

Document representation with Generalized Latent Semantic Analysis

Irina Matveeva
Department of Computer
Science
University of Chicago
Chicago, IL 60637
matveeva@cs.uchicago.edu

Ayman Farahat
Xerox Palo Alto Research
Center
3333 Coyote Hill Road
Palo Alto, CA 94304
farahat@parc.com

Christiaan Royer
Xerox Palo Alto Research
Center
3333 Coyote Hill Road
Palo Alto, CA 94304
royer@parc.com

ABSTRACT

Methods for dimensionality reduction, notably LSI, have been successfully applied to the information retrieval task and document classification on small document collections. Since they involve a computation of the eigenvalue or singular value decomposition of a document-term matrix, their use for large real world applications is somewhat limited. In addition to it, the information about the term similarity that these methods can use is inferred only from the current document collection. We present an algorithm that computes a low dimensional vector space representation of documents using point-wise mutual information as a term similarity measure. Point-wise term similarity can be computed using any additional resources, such as the Web. Our method uses the term by term matrix and can therefore be applied to large document collections. Experimental results show a competitive performance on the information retrieval task.

1. INTRODUCTION

There is a wide range of applications that require comparison of text data. Information retrieval, information filtering and text classification are just a few of them. Our approach can be placed among vector space model (VSM) methods to information retrieval (IR) and text classification. The approach is to compute a vector space representation of the documents and to use the inner product or cosine between the document vectors as a measure of similarity between the documents. Finding a meaningful vector space representation of text data is a difficult problem. Different approaches have been developed to overcome the term independence assumption of the traditional bag-of-words (BOW) [22] representation by such modifications of the representation space as expanding the document vectors with synonyms and related terms [6, 18], using n-grams and phrases [?, ?] as well as distributional term clusters [24, 4, 3] to represent

dimensions, just to name a few.

Multidimensional scaling (MDS) comprises a family of methods to compute a vector space representation for data for which only pairwise similarities or dissimilarities are available. Latent Semantic Analysis (LSA) [11] is one of the best known MDS algorithms used in IR. One of the advantages of the MDS framework is that the dimensions in the resulting vector space can be seen as semantic concepts and that the analysis of the semantic relatedness between terms is performed implicitly, in the course of a matrix decomposition. LSA was applied to the IR tasks, but its computational cost seemed to be higher than the performance gain, so it did not find a wide use in the IR community. The reason for LSA performance, however, is the particular similarity measure that it uses rather than the idea to use dimensionality reduction.

In this paper we investigate more general use of MDS for a low dimensional vector space representation of vocabulary terms. Generalized Latent Semantic Analysis (GLSA) is a framework to combine different measures of semantic association between terms and different methods of dimensionality reduction.

Our approach takes advantage of recent developments in many areas: availability of large document collections such as the Web and successful use of point-wise mutual information as a term similarity measure alternative to LSA, development of new graph-based dimensionality reduction algorithms, development of parallel linear algebra algorithms. In this paper, we show that GLSA algorithms, in particular a combination of point-wise mutual information to obtain the similarities matrix and metric MDS achieve good performance on the IR task.

2. DIMENSIONALITY REDUCTION FOR DOCUMENT REPRESENTATION

There has been a considerable amount of research of how to use dimensionality reduction techniques for document representation.

LSA makes the assumption that there is a latent semantic structure in a given document collection that is often obscured due to synonymy and polysemy [11]. LSA reveals the semantic structure and indexes documents not by individual terms, but by latent semantic concepts represented by them. There have been many attempt to improve LSA performance [2, 10].

The bag-of-concepts (BOC) vector space representation of documents was recently investigated by Sahlgren et al. [21]. This representation is computed as Random Indexing (RI) by projecting the BOW document vectors on random low dimensional vectors. Sahlgren et al. introduce a new method of iterative computation of the document vectors that overcomes the difficulties associated with high computational cost of the LSA decomposition.

