminus X$. If $A, B \subset \mathcal{C}$ are disjoint cuts then

- $\mathrm{e}(A) = \{e \mid e \in E \wedge e \subset A\}$ is a set of edges within cut A;
- $\mathrm{w}(A) = \sum_{e \in \mathrm{e}(A)} w(e)$ is called *weight of cut* A;
- $\mathrm{e}(A, B) = \mathrm{e}(A \cup B) \setminus (\mathrm{e}(A) \cup \mathrm{e}(B))$ is a set of edges between cuts A and B;
- $\mathrm{w}(A, B) = \sum_{e \in \mathrm{e}(A,B)} w(e)$ is called *weight of the connection* between cuts A and B;
- $\bar{\mathrm{w}} = \mathrm{w}(\mathcal{C}) / \binom{N}{2}$ is the *expected weight of edge* in graph $\mathbf{G}$;
- $\bar{\mathrm{w}}(X) = \bar{\mathrm{w}} \cdot \binom{|X|}{2}$ is the *expected weight of cut* X;
- $\bar{\mathrm{w}}(X, \bar{X}) = \bar{\mathrm{w}} \cdot |X| \cdot (N - |X|)$ is the *expected weight of the connection between cut X and the rest of the collection*;
- each cut X naturally splits the collection into three disjoint subsets $\mathcal{C} = X \cup Y_X \cup R_X$ where $Y_X = \{y \in \mathcal{C} \setminus X \mid \mathrm{e}(\{y\}, X) \neq \emptyset\}$ and $R_X = \mathcal{C} \setminus (X \cup Y_X)$.

2.2 Quality of cuts

Now we formalize the property of "being concept--formative" by a positive real function called *quality of cut*. A high value of quality means that a cut must be *specific* and *extensive*.

A cut X is called specific if (i) the weight $\mathrm{w}(X)$ is relatively high and (ii) the connection between X and the rest of the collection $\mathrm{w}(X, \bar{X})$ is relatively small. The first property is called *compactness* of cut, and is defined as $\mathrm{Com}(X) = \mathrm{w}(X) / \bar{\mathrm{w}}(X)$, while the other is called *exhaustivity* of cut, which is defined as $\mathrm{Exh}(X) = \bar{\mathrm{w}}(X, \bar{X}) / \mathrm{w}(X, \bar{X})$. Both functions are positive.

Thus, the specificity of cut X can be formalized by the following formula

$$\left(\frac{\mathrm{w}(X)}{\bar{\mathrm{w}}(X)}\right)^{\lambda_1} \cdot \left(\frac{\bar{\mathrm{w}}(X, \bar{X})}{\mathrm{w}(X, \bar{X})}\right)^{\lambda_2}$$

— the greater this value, the more specific the cut X; λ_1 and λ_2 are positive parameters, which are used for balancing the two factors.

The *extensity* of cut X is defined as a positive function $\mathrm{Ext}(X) = \log_{t_{ext}} |X|$ where t_{ext} is a threshold size of cut.

Definition: The total *quality of cut* $\mathrm{Q}(X)$ is a positive real function composed of all factors mentioned above and is defined as

$$\mathrm{Q}(X) = \mathrm{Com}(X)^{\lambda_1} \cdot \mathrm{Exh}(X)^{\lambda_2} \cdot \mathrm{Ext}(X)^{\lambda_3}$$

where the three lambdas are parameters whose purpose is balancing the three factors.

To be concept-formative, a cut (i) must have a sufficiently high quality and (ii) must be locally optimal.

2.3 Local optimization of cuts

A cut $X \subset \mathcal{C}$ is called *locally optimal* regarding quality function Q if each cut $X' \subset \mathcal{C}$ which is only a small modification of the original X does not have greater quality, i.e. $\mathrm{Q}(X') \leq \mathrm{Q}(X)$.

