Using BFA with WordNet Ontology Based Model for Web Retrieval

Václav Snášel, Pavel Moravec
Department of Computer Science, FEECS
VŠB – 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

In the area of information retrieval, the dimension of document vectors plays an important role. We may need to find a few words or concepts, which characterize the document based on its contents, to overcome the problem of the "curse of dimensionality", which makes indexing of high-dimensional data problematic. To do so, we earlier proposed a Wordnet and Wordnet+LSI (Latent Semantic Indexing) based model for dimension reduction. While LSI works on the whole collection, another procedure of feature extraction (and thus dimension reduction) exists, using binary factorization. The procedure is based on the search of attractors in Hopfield-like associative memory. Separation of true attractors (factors) and spurious ones is based on calculation of their Lyapunov function. Being applied to textual data the procedure conducted well and even more it showed sensitivity to the context in which the words were used. In this paper, we suggest that the binary factorization may benefit from the Wordnet filtration.

1. 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 conceptualization. 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. There are still several problems of ontologies to solve [2]. In our work we used WordNet because its use is free for research purposes and being a generic ontology it provides a large number of concepts that may contribute to a mapping of LSI-concepts.

1 `http://www.cogsci.princeton.edu/~wn/`

The *information retrieval* [14, 1] deals among other things with storage and retrieval of multimedia data, that can be usually represented as vectors in multidimensional 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* [13] providing qualitatively better results than the *Boolean model* [14], 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.

Binary factor analysis (*BFA*) maps documents to a (binary) factor space. The original term weights in documents, factors and vectors in factor space are binary, so we don't have to work with weights with unknown meanings as in case of LSI.

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 this new indexing structure reflects 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 rest of this paper is organised as follows. In the second section, we describe classic vector model and above mentioned problems. The third section explains the LSI method. In the fourth section a basic description of English *WordNet* ontology will be given. In the fifth summarise our previous work on mapping of LSI concepts on WordNet, provide an insight on binary factor analysis (BFA) and in the sixth section we propose our improvements.

2. 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 $t_1 \ldots t_n$ – where n is the total number of terms in collection and m columns – document vectors $d_1, \ldots, d_m$, where m is the size of collection (or *corpus*) C.

Term weights can be calculated in many different ways: $w_{ij} \in \{0, 1\}$, 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 (usually $tf.idf$ – count of term occurrences in the document multiplied by a logarithm of the inverse portion of documents containing the term). The normalization of document vectors is sometimes applied during index generation phase to make the calculation in the 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 similarity, usually using the *cosine measure*:

$$sim(d_j, q) = \frac{d_j \cdot q}{||d_j||.||q||}$$

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

The main problem of the vector model is that the document vectors have a big dimension (e.g. 150,000) and are quite sparse (i.e. most co-ordinates 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 to decrease memory usage, but then the access time is much longer and we are limited by the fact, that we cannot access either the term or the document vectors quickly. Another way is to use combined storage with both row and column compression, but updating would still pose 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 in the same way or even worse than sequential scan in higher dimension. Moreover, the vectors are placed almost equidistantly from each other, which makes clustering ineffective.

Third, the synonyms of terms and other semantically related words are not taken into account.

The first two problems can be addressed for queries containing only a few words by *inverted list*, which is in fact a 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.

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.

3. Latent semantic indexing

LSI [4] 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.

Theorem 1 (Singular value decomposition [4]) *Let A is an $n \times m$ rank-r matrix and values $\sigma_1, \ldots, \sigma_r$ are calculated from eigenvalues of matrix AA^T as $\sigma_i = \sqrt{\lambda_i}$. Then there exist column-orthonormal matrices $U = (u_1, \ldots, u_r)$ and $V = (v_1, \ldots, v_r)$, where $U^T U = I_n$ a $V^T V = I_m$, and a diagonal matrix $\Sigma = diag(\sigma_1, \ldots, \sigma_r)$, where $\sigma_i > 0, \sigma_i \geq \sigma_{i+1}$. The decomposition*

$$A = U \Sigma V^T$$

is called singular decomposition *of matrix A and the 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 spare 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 ([4]) *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 $D_k = \Sigma_k V_k^T$ with k rows (called *LSI concepts* instead of terms) is used. To execute a query Q in the concept-space, we create a reduced query vector $q_k = U_k^T q$ (alternatively, we can use $D'_k = V_k^T$ instead of D_k and $q'_k = \Sigma_k^{-1} U_k^T q$ instead of q_k). The similarity (and importance) of terms in concept space can be calculated from the term-by-concept matrix $T_k = U_k \Sigma_k$ (or $T'_k = U_k$ in alternate approach).