LSA and RI were developed to compute a vector space representation of documents. They can also be seen as methods of obtaining low dimensional term vectors but their duality ties the computation of term vectors to a particular document collection,

Terms offer a much greater flexibility in exploring similarity relations than documents. Semantic similarities between terms can be analyzed using any large document collection such as the Web. The Word Space Model (WSM) developed by Schuetze [23, 29] for word sense disambiguation is another special case of MDS that computes the term vectors directly. Instead of using document co-occurrence statistics, it uses term co-occurrence in the contexts of the top most frequent informative terms. Widdows [29] used WSM vectors to represent documents for IR, but restricted their computation to the given document collection. The similarities between the terms and the top informative terms can be computed in a number of ways, but usually normalized counts are used [23, 29]. Schuetze experimented with different term association measures but found that normalized counts give best performance [23, 29]. LSA and WSM differ in the ways they obtain pair-wise term similarities. After that the eigenvalue decomposition of the similarities matrix is used to obtain the term vectors.

Our approach was in part motivated by the large amount of interest for using large Web-based document collections to analyze semantic relations between terms such as synonymy [28, 27, 8]. Interestingly, Steyvers et al. [25] use MDS for term representation but used human judgments to obtain a similarities matrix. MDS preserved similarities expressed in the input matrix, as will be discussed below in detail, and their experiments confirmed that this mathematical property of MDS carries out to preserving human association judgments. One of the important issues that we explore here is what methods of obtaining corpus-based measures of term association can be combined with MDS.

Similarity measure is not the only parameter in this setting, one can make use of different methods for dimensionality reduction as well. Laplacian Eigenmaps Embedding [5], Isomap [26], Locality Preserving Projections (LPP) [16] are examples of graph-based modifications of the MDS approach to dimensionality reduction. Latent Dirichlet Associations (LDA) [7] algorithm offers a method for computing term vectors within a generative language model based on the idea of latent semantic analysis.

Within the Generalized Latent Semantic Analysis (GLSA) framework we focus on the vector space representation of term vectors. This allows us to generalize the use of dimensionality reduction in two ways. First, we consider different measures of semantic association for the term similarity matrix. We no longer consider term-document matrix computed used some particular document collection. Instead, we construct the term similarities matrix using such large resources as the Web and then use the term vectors to represent the document in any given collection.

Second, we propose the use of different dimensionality reduction techniques. The vocabulary size is in many cases smaller than the size of the document collection. Thus, we can use metric MDS as well as the new graph-based algorithms, such as Laplacian Eigenmaps Embedding.

All algorithms within the GLSA framework deal with synonymy and polysemy in the same way as LSA, by providing a similar notion of extracting latent semantic concepts.

The rest of the paper is organized as follows. Section 3 contains the outline of the GLSA algorithm, and discussed the method of dimensionality reduction and term association measure used in this paper. In section 4 presents the results of the IR experiments followed by discussion in section 5.

3. GENERALIZED LATENT SEMANTIC ANALYSIS

3.1 GLSA Algorithm

As noted above, we take a different approach to computing a vector space representation of documents by generating the term vectors first. The GLSA algorithm has the following steps. Assume that we have a document collection C with N documents that we would like to use for IR, the vocabulary size for this collection is V . We also have a large Web based corpus W .

1. Construct the weighted term document matrix $D_{V \times N}$ using C
2. Obtain a matrix of pair-wise similarities $S_{V \times V}$ using W
3. Obtain a low dimensional vector space representation of terms that preserves these similarities, $U_{k \times V}$
4. Compute document vectors by taking linear combinations of term vectors

$$\hat{D} = U_k D$$

The columns of \hat{D} are documents in the k dimensional space

It is evident from the above description that this approach can be combined with any kind of similarity measure on the space of terms as well as any method of dimensionality reduction. The traditional term document matrix is used in the last step to provide the weights in the linear combination of term vectors. We used entropy-based weights that proved very effective in a number of applications, eg [12].

As mentioned above, to be applicable to the information retrieval task, an algorithm should have a procedure of folding-in out-of-collection document, i.e. of computing a vector space representation of queries in the same space as document vectors. Since GLSA computes term vectors, query vectors are obtained in the same way as document vectors, namely as linear combination of term vectors.

The construction of document vectors as weighted linear combinations of the term vectors is an established procedure, used by Choi et al. [13] in the LSA setting, in the Word Space model by Schuetze [23], and recently by Sahlgren et al. [21].

We will briefly discuss two methods of dimensionality reduction that we used in the paper: classical metric MDS [9] and Laplacian Eigenmaps Embedding [5] as well as some

measures of term association. We are currently working on extending our system for combine more different techniques.

