WordNet Ontology Based Model for Web Retrieval

Václav Snášel, Pavel Moravec
Department of Computer Science
VSB – Technical University of Ostrava
17. listopadu 15, 708 33 Ostrava-Poruba, CZ
{vaclav.snasel,pavel.moravec}@vsb.cz

Jaroslav Pokorný
Department of Software Engineering
Charles University,
Malostranské náměstí 25, 118 00 Prague, CZ
pokorny@ksi.ms.mff.cuni.cz

***Abstract*— It is well known that ontologies will become a key piece, as they allow making the semantics of Semantic Web content [16] explicit. In spite of the big advantages that the Semantic Web promises, there are still several problems to solve. Those concerning ontologies include their availability, development and evolution. In the area of information retrieval, the dimension of document vectors plays an important role. Firstly, with higher index dimensions the indexing structures suffer from the "curse of dimensionality" and their efficiency rapidly decreases. Secondly, we may not use exact words when looking for a document, thus we miss some relevant documents. LSI is a numerical method, which discovers latent semantics in documents by creating concepts from existing terms. In this paper we present a basic method of mapping LSI concepts on given ontology (WordNet), used both for retrieval recall improvement and dimension reduction. We offer experimental results for this method on a subset of TREC collection, consisting of Los Angeles Times articles.**

***Index Terms*— vector model, LSI, information retrieval, WordNet, ontology**

I. Introduction

An ontology is a specification of an abstract, simplified view of the world that we wish to represent for some purpose. This view is called conceptualisation. Therefore, an ontology defines a set of representational terms, that typically include concepts and relations. Interrelationships among the concepts describe a target world. An ontology can be constructed in two ways, domain dependent and generic. CYC, WordNet [1], and Sensus are examples of generic ontologies. In our work we used WordNet because its use is free for research purposes.

For our purpose, we choose a domain-dependent ontology WordNet. This is because, first, a domain-dependent ontology provides concepts in a fine grain, while generic ontologies provide concepts in coarser grain. Second, a generic ontology provides a large number of concepts that may contribute to a mapping of LSI-concepts.

The *information retrieval* [15], [1] deals among other things with storage and retrieval of multimedia data, that can be usually represented as vectors in multi-dimensional space. This is especially suitable for *text retrieval*, where we store a *collection* (or *corpus*) of texts. There are several models used in text retrieval, from which we will use the *vector model* [12], [14] providing qualitatively better results than the *Boolean model* [15], which combines word matching with Boolean operators.

In the vector model, we have to solve several problems. The ones addressed in this paper are the size of resulting index and search efficiency.

Latent semantic indexing (*LSI*) adds an important step to the indexing process. In addition to recording which terms a document contains, the method examines the document collection as a whole, to see which other documents contain some of those same terms. LSI considers documents that have many terms in common to be semantically close, and ones with few words in common to be semantically distant.

To measure the improvement of a new indexing method, we can use several measures, both quantitative and qualitative. The quantitative measures show us the performance of an indexing structure. They include number of disc accesses – *disc access cost* (*DAC*) – or total time of performed indexing and search – *wall clock time*. The qualitative measures tell us how good does this new indexing structure reflect reality when obtaining an *answer set* A for a given *query* Q. The most commonly used qualitative measures are *precision* (P) and *recall* (R) [1].

The paper is a follow-up of the paper presented in

[1] http://www.cogsci.princeton.edu/~wn/

PSMP3 workshop at CIC'04 conference [10]. The rest of this paper is organised as follows. In the second section, we describe classic vector model and above mentioned problems. In the third section, we describe qualitative measures used for evaluation of retrieved data. The fourth section explains the latent semantic indexing method. In the fifth section a basic description of English *WordNet* ontology will be given. In the sixth section we will offer a way how to map LSI concepts on WordNet and in the seventh section experimental results of proposed method on TREC collection. In conclusions we give ideas for future research.

II. Vector Model

In vector model, a document D_j is represented as a vector d_j of term weights, which record the extent of importance of the term for the document.

To portrait the vector model, we usually use an $n \times m$ *term-by-document matrix* A, having n rows – term vectors $t_1 \dots t_n$ – where n is the total number of terms in collection and m columns – document vectors $d_1, \dots d_m$, where m is the size of collection (or *corpus*) C.