If every document is relevant to only one topic (for more details see [12]), 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 the problems mentioned in section 2. And since the first co-ordinates of D_k have the greatest influence on similarity, the clustering results should be better.

The value of k was experimentally determined as several tens or hundreds (e.g. 50–250), it is known to be dependent on the number of topics in collection, however its exact value cannot be simply determined. For a illustration of rank-k SVD see Figure 1.

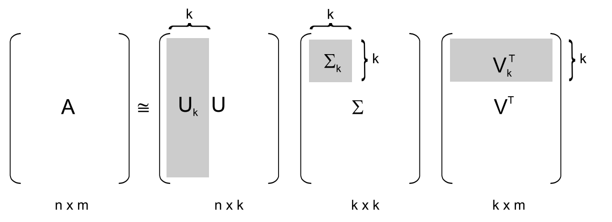

Figure 1. rank-k SVD

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

Since the LSI concepts are a linear combination of original terms (with either positive or negative weights), we can identify the ones most significant ones for each concept (those with extreme weights – either positive or negative) in the term-by-concept matrix. An example of such terms derived from 30,000 Los Angeles Times articles contained in TREC collection (years 1989-1990) is given in figure 3.

bush (0.2449) reagan (0.1731) soviet (0.1215) president (0.1006)
officers (-0.0984) county (-0.1476) council (-0.1657) city (-0.2299) police (-0.2920)

Figure 2. Example of terms with absolute weights over 0.1 for 5th LSI concept

4. 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 sets of synonyms (*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.[2]

The ontology is organised not only by the "is-the-synonym-of" relation; the verbs and nouns are hierarchically organised via the *hypernym/hyponym* (superior/inferior concepts), and *holonym/meronym* (is part of/has parts) relation, too.

An example of hypernyms for “philosophy” is given in figure 3.

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

2 Papers with more detailed description of WordNet and its use can be found at `http://engr.smu.edu/~rada/wnb/`

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

Figure 3. Example of hypernyms for the synset of term "philosophy"

and cross-lingual information retrieval, which was demonstrated by the users in the project.

5. Previous work

In this section, we will shortly describe the mapping terms (from LSI concepts) on WordNet and the Binary Factor Analysis (BFA).

5.1. Mapping terms on WordNet

Our approach, mentioned in [15] is similar to query expansion [16], however we do not only modify the query vectors, but also the document ones, which means we can use document vectors as queries, too.

In the first step we map the terms contained in indexed documents on corresponding synsets. Secondly, we apply a portion of synset weight on semantically related synset (hypernym, part of, . . .).

The dimension will be usually reduced, because every synset contains several terms and especially for bigger collections will be the number of used synsets much lower than the number of terms.

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 concepts.

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. While we can't choose the synsets manually, it can be done by employing LSI (or another technique of detection of important terms in corpus as we will show in next section) either on whole collection or a collection sample [6]. The main idea is to use the ontology to map LSI concepts (C) to synsets, $f : C \longrightarrow Synsets$.

It is obvious, that both our approaches will improve recall at the expense of precision. However, the harmonic mean of precision and recall (F *score*) should not become much worse.

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 the highest weights: *president, monarch, premier*. We map these terms to corresponding synsets and then to more general synsets in WordNet hierarchy. If we use hypernyms, 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*.

Similar approaches were tested independently by other authors. A good survey of syntax-based techniques that are currently used for dimensional reduction is presented in [3].

5.2. Binary factor analysis

Factor analysis is one of the most efficient method to overcome informational redundancy of high-dimensional data set. Factors extraction is a procedure which maps objects from original space variables into the space of factors. Original signals, factor scores and factor loadings are binary, i.e. possess the values 0 or 1. To avoid computational problems with data large dimensionality a procedure of binary nonlinear factorization based on the search of attractors in Hopfield-like associative memory was developed. In this case a complex vector signal (pattern) has a form of the Boolean sum of weighted binary factors:

$$\mathbf{X} = \bigvee \mathbf{S}_l \mathbf{f}^l. \tag{1}$$

A binary factorization neural network with parallel dynamics was recently used [8] because it has a lot of similarities with the iterative procedure for linear factorization. The network starts from a random initial state, network activity stabilizes in some attractor which corresponds to one of true factors or one of spurious factors. To separate true and spurious attractors a procedure based on calculation of their Lyapunov function [9] is used. Unlearning of already found factors prevents their repeated retrieval. Some background on this topic can be found in work [7].

5.2.1. Hopfield network The neural network under consideration consists of N neurons of the McCulloch-Pitts type (integrate-and-fire binary neurons) with gradually ranged synaptic connections between them. Only a fully connected case is considered here.