3.2 MDS

The classical MDS approach begins with a matrix of pairwise distances or dissimilarities and to compute a vector space representation so that the distances between the resulting vectors are close to the original dissimilarities [9]. Thus, one can obtain a vector space representation for the data for which there is no meaningful a priori vector space representation. The classical MDS method uses the dissimilarities to compute a similarities matrix. Therefore, we can apply MDS if we begin with a matrix of pairwise similarities.

Eckart and Young [14] have shown that given any matrix X and its singular-value decomposition

$$X = U\Sigma V^T,$$

the matrix

$$X_k = U_k \Sigma_k V^T$$

obtained by setting all but the first k diagonal elements in Σ to zero is the rank k matrix that minimizes

$$\|X - X_k\|_F^2$$

U and V are column orthogonal matrices containing the left and the right singular vectors, respectively. Σ is a diagonal matrix containing the singular values sorted in the decreasing order.

If we take X to be a symmetric positive semi-definite matrix, for instance a matrix of inner products between the data points, the left singular vectors in U are the eigenvectors of X and the diagonal elements in Σ are the corresponding eigenvalues. In other words,

$$X = U\Sigma U^T,$$

In this case we can assume that there is a vector space representation of the data such that the inner products between the data vectors correspond to the entries of X : $X = DD^T$. If we require the data vector to have dimensionality k , we are looking for the best rank k approximation to X in term of the Frobenius norm:

$$\min \|X - DD^T\|_F^2$$

It can be seen from the above, that vector space representation of the data that best preserves the original similarities in X is given by the first k eigenvectors of X : $D = U_k \Sigma^{\frac{1}{2}}$.

3.2.0.1 GLSA with MDS.

There is one issue about using MDS in our approach that should be addressed. In the experiments presented in the following sections, the term similarity matrix S contains the PMI scores for each pair of terms:

$$S[i][j] = PMI(t_i, t_j).$$

The k dimensional term vectors are obtained from the eigenvectors of S :

$$\hat{D} = U_k \Sigma^{\frac{1}{2}}$$

Clearly, S has to be positive semi-definite (psd) for this approach to work. Generally, the matrix S may not be positive semi-definite. However, T. Cox and M. Cox mention

that in practice MDS can be applied to non-psd matrices as well. If the negative eigenvalues are small in absolute value they make no significant contribution to the values of distance and inner product in the MDS space and can safely be ignored ([9], p. 37). This is particularly true, when one the first k largest eigenvalues are used. In either case, the number of dimensions of the MDS space is limited to the number of positive eigenvalues.

3.3 Laplacian Eigenmaps Embedding

The Laplacian Eigenmaps Embedding [5] algorithm has the following setup. We are given n data points as vectors in R^N x_1, \dots, x_n that belong to k dimensional manifold M embedded in R^N . The goal is to find a lower dimensional representation for these points $y_1, \dots, y_n \in R^k$, where $k < N$. First, a neighborhood graph of the data is constructed. In the simplest case, it is a n -nearest neighbors graph $G = (V, E)$. The adjacency matrix W can contain either binary or any real weights associated with the edges. The local similarities can be inferred from the adjacency matrix. The value $W[i][j]$ is non-zero if the corresponding data points are similar. The graph Laplacian L is computed as $L = D - W$, where D is a diagonal matrix and $D[i][i] = \sum_j W[i][j]$.

The main contribution of this approach is that introduces the notion of preserving the similarity only locally. This can be seen from its objective function. The problem is to minimize under certain constraints

$$\min \sum_{ij} \|y_i - y_j\|^2 W[i][j]$$

There is a penalty if similar points have such a representation that maps them far apart. The eigenvectors of the Laplacian of the graph are used as a new representation of the data.

The input to this algorithm is the adjacency matrix of the data. In our case, the data points are terms and the adjacency matrix can be constructed using pairwise term similarity. Thus, we can use this algorithm instead of MDS to compute a low dimensional vector space representation for terms.

3.4 Measure of Semantic Association

GLSA focuses on the vector space representation of the vocabulary terms. To be able to apply the classical MDS as well as other dimensionality reduction techniques, we first have to obtain a matrix of semantic association between pairs of terms.