Now we describe a local search procedure whose purpose is to optimize any input cut X; if X is not locally optimal, the output of the `Local_Search` procedure is a locally optimal cut X^* which results from the original X as its local modification. First we need the following definition:

Definition: *Potential of document* $d \in \mathcal{C}$ with respect to cut $X \subset \mathcal{C}$ is a real function
$\mathrm{P}(d, X)\colon \mathcal{C} \times \mathscr{P}(\mathcal{C}) \longmapsto \mathbf{R}$ defined as

$$\mathrm{P}(d, X) = \mathrm{Q}(X \cup \{d\}) - \mathrm{Q}(X \setminus \{d\}).$$

The `Local_Search` procedure is described in Fig. 1. Note that

1. `Local_Search` gradually generates a sequence of cuts $X^{(0)}, X^{(1)}, X^{(2)}, \ldots$ so that

```
     Input:   the graph of text collection $\mathcal{C}$;
              an initial cut $X = X^{(0)} \subset \mathcal{C}$.
    Output:   locally optimal cut $X^*$.

 Algorithm:   $i \leftarrow 0$
     loop:    $z \leftarrow \arg\min_{x \in X^{(i)}} \mathrm{P}(x, X^{(i)})$
              if $\mathrm{P}(z, X^{(i)}) < 0$ then {
                 $X^{(i+1)} \leftarrow X^{(i)} \setminus \{z\}$
                 $i \leftarrow i + 1$
                 goto loop
              }
              $z \leftarrow \arg\max_{y \in \mathcal{C} \setminus X^{(i)}} \mathrm{P}(y, X^{(i)})$
              if $\mathrm{P}(z, X^{(i)}) > 0$ then {
                 $X^{(i+1)} \leftarrow X^{(i)} \cup \{z\}$
                 $i \leftarrow i + 1$
                 goto loop
              }
              $X^* \leftarrow X^{(i)}$
              end
```

Figure 1: The Local Search Algorithm

(i) $\mathrm{Q}(X^{(i-1)}) < \mathrm{Q}(X^{(i)})$ for $i \geq 1$, and

(ii) cut $X^{(i)}$ always arises from $X^{(i-1)}$ by adding or taking away one document into/from it;

2. since the quality of modified cuts cannot increase infinitely, a finite $k \geq 0$ necessarily exists so that $X^{(k)}$ is locally optimal and consequently the program stops at least after the k-th iteration;
3. each output cut X^* is locally optimal.

Now we are ready to precisely define concept-formative clusters:

Definition: A cut $X \subset \mathcal{C}$ is called a *concept-formative cluster* if

(i) $\mathrm{Q}(X) \geq Q_t$ where Q_t is a threshold quality and

(ii) $X = X^*$ where X^* is the output of the `Local_Search` algorithm.

The whole procedure for finding concept-formative clusters consists of two basic stages: first, a set of *initial cuts* is found within the whole collection, and then each of them is used as a seed for the `Local_Search` algorithm, which locally optimizes the quality function Q.

Note that $\lambda_1, \lambda_2, \lambda_3$ are crucial parameters, which strongly affect the whole process of searching and consequently also the character of resulting concept-formative clusters. We have optimized their values by a sort of machine learning, using a small manually annotated collection of texts. When optimized λ-parameters are used, the `Local_Search` procedure tries to simulate the behavior of human annotator who finds topically coherent clusters in a training collection. The task of λ-optimization leads to a system of linear inequalities, which we solve via linear programming. As there is no scope for this issue here, we cannot go into details.

3 Virtual concepts

In this section we first show that concept-formative clusters can be viewed as fuzzy sets. In this sense, each concept-formative cluster can be characterized by a membership function. Fuzzy clustering allows for some ambiguity in the data, and its main advantage over hard clustering is that it yields much more detailed information on the structure of the data (cf. (Kaufman and Rousseeuw, 1990), chapter 4).

Then we define *virtual concepts* as linear functions which estimate degree of membership of documents in concept-formative clusters. Since virtual concepts are weighted mixtures of words represented as vectors, they can also be seen as virtual documents representing specific topics that emerge in the analyzed collection.

Definition: *Degree of membership* of a document $d \in \mathcal{C}$ in a concept-formative cluster $X \subset \mathcal{C}$ is a function $\mathrm{M}(d, X)$: $\mathcal{C} \times \mathcal{P}(\mathcal{C}) \longmapsto \mathbf{R}$. For $d \in X \cup Y_X$ we define $\mathrm{M}(d, X) = \exp(\alpha \mathrm{P}(d, X))$ where $\alpha > 0$ is a constant. For $d \in R_X$ we define $\mathrm{M}(d, X) = 0$.