Network is trained by a set of M patterns of the form $\mathbf{X}^m = \bigvee_{l=1}^{L} \beta_l^m \mathbf{f}^l$, where $\mathbf{f}^l \in B_n^N$ [3] are L factors (N dimensional vectors) and for every m-th pattern $\beta_l^m \in B_C^L$ it is a corresponding factor scores vector. As follows from the definition every factor contains exactly $n = Np$ ones. Every complex pattern $\mathbf{X}^m$ contains in turn exactly the C factors, so it is quite natural to call the *complexity* of the pattern as C. The factors and factor scores are assumed to be statistically independent. In a limit case $C = 1$ patterns become pure factors and we obtain an ordinary Hopfield case.

5.2.2. Learning procedure The connection matrix $\mathbf{J}$ of this network is a covariation matrix of input signals obtained by using the correlational Hebbian learning rule:

$$J_{ij} = \sum_{m=1}^{M} (X_i^m - q^m)(X_j^m - q^m),\ i \neq j,\ J_{ii} = 0, \quad (2)$$

where M is the number of patterns in the learning set and $q^m = \sum_{i=1}^{N} X_i^m / N$ is the total activity of the m-th pattern. Its activity is determined by iterative procedure:

$$X_i(t+1) = \Theta(h_i(t) - T(t)),\ i = 1, \cdots, N \quad (3)$$

where Θ - step function, and $T(t)$ - activation threshold. And third, its activity has following Lyapunov function

$$\Lambda(t+1) = \mathbf{X}^T(t+1)\mathbf{J}\mathbf{X}(\mathbf{t}). \quad (4)$$

Activity of Hopfield-like network with parallel dynamics converges not only to point attractors [9] but also to cyclic attractors of the length two.

Theoretical analysis and computer simulation performed by Frolov et al. [8] completely confirmed the validity of Hopfield-like network for binary factorization. However, Hopfield-like network has one principal peculiarity. The network dynamics converges to one of the factors (true attractor) only when initial state falls inside its attraction basin. Otherwise it converges to one of the spurious attractors. Thus binary factorization requires special recall procedure to separate true and spurious attractors.

5.2.3. Recall procedure To separate true and spurious attractors a two-run recall procedure was developed. Its initialization starts by presentation of random initial pattern $\mathbf{X}^{in}$ with $k_{in} = r_{in}N$ active neurons. On presentation of $\mathbf{X}^{in}$, network activity $\mathbf{X}$ evolves to some attractor. The evolution is determined by equation (3). On each time step k_{in} "winners" (neurons with the greatest synaptic excitation) are chosen and only they are active on the next time step. When activity stabilizes at the initial level of activity k_{in}, $k_{in} + 1$ neurons with maximal synaptic excitation are chosen for the next iteration step, and network activity evolves to some attractor at the new level of activity $k_{in} + 1$. Then level of activity increases to $k_{in} + 2$, and so on, until number of active neurons reaches the final level $r_f N$. Thus, the whole procedure (one trial) contains $(r_f - r_{in})N$ iteration steps and several time steps inside each iteration step to reach some attractor for fixed level of activity.

At the end of each iteration step a relative Lyapunov function was calculated by formula: $\lambda = \Lambda/(rN)$ where Λ is given by (4). The relative Lyapunov function gives a mean synaptic excitation of active neurons. The time course of the relative Lyapunov function along the recall trajectory provides criterion for separation of true and spurious attractors. Attractors with the highest Lyapunov function would be obviously winners in the most trials of the recall process. Thus, more and more trials are required to obtain new attractor with relatively small value of Lyapunov function. To overcome this problem attractors with high Lyapunov function should be deleted from the network memory. The deletion was performed according to Hebbian unlearning rule by subtraction $\Delta J_{ij}, j \neq i$ from synaptic connections J_{ij} where

$$\Delta J_{ij} = \frac{\eta}{2} J(\mathbf{X})[(X_i(t-1) - r)(X_j(t) - r) + (X_j(t-1) - r)(X_i(t) - r)] \quad (5)$$

$J(\mathbf{X})$ is the average synaptic connection between active neurons of the attractor, $\mathbf{X}(t-1)$ and $\mathbf{X}(t)$ are patterns of network activity at last time steps of iteration process, r is the level of activity, and η is an unlearning rate. For point attractor $\mathbf{X}(t) = \mathbf{X}(t-1)$ and for cyclic attractor $\mathbf{X}(t-1)$ and $\mathbf{X}(t)$ are two states of attractor.

More recent information about binary factor analysis are to be published in [11].

6. Proposed fusion of WordNet and BFA

WordNet can be used with BFA in two ways. The first (and expected) one is to map the terms on synsets (or their hierarchies) and use synsets as input to the Hopfield neural network. This should result in factors consisting of (more general) synsets instead of terms.