There is a number of well established methods to compute collection-based term associations, such as point-wise mutual information, likelihood ratio, χ^2 test etc. [19]. All these measures suffer from data sparsity, the availability of large collections such as the Web, however, might help to alleviate this problem. The main applications of these measures of pairwise term associations were collocation discovery and semantic proximity tests [19, 28, 27, 8, 30]. In the synonymy test [28, 27] these measures are applied to a small number of carefully selected term pairs which is quite different from the GLSA situation when all pairwise comparisons are needed. We were interested in exploring the possibility of using these similarity measures to compute the GLSA similarities matrix.

We used the following three association measures that are based on the term co-occurrence statistics. We write the

number of co-occurrences of terms x and y as $C(x, y)$, the number of term occurrences as $C(x)$ and $C(Y)$, the total number of documents is N . All the measures that we used serve as comparison test between the assumption that the terms occur independently and the assumption that they are dependent.

3.4.1 χ^2 test

The values assigned by this test are obtained from the 2-dimensional contingency table associated with each term pair. Following Schuetze, [23], we compute the following value:

$$\chi^2 = \frac{N(N_{++}N_{--} - N_{+-}N_{-+})^2}{(N_{++} + N_{+-})(N_{-+}N_{--})(N_{++}N_{-+})(N_{+-}N_{--})},$$

where $N_{++} = C(x, y)$, $N_{+-} = C(x) - C(x, y)$, $N_{-+} = C(y) - C(x, y)$ and $N_{--} = N - N_{++} - N_{+-} - N_{-+}$.

3.4.2 Log Likelihood ratio

In this case the above two hypothesis are tested under the assumption of the binomial term distribution. $H1$ is that $P(x|y) = P(x|\neg y)$ and $H2$ is that $P(x|y) \neq P(x|\neg y)$. These hypothesis determine the parameter of the binomial distribution and so the likelihood of the data under $H1$ and $H2$ can be computed as $L(H1)$ and $L(H2)$, see [19] for details.

$$\log \lambda = \log \frac{L(H1)}{L(H2)}$$

Following the discussion in [19], we used $-2 * \log(\lambda)$.

3.4.3 Point-wise mutual information

If we take a pair of vocabulary terms, t_1 and t_2 and map them into binary random variables, X and Y , we can compute their point-wise mutual information as the amount of information that occurrence of t_1 contains about the occurrence of t_2 . X equals 1 if t_1 appears in a given document and equals 0 otherwise. The PMI similarity is computed as

$$\begin{aligned} PMI(t_1, t_2) &= \log \frac{P(X=1, Y=1)}{P(X=1)P(Y=1)} = \\ &= \log \frac{P(Y|X)}{P(Y)} \end{aligned}$$

3.4.4 PMI-based score

Turney [28] used statistical information from the Web to compute point-wise mutual information (PMI) between vocabulary terms and obtained better results on the synonyms test than LSA. In this paper, we are particularly interested in PMI as a term similarity measure because it has a number of successful applications. Lin and Pantel [20] used PMI as a measure of term similarity to compute document clusters, Chklovski and Pantel [8] use web-based PMI to extract semantic relations between verbs. It also has been shown to perform better than other statistical measures of association. Terra and Clarke [27] studied a number of different measures of word association computed using a terabyte-size corpus of Web data applied to the synonyms task and showed that PMI achieves the best results.

Manning [19] observed that one of the weaknesses of PMI is that the measure of dependency between pairs of

dependent terms is influenced by the individual counts of the terms, as the terms get rare, the dependency measure increases. This may not be optimal for collocation discovery. It would be interesting to understand whether large corpora help to alleviate this problem and whether this property of PMI makes is particularly suitable for GLSA.

The above approaches consider pairs of words and use PMI as association measure. We use PMI to obtain a matrix of pair-wise term similarities and apply dimensionality reduction to compute term vectors.

3.5 Low-dimensional Term Vectors

In the experiments presented in this paper we used classical MDS [9] and Laplacian Embedding algorithm [5] to compute a vector space representation for terms.

In particular, we performed step 3 in the following two ways.