The following holds true for any concept-formative cluster X and any document d:

- $\mathrm{M}(d, X) \geq 1$ iff $d \in X$;
- $\mathrm{M}(d, X) \in (0, 1)$ iff $d \in Y_X$.

Now we formalize the notion of virtual concepts. Let $\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_N \in \mathbf{R}^m$ be vector representations of documents $d_1, d_2, \ldots, d_N$, where

```
Input:
  $n$ pairs $\langle \mathbf{d}_1, \mathrm{M}(\mathbf{d}_1)\rangle, \ldots, \langle \mathbf{d}_n, \mathrm{M}(\mathbf{d}_n)\rangle$
      where $\mathbf{d}_1, \ldots, \mathbf{d}_n \in \mathbf{R}^m$;
  $k$ ... maximal number of words in output concept;
  $t\_err$ ... quadratic residual error threshold.
Output:
  $\mathbf{c}^* \in \mathbf{R}^m$ ... output concept;
  $err^*$ ... quadratic residual error;
  $k^*$ ... number of words in the output concept.
Algorithm:
  $I \leftarrow \emptyset, \quad err^* \leftarrow +\infty$
  while $(|I| < k \wedge err^* > t\_err)$ do {
    $err^* \leftarrow +\infty$
    for each $i \in \{1, \ldots, m\} \setminus I$ do {
      $\mathbf{c} \leftarrow$ output of $\mathrm{MLR}(\{\langle \mathbf{d}_j, \mathrm{M}(\mathbf{d}_j)\rangle\}_{j=1}^{n}, I \cup \{i\})$
      $err \leftarrow \sum_{j=1}^{n} (\mathrm{M}(\mathbf{d}_j) - \mathbf{c} \cdot \mathbf{d}_j)^2$
      if $err < err^*$ then {
        $\mathbf{c}^* \leftarrow \mathbf{c}, \quad i^* \leftarrow i, \quad err^* \leftarrow err$
      }
    }
    $I \leftarrow I \cup \{i^*\}$
  }
  $k^* \leftarrow |I|$
  end
```

Figure 2: The Greedy Regression Algorithm

m is the number of indexed terms. We look for such a vector $\mathbf{c}_X \in \mathbf{R}^m$ so that

$$\mathbf{c}_X \cdot \mathbf{d}_i \approx \mathrm{M}(d_i, X)$$

approximately holds for any $i \in \{1, \ldots, N\}$. This vector $\mathbf{c}_X$ is then called *virtual concept corresponding to concept-formative cluster X*.

The task of finding virtual concepts can be solved using the *Greedy Regression Algorithm* (GRA), originally suggested by Semecký (2003).

3.1 Greedy Regression Algorithm

The GRA is directly based on multiple linear regression (see e.g. (Rice, 1994)). The GRA works in iterations and gradually increases the number of non-zero elements in the resulting vector, i.e. the number of words with non-zero weight in the resulting mixture. So this number can be explicitly restricted by a parameter. This feature of the GRA has been designed for the sake of generalization, in order to not overfit the input sample.

The input of the GRA consists of (i) a sample set of document vectors with the corresponding values of $\mathrm{M}(d, X)$, (ii) a maximum number of non-zero elements, and (iii) an error threshold.

The GRA, which is described in Fig. 2, requires a procedure for solving multiple linear regression (MLR) with a limited number of nonzero elements in the resulting vector. Formally, $\mathrm{MLR}(\{\langle \mathbf{x}_j, y_j\rangle\}_{j=1}^{n}, J)$ gets on input

- a set of n vectors $\mathbf{x}_j \in \mathbf{R}^m$;
- a corresponding set of n values $y_j \in \mathbf{R}$ to be approximated; and
- a set of indexes $J \subset \{1, \ldots, m\}$ of the elements which are allowed to be non-zero in the output vector.

The output of the MLR is a vector

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{j=1}^{n} (\mathbf{x} \cdot \mathbf{x}_j - y_j)^2$$

where each considered $\mathbf{x} = \langle x_1, \ldots, x_m\rangle$ must fulfill $x_i = 0$ for any $i \in \{1, \ldots, m\} \setminus J$.

Implementation and time complexity

For solving multiple linear regression we use a public-domain Java package JAMA (2004), developed by the MathWorks and NIST. The computation of inverse matrix is based on the LU decomposition, which makes it faster (Press et al., 1992).