Term weights can be calculated in many different ways – $t_i \in \{0, 1\}$, as a membership grade to a fuzzy set, or as a product of functions of term frequency both in a document and in the whole collection [13] (usually $tf * idf$ – count of term occurrences in the document multiplied by a logarithm of the inverse portion of documents containing the term). The normalisation of document vectors is sometimes applied during index generation phase to make the calculation in retrieval phase faster.

A query Q is represented as an n-dimensional vector Q in the same vector space as the document vectors. There are several ways how to search for relevant documents. Generally, we can compute some L_n metrics to represent the similarity of query and document vectors. However, in text retrieval better results can be obtained by computing *cosine measure*:

$$sim(d_j, q) = \frac{d_j \ . \ q}{||d_j||.||q||} = \frac{\sum_{i=1}^{n}(w_{i,j} \ . \ q_i)}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2 \ . \ \sum_{i=1}^{n} q_i^2}}$$

As one can see, we do not only obtain documents which are considered relevant, but according to their distance (or similarity) to the query vector, we can order them and obtain rank for every document in answer set. We can define a *threshold* t, too, meaning that all documents closer than this threshold will be considered relevant, whilst the rest will be irrelevant. However, the choice of the threshold is not exact and its value is usually determined experimentally.

The main problem of vector model is that the document vectors have a big dimension (e.g. 150,000) and are quite sparse (i.e. most coordinates are zero). If we store them as classical vectors, the storage volume is huge – consider size of a term-by-document matrix consisting of 100,000 terms and 200,000 documents.

We can use existing compression schemes for the term-by-document matrix representation like the *compressed column storage* (*CCS*) to decrease memory usage, but then the access time is much longer and we are limited by the fact, that we cannot access the term vectors quickly. Another way is to use combined storage with both CCS and *compressed row storage* (*CRS*). Anyway, updating would still be a problem.

The second problem is the so-called *"curse of dimensionality"*, which causes classical indexing structures like M-trees, A-trees, iDistance, etc. see [5], to perform same or even worse than sequential scan in higher dimension. Moreover, the vectors are placed almost equidistantly from each other, which makes clustering ineffective.

Third, even there is a better chance that we can find relevant documents when using some terms which are not contained in them, the synonyms and other semantically related words are not taken in account.

The first two problems can be addressed for queries containing only a few words by *inverted list*, which is in fact compressed storage of term vectors. Only term vectors for terms contained in a query Q are loaded and processed, computing rank for all documents containing at least one of the terms at once. However, the inverted list is not efficient when searching for similar documents, because significant part of index must be processed.

latent semantic indexing (*LSI*) adds an important step to the indexing process. In addition to recording which terms a document contains, the method examines the document collection as a whole, to see which other documents contain some of those same terms. LSI considers documents that have many terms in common to be semantically close, and ones with few words in common to be semantically distant.

III. Qualitative measures of Retrieval Methods

Since we need an universal evaluation of any retrieval method, we use some measures to determine quality of

such method. In case of Information Retrieval we usually use two such measures - *precision* and *recall*. Both are calculated from the number of objects relevant to the query Rel – determined by some other method, e.g. by manual annotation of given collection and the number of retrieved objects Ret. Based on these numbers we define precision (P) as a fraction of retrieved relevant objects in all retrieved objects and recall (R) as a fraction of retrieved relevant objects in all relevant objects. Formally:

$$P = \frac{|Rel \cap Ret|}{|Rel|} \tag{1}$$

$$R = \frac{|Rel \cap Ret|}{|Ret|} \tag{2}$$

So we can say that recall and precision denote, respectively, completeness of retrieval and purity of retrieval. Unfortunately, it was observed that with the increase of recall, the precision usually decreases. This means that when it is necessary to retrieve more relevant objects, a higher percentage of irrelevant objects will be probably retrieved.

As described in [7], to obtain a single ratio for evaluation of the retrieval performance of two systems, we can employ an F score.

The F score is the harmonic mean of recall and precision, a single measure that combines recall and precision. The function ensures that an F score will have values within the interval [0,1]. The F score is 0 when no relevant documents have been retrieved, and it is 1 when all retrieved documents are relevant. Furthermore, the harmonic mean F assumes a high value only when both precision and recall are high. Therefore, determination of the maximum value for F can be interpreted as an attempt to find the best possible compromise between recall and precision.