The second possibility is usage of terms obtained in factors for search of common generalizations. While this approach was not usable in case of LSI concepts because of a high number of terms, it may yield interesting results in case of factors (which contain only a few terms), thus resulting in automatic categorisation. For instance, in an example shown in table 1, the third factor contains words *government, administration, officials* and *Senate*. Government

3 $B_n^N = \{X | X_i \in \{0,1\}, \sum_{i=1}^{N} X_i = n\}$

is the most general one here, being one of the holonyms of administration and Senate (US Government). So we could categorise the message as dealing with government (or politics, if we find a generalisation).

Table 1. Example of resulting factors from [10]

U.S. COMMERCE SECRETARY QUESTIONS FUJITSU DEAL *WASHINGTON*, March 3
Commerce Secretary Malcolm Baldrige said he felt a proposed takeover by **Japan's**^{1} <Fujitsu Ltd> of U.S.-based Fairchild Semiconductor Corp, a subsidiary of Schlumberger Ltd <SLB>, should be carefully reviewed. He **told**^{2} the Semiconductor Industry Association the deal would soon be discussed by representatives of several different **government**^{3} departments. The **Reagan administration**^{3} has previously expressed concern that the proposed takeover would make Fujitsu a powerful part of the U.S. **market**^{1} for so-called supercomputers at a time when **Japan**^{1} has not bought any American-made supercomputers. In addition, U.S. defense **officials**^{3} have said they were worried semiconductor technology could be transferred out of the United States, eventually giving **Japanese**^{1}-made products an edge in American high-technology markets for defense and other goods. Treasury Secretary James Baker recently **told**^{2} a **Senate**^{3} committee the proposed takeover would be reviewed by the cabinet-level **Economic**^{1} Policy Council.

Terms marked ^{1} are contained in the first factor, terms marked ^{2} are common words – contained in both factors and terms marked ^{3} are words contained in the second factor.

7. Conclusion

We have shown on an example output from BFA, that BFA can benefit from ontology (WordNet) mapping. The resulting effect must be however evaluated by further experiments, which requires a future cooperation with the authors of BFA method. Moreover, the WordNet may not be the best type of ontology used for the mapping, because it tries to capture the language properties instead of concept hierarchy. We are currently looking for such ontology, to improve our method.

Acknowledgment

This research was supported in part by Information society project 1ET100300419.

References

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

[2] V. R. Benjamins, J. Contreras, O. Corcho, and A. Gómez-Pérez. Six Challenges for the Semantic Web. *The VLDB Journal*, 1, Issue 3,, 2004.

[3] B. Berendt, A. Hotho, and G. Stumme. Semantic Web Mining and the Representation, Analysis, and Evolution of Web Space. In *Proceedings of RAWS 2005 Workshop*, pages 1–16, Točná, Czech Republic, 2005.

[4] M. Berry, S. Dumais, and T. Letsche. Computational Methods for Intelligent Information Access. In *Proceedings of the 1995 ACM/IEEE Supercomputing Conference*, San Diego, California, USA, 1995.

[5] C. Böhm, S. Berchtold, and D. Keim. Searching in High-Dimensional Spaces – Index Structures for Improving the Performance of Multimedia Databases. *ACM Computing Surveys*, 33(3):322–373, 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] A. Frolov, D. Husek, and P. Muravjev. Informational efficiency of sparsely encoded Hopfield-like autoassociative memory. In *Optical Memory and Neural Networks (Information Optics)*, pages 177–198, 2003.

[8] A. Frolov, A. Sirota, D. Husek, and P. Muravjev. Binary factorization in Hopfield-like neural networks: single-step approximation and computer simulations. In *Neural Networks World*, pages 139–152, 2004.

[9] E. Goles-Chacc and F. Fogelman-Soulie. Decreasing energy functions as a tool for studying threshold networks. In *Discrete Mathematics*, pages 261–277, 1985.

[10] D. Húsek, H. Řezanková, and V. Snášel. Neural Network Nonlinear Factor Analysis of High Dimensional Binary Signals. In *Proceedings of SIP'05*.

[11] D. Húsek, H. Řezanková, V. Snášel, A. Frolov, and P. Polyakov. Neural Network Nonlinear Factor Analysis of High Dimensional Binary Signals. In *Proceedings of SITIS'05*.

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

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

[14] G. Salton and G. McGill. *Introduction to Modern Information Retrieval*. McGraw-ill, New York, USA, 1983.

[15] V. Snášel, P. Moravec, and J. Pokorný. WordNet Ontology Based Model for Web Retrieval. In *Proceedings of WIRI'05 Workshop*, Tokyo, Japan, 2005. IEEE Press.

[16] E. M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 61–69, Dublin, Ireland, 1994.