As for the asymptotic time complexity of the GRA, it is in $\mathcal{O}(k \cdot m \cdot \text{complexity\_of\_the\_MLR})$ since the outer loop runs k times at maximum and the inner loop always runs nearly m times. The MLR substantially consists of matrix multiplications in dimension $n \times k$ and a matrix inversion in dimension $k \times k$. Thus the complexity of the MLR is in $\mathcal{O}(k^2 \cdot n + k^3) = \mathcal{O}(k^2 \cdot n)$ because $k < n$. So the total complexity of the GRA is in $\mathcal{O}(m \cdot n \cdot k^3)$.

To reduce this high computational complexity, we make a term pre-selection using a heuristic method based on linear programming. Then, the GRA does not need to deal with high-dimensional vectors in $\mathbf{R}^m$, but works with vectors in dimension $m' \ll m$. Although the acceleration is only linear, the required time has been reduced more than ten times, which is practically significant.

3.2 Experiments

The experiments reported here were done on a small experimental collection of $N = 39,667$

Czech documents. The texts were articles from two different newspapers and one journal. Each document was morphologically analyzed and lemmatized (Hajič, 2000) and then indexed and represented as a vector. We indexed only lemmas of nouns, adjectives, verbs, adverbs and numerals whose document frequency was greater than 10 and less than $20,000$. Then the number of indexed terms was $m = 35,393$. The cosine similarity was used to compute the document similarity; threshold was $\mathrm{w}_t = 0.3$. There were $472,427$ edges in the graph of the collection.

We had computed a set of concept-formative clusters and then approximated the corresponding membership functions by virtual concepts.

The first thing we have observed was that the quadratic residual error systematically and progresively decreases in each GRA iteration. Moreover, the words in virtual concepts are obviously intelligible for humans and strongly suggest the topic. An example is given in Table 1.

words in the concept		**the weights**	
Czech lemma	literally transl.	$k = 5$	$k = 10$
bosenský	Bosnian	1.32	1.01
Srb	Serb	0.67	0.70
UNPROFOR	UNPROFOR	0.59	0.45
OSN	UN	0.57	0.60
Sarajevo	Sarajevo	0.40	0.42
muslimský	Muslim (adj)	—	0.62
odvolat	withdraw	—	3.14
srbský	Serbian	—	0.35
generál	general (n)	—	0.75
list	paper	—	−1.21
quadratic residual error:		3.54	0.260

Table 1: Two virtual concepts ($k = 5$ and $k = 10$) corresponding to cluster #318.

Another example is cluster #19 focused on "pension funds", which was approximated ($k = 20$) by the following words (literally translated):

pension$^+$ (adj), pension$^+$ (n), fund$^+$, additional insurance$^+$, inheritance$^+$, payment$^-$, interest$^+$ (n), dealer$^+$, regulation$^-$, lawsuit$^+$, August$^-$ (adj), measure$^-$ (n), approve$^+$, increase$^+$ (v), appreciation$^+$, property$^+$, trade$^-$ (adj), attentively$^+$, improve$^+$, coupon$^-$ (adj).

(The signs after the words indicate their positive or negative weights in the concept.) Figure 3 shows the approximation of this cluster by virtual concept.

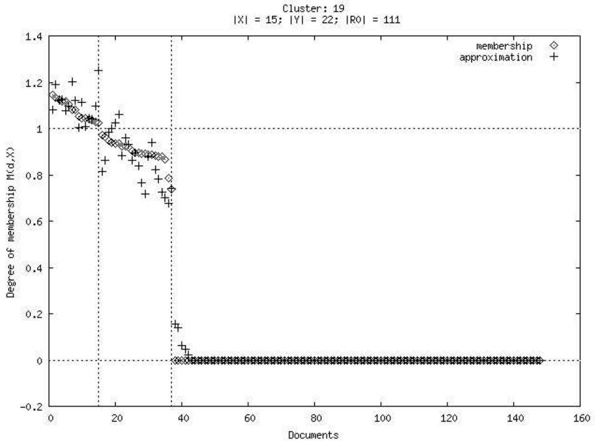

Figure 3: The approximation of membership function corresponding to cluster #19 by a virtual concept (the number of words in the concept $k = 5$).