The universal version of F score employs a coefficient β, by which can be the precision-recall ratio tuned:

$$F_\beta = \frac{(1+\beta^2) \times P \times R}{\beta^2 P + R} \tag{3}$$

We will use the F score in its basic form, with $\beta = 1$

$$F = \frac{2 \times P \times R}{P + R} \tag{4}$$

IV. Latent Semantic Indexing

LSI [2], [3] is an algebraic extension of classical vector model. First, we decompose the term-by-document matrix A by either *principal component analysis* (*PCA*), which computes eigenvalues and eigenvectors of covariance matrix or *singular value decomposition* (*SVD*), calculating singular values and singular vectors of A. SVD is especially suitable in its variant for sparse matrices Lanczos [8].

Theorem 1 (Singular value decomposition [4]): Let A is an $n \times m$ rank-r matrix. Be $\sigma_1 \geq \ldots \geq \sigma_r$ from eigenvalues of matrix $B_1 = AA^T$ or $B_2 = A^TA$, $\sigma_i = \sqrt{\lambda_i}$ and it holds that $\sigma_i > 0, \sigma_i \geq \sigma_{i+1}$. Then there exist matrices $U = (u_1, \ldots, u_r)$ and $V = (v_1, \ldots, v_r)$, whose column vectors are orthonormal, and a diagonal matrix $\Sigma = diag(\sigma_1, \ldots, \sigma_r)$. Moreover, $U^TU = I_n$ and $V^TV = I_m$. The decomposition $A = U\Sigma V^T$ is called singular decomposition *of matrix A and numbers $\sigma_1, \ldots, \sigma_r$ are* singular values *of the matrix A. Columns of U (or V) are called* left (or right) singular vectors *of matrix A.*

Now we have a decomposition of original term-by-document matrix A. Needless to say, the left and right singular vectors are not sparse. We have at most r nonzero singular numbers, where rank r is smaller of the two matrix dimensions. However, we would not conserve much memory by storing the term-by-document matrix this way. Luckily, because the singular values usually fall quickly, we can take only k greatest singular values and corresponding singular vector co-ordinates and create a *k-reduced singular decomposition* of A.

Definition 1: Let us have $k, 0 < k < r$ and singular value decomposition of A

$$A = U\Sigma V^T = (U_k U_0) \begin{pmatrix} \Sigma_k & 0 \\ 0 & \Sigma_0 \end{pmatrix} \begin{pmatrix} V_k^T \\ V_0^T \end{pmatrix}$$

We call $A_k = U_k \Sigma_k V_k^T$ a k-reduced singular value decomposition (rank-k SVD).

We would not conserve any space with the matrix A_k. So instead of the A_k matrix, a concept-by-document matrix $D_k = V_k \Sigma_k$ with k concepts is used. To execute a query Q in the concept-space, we create a reduced query vector $q_k = U_k^T Q$ [2]. The similarity of terms in concept space can be calculated from $U_k \Sigma_k$ [3].

If every document is relevant to only one topic (for more details see [11]), we obtain a *latent semantics* – semantically related terms will be close in concept space and will result in similar answer set when querying. This addresses the third of problems mentioned in section 2. And since the first co-ordinates of D_k have the greatest

[2]The second approach is to use a matrix $D'_k = V_k$ instead of D_k and $q'_k = U_k^T \Sigma_k^{-1}$

[3]or U_k in second approach

influence on similarity, the clustering results would be better.

The value of k was experimentally determined as several tens or hundreds (e.g. 50–250), exact value of k is however a mystery; it is dependent on the number of topics in collection. For a illustration of rank-k SVD see Figure 1.

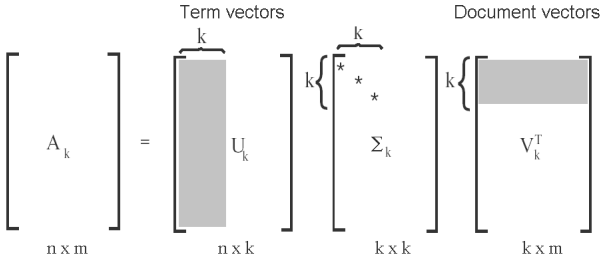

Fig. 1. k-reduced singular value decomposition

Rank-k SVD is the best rank-k approximation of original matrix A. This means, that any other decomposition will increase the sum of squares of matrix $A-A_k$. However, it does not implicate that we could not obtain better precision and recall values with a different approximation.

The LSI is hard to compute and once computed, it reflects only the decomposition of original term-by-document matrix. If several hundreds of documents or terms have to be added to existing decomposition (*folding-in*), the decomposition may become inaccurate. The recalculation of LSI is expensive, so it is impossible to recalculate LSI every time documents and terms are inserted. The *SVD-Updating* [9] is a partial solution, but since the error slightly increases with inserted documents and terms, if the updates happen frequently, the recalculation of SVD may be needed soon or later.

V. WordNet Ontology

WordNet is an online lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory. English nouns, verbs, adjectives and adverbs are organised into synonym sets (*synsets*), each representing one underlying lexical concept.

The goal of WordNet project is the creation of dictionary and thesaurus, which could be used intuitively. The next purpose of WordNet is the support for automatic text analysis and artificial intelligence. WordNet is also useful for determining semantic connections between sets of synonyms, for tracing morphological connections between words.

The ontology is organised not only by the "is-the-synonym-of" relation; the verbs and nouns are hierarchically organised via the *hypernym/hyponym* relation (superior/inferior concepts), too. An example of hypernyms for "ontology" is given in figure 2.

```
psychological feature
 → cognition, knowledge, noesis
  → content, cognitive content, ...
   → knowledge domain, knowledge base
    → discipline, subject, field, ...
     → humanistic discipline, ...
      → philosophy
       → metaphysics
        → ontology
```

Fig. 2. Example of hypernyms for the synset of term "ontology"

EuroWordNet is a multilingual database with WordNets for several European languages (Dutch, Italian, Spanish, German, French, Czech and Estonian). The WordNets are structured in the same way as the American WordNet for English (Princeton WordNet) in terms of synsets (sets of synonymous words) with basic semantic relations between them. Each WordNet represents a unique language-internal system of lexicalizations.

In addition, the WordNets are linked to an *Inter-Lingual-Index*, based on the Princeton WordNet. Via this index, the languages are interconnected so that it is possible to go from the words in one language to similar words in any other language.

This index also gives access to a shared top-ontology of 63 semantic distinctions which provides a common semantic framework for all the languages, while language specific properties are maintained in the individual WordNets. The database can be used, among others, for monolingual and cross-lingual information retrieval, which was demonstrated by the users in the project.

VI. Mapping terms to higher levels of WordNet Ontology

Since we know the hierarchy of concepts defined by the WordNet hypernym organisation, we can use either all hypernyms of given term from l top levels (*top-down*) or hypernyms from ith higher level up to jth one above the term synset to replace terms with ontology concepts (*bottom-up*). In case of $i = j = 0$ we obtain WordNet synsets for terms found in document. Weights

of concepts are created by applying a fraction of term weight dependent on distance from term synset.

It is obvious, that both approaches will improve recall at the expense of precision. However, the F score should not become much worse as mentioned in section II. The dimension will be usually reduced, because every synset contains several terms and especially for bigger collections will be number of used synsets much lower than number of terms.

Because we may still obtain quite a lot of synsets instead of terms, it is a good idea to select only the most interesting ones. This can be done by employing LSI either on whole collection or a collection sample [6].

We are able to identify the most relevant terms for given LSI concept from the term matrix T_k, and when identification of their parent concepts in WordNet ontology will lead to the most interesting WordNet concepts. We may choose a given number of these concepts and ignore the rest Or we can use LSI on WordNet concepts to reduce the dimension even further, either by using only the strongest terms or by using all terms and selecting the strongest generated concepts.

To exemplify first of the above mentioned methods, suppose that we have a collection of text documents. We calculate LSI of such collection into a small dimension (say 50) and obtain 50 LSI concepts, where first of them is the most common one while the last the most special one. Suppose that one of the concepts contains following terms among those with highest weights: *president, monarch, premier*. We map these terms to corresponding synsets and then to higher levels in WordNet hierarchy. If we use a higher level, all these terms will be mapped to concept *head_of_state*, whose weight will be higher than others, which will be generated only by terms with lower weights in this LSI concept or those that do not correlate with other important terms. If we use more levels, other terms will be mapped on *head_of_state*: e.g. names of presidents like *Bush, Clinton, Kennedy*.