4 Discussion

4.1 Related work

A similar approach to searching for topics and employing them for document retrieval has been recently suggested by Xu and Croft (2000), who, however, try to employ the topics in the area of distributed retrieval.

They use document clustering, treat each cluster as a topic, and then define topics as probability distributions of words. They use the Kullback--Leibler divergence with some modification as a distance metric to determine the closeness of a document to a cluster. Although our virtual concepts cannot be interpreted as probability distributions, in this point both approaches are quite similar.

The substantial difference is in the clustering method used. Xu and Croft have chosen the K-Means algorithm, "for its efficiency". In contrast to this hard clustering algorithm, (i) our method is consistently based on empirical analysis of a text collection and does not require an a priori given number of topics; (ii) in order to induce permeable topics, our concept-formative clusters are not disjoint; (iii) the specificity of our clusters is driven by training samples given by human.

Xu and Croft suggest that retrieval based on topics may be more robust in comparison with the classic vector technique: Document ranking against a query is based on statistical correlation between query words and words in a document. Since a document is a small sample of text, the statistics in a document are often too sparse to reliably predict how likely the document is relevant to a query. In contrast, we have much more texts for a topic and the statistics are more stable. By excluding clearly unrelated topics, we can avoid retrieving many of the non-relevant documents.

4.2 Future work

As our work is still in progress, there are some open questions, which we will concentrate on in the near future. Three main issues are (i) evaluation, (ii) parameters setting (which is closely connected to the previous one), and (iii) an effective implementation of crucial algorithms (the current implementation is still experimental).

As for the evaluation, we are building a manually annotated test collection using which we want to test the capability of our model to estimate inter--document similarity in comparison with the classic vector model and the LSI model. So far, we have been working with a Czech collection for we also test the impact of morphology and some other NLP methods developed for Czech. Next step will be the evaluation on the English TREC collections, which will enable us to rigorously evaluate if our model really helps to improve IR tasks.

The evaluation will also give us criteria for parameters setting. We expect that a positive value of w_t will significantly accelerate the computation without loss of quality, but finding the right value must be based on the evaluation. As for the most important parameters of the GRA (i.e. the size of the sample set n and the number of words in concept k), these should be set so that the resulting concept is a good membership estimator also for documents not included in the sample set.

5 Summary

We have designed and implemented a system that automatically discovers specific topics in a text collection. We try to employ it in document indexing. The main directions for our future work are thorough evaluation of the model and optimization of the parameters.

Acknowledgments

This work has been supported by the Ministry of Education, project Center for Computational Linguistics (project LN00A063).

References

Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. 1999. *Modern Information Retrieval*. ACM Press / Addison-Wesley.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407.

Inderjit S. Dhillon and D. S. Modha. 2001. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1/2):143–175.

Jan Hajič. 2000. Morphological tagging: Data vs. dictionaries. In *Proceedings of the 6th ANLP Conference, 1st NAACL Meeting*, pages 94–101, Seattle.

Martin Holub. 2003. A new approach to conceptual document indexing: Building a hierarchical system of concepts based on document clusters. In *M. Aleksy et al. (eds.): ISICT 2003, Proceedings of the International Symposium on Information and Communication Technologies*, pages 311–316. Trinity College Dublin, Ireland.

JAMA. 2004. JAMA: A Java Matrix Package. Public-domain, http://math.nist.gov/javanumerics/jama/.

Leonard Kaufman and Peter J. Rousseeuw. 1990. *Finding Groups in Data*. John Wiley & Sons.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. 1992. *Numerical Recipes in C*. Second edition, Cambridge University Press, Cambridge.

John A. Rice. 1994. *Mathematical Statistics and Data Analysis*. Second edition, Duxbury Press, California.

Jiř´ı Semeck´y. 2003. Semantic word classes extracted from text clusters. In *12th Annual Conference WDS 2003, Proceeding of Contributed Papers*. MATFYZPRESS, Prague.

Kari Torkkola. 2002. Discriminative features for document classification. In *Proceedings of the International Conference on Pattern Recognition*, Quebec City, Canada, August 11–15.

Jinxi Xu and W. Bruce Croft. 2000. Topic-based language models for distributed retrieval. In *W. Bruce Croft (ed.): Advances in Information Retrieval*, pages 151–172. Kluwer Academic Publishers.