The number of retrieved relevant documents (recall) should be higher than with classical vector model, because we are using more general concepts from higher levels of given ontology for indexing and querying; the proof can be found in [7]. However we must expect drop in precision, because of the same reason – suppose we were looking for a *president* and got a document speaking of prime minister. Under certain circumstances this is exactly what we needed, but more often this will result in non-relevant documents in the result set. The F score for our method should behave a little better, because the ontology concepts are generally/almost always relevant especially in a small domain [7].

A problem is that hypernym hierarchy was created only for nouns and verbs. Adjectives and adverbs can't be handled this way, which brings some complications. We can either use all WordNet concepts from this area, or silently ignore all words but nouns and verbs, which will cause either an increase of reduced dimension or worse recall. The same problem is with numbers and names. While a number can be easily identified, we can create a category "Number" to place all numbers in, we cannot do this with some names (like Liu, Hassan, etc.) and other terms, which don't have a counterpart in WordNet. We may consider several alternatives how to solve this problem and

- ignore all such words,
- use existing synsets instead of hypernyms for adjectives and adverbs and ignore the rest (especially if we include term synsets in mapping),
- use all these words as pseudo-concepts,
- or create a fixed number of random concepts as with *Random projection* [11], and use one of the above mentioned methods.

Since the most important terms are usually nouns, we may use the first or the last alternative.

VII. Experimental Results

For the simplicity of LSI calculation, 5000 Los Angeles Times articles (01/1989) from the TREC 5 document collection were indexed. LSI was calculated into a dimension of 200.

Unknown terms, adjectives and adverbs were filtered out; no special heuristic for synset selection was used.

The evaluation of F ratio, precision and recall was made by employing 50 TREC queries. For determination of F ratio, only relevant/non-relevant documents according to TREC query results were used, whilst for classical precision and recall was assumed that all unevaluated documents are non-relevant.

Tests were executed on AMD Athlon 2200+ with VIA KT-400 chipset and 1GB DDR-333 RAM. The LSI and WordNet access routines were written in C/C++; English WordNet 2.0 was used for LSI term mapping.

Firstly, the precision, recall and F score of classical vector model and vector model with LSI-based term filtration (see below) for 200 top concepts were calculated to act as baselines. Threshold for minimal cosine measure was chosen to be 0.2, since the resulting F score of classical vector model was best in this case.

Both methods of mapping (top-down & bottom-up) were tested. Unfortunately, the precision of the top-down

method is too low, being under 0.1%, which might be caused by wrong weight generation. The results could of these method be confusing (with regard to high recall values – almost all documents were selected) and were removed. The bottom-up method behaved much better.

The F score, recall and precision for hypernyms k – $k + l$ levels above the terms' synsets are shown in figures 5 and 3, the X axis determines number of used hypernym levels whilst in the p_z and r_z z means the starting level – number of hypernym levels above term synset. It can be observed, that when going too high in the hierarchy, the effect is similar to the top-down method – the precision drops below 0.1%, which means that almost all documents are selected.

Times of reduced index generation are shown in table I.

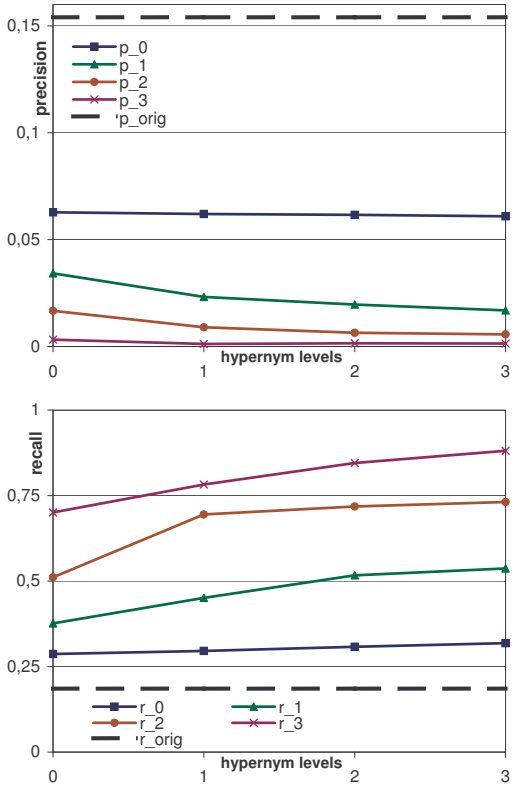

Fig. 3. Precision and recall of original mapping

The original dimension of document collection was 55791; the reduced dimension was between 1747 and 28871, with dimension of 26113 for term synsets only. This dimension may be further reduced by better choice of corresponding synsets or by applying LSI after WordNet mapping.

In second test, the LSI concepts were used for term filtration, at most 250 most important terms in concepts (according to absolute weights in concepts) with weight at least 0.015 were selected and the rest was ignored. The results are shown in figures 4 and 6.

TABLE I
GENERATION TIME OF ORIGINAL MAPPING [S]

	Skipped hypernym levels			
	0	1	2	3
0	212	202	198	190
1	254	229	214	201
2	281	247	226	210
3	301	258	234	214

Rows represent used hypernym level count l.

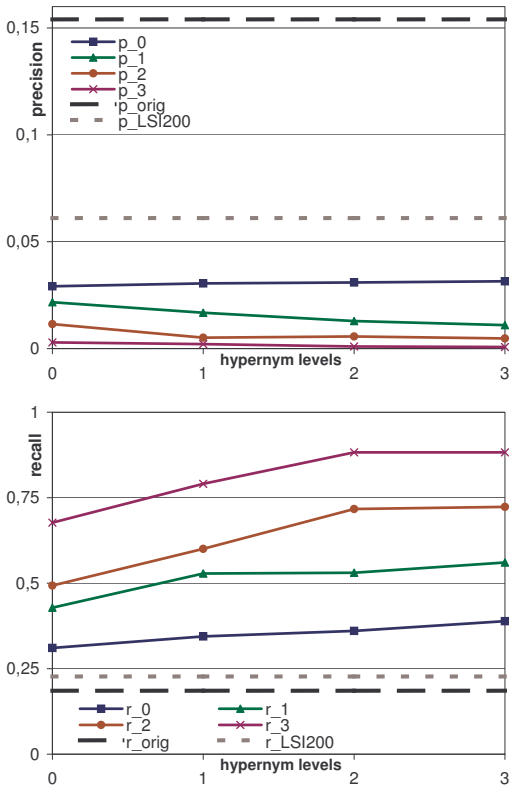

Fig. 4. Precision and recall of LSI-filtered mapping

The description of graphs is same as in case of first experiment, comparison for classical term-by document matrix generated by LSI concepts term filtration was included. Times of reduced index generation with LSI-filtered terms are shown in table II. The LSI calculation time was not included.

We can see that recall increases when we move upwards in hierarchy, which we expected. The absolute

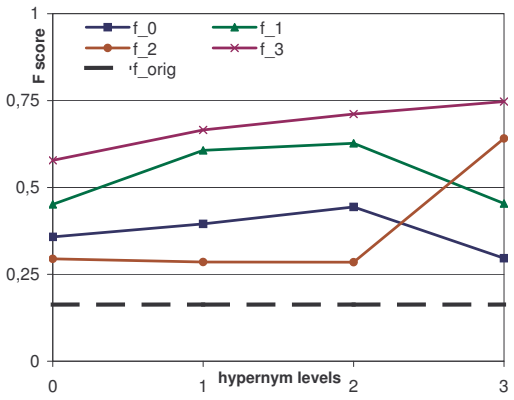

Fig. 5. F score of original mapping

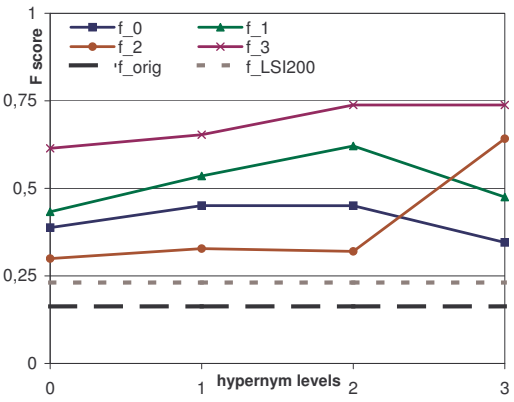

Fig. 6. F score of LSI-filtered mapping

precision seems to drop, however the original precision was not high either, so this may be partly caused by finding new relevant documents. In conclusions we give ideas for future research.

VIII. Conclusion

We have shown, that mapping terms on WordNet hypernyms improves recall, bringing more relevant documents. The LSI filtration enhances recall even more, producing smaller index, too. The question is, whether use expensive method as LSI just for the term filtration. The third approach – using LSI on generated hypernym-by-document matrix has yet to be tested.

At last but not least, we used all synsets given term is included in, not only the most relevant ones. We could drop synsets with weight below given threshold (which has yet to be determined) thus improving the retrieval precision. However, because we don't usually know the order of terms in documents (only the weights), we may not use some lexical constructs like the usage of co-

TABLE II
Generation time of LSI-filtered mapping [s]

	Skipped hypernym levels			
	0	1	2	3
0	40	39	36	35
1	49	44	40	38
2	56	49	43	40
3	60	51	45	40

Rows represent used hypernym level count l.

locations.

We are currently studying two related topics which could improve the retrieval even more. Firstly, if we are able to describe LSI concepts based on terms with highest weights in concept (e.g. *Bush, president, government, senator, state, federal, law* should probably lead to something like *politics*, whilst *Bush, Iraq, troops, Gulf, Afghanistan* to *military*). Secondly, if we can use these descriptions along with known LSI concept hierarchy to create tree-like structure of LSI concepts for every document and using it with tree matching algorithms for more efficient retrieval.

Was the top-level concept approach made usable with higher precision, we could use EuroWordNet's Inter-Lingual-Index for mapping of given concepts. This should enable us to identify texts concerning same topics, which are written in different languages.

Acknowledgment

This research was supported in part by GACR grant 201/03/1318 and Information society project 1ET100300419.

References

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, New York, 1999.

[2] M. Berry and M. Browne. *Understanding Search Engines, Mathematical Modeling and Text Retrieval*. Siam, 1999.

[3] M. Berry, S. Dumais, and T. Letsche. Computation Methods for Intelligent Information Access. In *Proceedings of the 1995 ACM/IEEE Supercomputing Conference*, 1995.

[4] M. W. Berry and R. D. Fierro. Low-Rank Orhogonal Decomposition for Information Retrieval Applications. *Numerical Algebra with Applications*, 1(1):1–27, 1996.

[5] S. B. C. Böhm and D. Keim. Searching in High-Dimensional Spaces Index Structures for Improving the Performance of Multimedia Databases. *ACM, Computing Surveys*, 33(3):322373, 2001.

[6] A. Frieze, R. Kannan, and S. Vempala. Fast Monte-Carlo Algorithms for Finding Low Rank Approximations. In *Proceedings of 1998 FOCS*, pages 370–378, 1998.

[7] L. Khan, D. McLeod, and E. Hovy. Retrieval effectiveness of an ontology-based model for information selection. *The VLDB Journal*, 13:7185, 2004.

[8] R. M. Larsen. Lanczos bidiagonalization with partial reorthogonalization. Technical report, University of Aarhus, 1998.

[9] G. W. O'Brien. Information Management Tools for Updating an SVD-Encoded Indexing Scheme. Technical Report ut-cs-94-258, The University of Tennessee, Knoxville, USA, December, 1994.

[10] J. S. P. Moravec, M. Obitko and V. Snášel. WordNet Ontology Based Model for Information Retrieval. In *Proceedings of CIC'04 Conference*, 2004.

[11] C. H. Papadimitriou, H. Tamaki, P. Raghavan, and S. Vempala. Latent semantic indexing: A probabilistic analysis. In *Prooceedings of the ACM Conference on Principles of Database Systems (PODS)*, pages 159–168, 1998.

[12] G. Salton. *The SMART Retrieval System – Experiments in Automatic Document Processing*. Prentice Hall Inc., Englewood Clifs, 1971.

[13] G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[14] G. Salton and M. Lesk. Computer evaluation of indexing and text processing. *Journal of the ACM*, 15(1):8–39, January 1968.

[15] G. Salton and G. McGill. *Introduction to Modern Information Retrieval.* McGraw-ill, 1983.

[16] O. C. V. Richard Benjamins, Jesús Contreras and A. Gómez-Pérez. Six Challenges for the Semantic Web. *The VLDB Journal*, 1, Issue 3,, 2004.