1. Step 3 with MDS

- Compute the eigenvalue decomposition of S

$$S = U\Sigma U^T$$

- Take the k eigenvectors corresponding to the k largest eigenvalues, U_k . The rows of $U_k \Sigma_k^{-\frac{1}{2}}$ represent the term vectors in the k dimensional space

2. Step 3 with Laplacian Embedding

- Compute the nearest neighbors adjacency matrix W , $W[i][j] > 0$ if terms i and j are similar
- Compute the diagonal degree matrix D , $D[i][i] = \sum_{j=1}^N W[i][j]$
- Compute the graph Laplacian L , $L = D - W$
- Compute the eigenvectors of L

$$LU = \lambda U$$

- Take the k eigenvectors corresponding to the k smallest eigenvalues, U_k . The rows of U_k represent the term vectors in the k dimensional space

In the first case, we obtain term vectors that best preserve pair-wise similarities in S , in the second case we obtain term vectors that are close if their corresponding similarity value in S is large.

The advantage is our approach is twofold. First, we begin with a term similarity matrix to compute a vector representation for terms based exclusively on this matrix. PMI has been shown to be a good term similarity measure in a variety of applications. In addition to it, our approach only requires an eigenvalue decomposition of a term by term matrix. The size of the vocabulary is usually much smaller than the collection size. More importantly, the PMI score can be computed using any other resources apart from the document collection.

4. EXPERIMENTS

In the experiments reported here we used a Web-based collection with 4,428,708 documents.

4.1 Term Co-occurrence Counts

We computed the vocabularies for each of the collections and for each pair of terms x and y , we obtained the counts $C(x, y)$, $C(x)$ and $C(y)$ using the documents in the collection. Based on these counts, we computed similarity matrices using different measures of terms association. We then used MDS and Laplacian Embedding to obtain term vectors. The rest of the procedure is as outlined in 3.1. The pair-wise term co-occurrences can be computed in two ways. One approach is to count the number of documents in which the terms co-occurred. This corresponds to using the AND operator available in many search engines. The other approach is to restrict the co-occurrence of two terms to a window of a particular size. In this case, NEAR operator can be used. Turney [28] showed that the use of NEAR can lead to a substantial performance improvement. In the experiments reported in this paper we obtained the counts using AND and NEAR operators, for NEAR we used a window of size 16. Yet another method would be to use sliding windows of certain size to count the number of “close” co-occurrences of term pairs, as in [27]. Since it accounts for a number of times term co-occur in a particular document and for terms frequencies in the documents, this method can be similar to applying the popular tf-idf weighting scheme [17]. We are currently extending our system to include this method. Unless stated otherwise, we use PMI as the term association measure.

4.2 Retrieval Experiments

We compared the performance of GLSA with MDS ($GLSA_{MDS}$) and the GLSA with Laplacian Embedding ($GLSA_{LE}$) with LSA and PLSA using four medium sized document collections: MED (1033 document abstracts from the National Library of Medicine, CRAN (1400 documents from the Cranfield Institute of Technology), CISI (1460 abstracts in library science from the Institute for Scientific Information) and CACM (3204 abstracts of articles of Communications of ACM). Each of these document collections has a number of queries and the relevance information. For each query we compute the values of precision and recall using the standard 11 points. The precision values are averaged over all queries and used for performance comparison. We report the average of precision at 0.2, 0.5 and 0.7 levels of recall.

4.3 Data Preprocessing

We constructed the term-document matrix for each of the document collections using the standard preprocessing methods such as stop word removal. We split compound words that contain “-” and kept only words that did not contain special characters. One of the important choices was to use either unstemmed or stemmed words for indexing. We used both approaches in the experiments reported here. Cases when no stemming was applied are marked with “I”, and the cases with stemming are marked with “II”.

4.4 Out-of-vocabulary Terms

At this stage of the development of our system, we use the fixed Web-based collection we described above. Since we work with two sets of the vocabulary terms: from the standard collections (retrieval) and the Web-based collection (term counts) it may happen that we need counts for terms that are not present in the Web-based collection. This could be expected since some of the standard collections are very

Data	LSA		$GLSA_{MDS}$		
	II	I	II	I	II
Med	0.59	0.46	0.71	0.65	0.59
CISI	0.23	0.26	0.17	0.19	0.14
CACM	0.27	0.36	0.29	0.24	0.19
CRAN	0.42	0.45	0.45	0.37	0.31
TIME	0.69	N/A	0.68		
NPL	0.21	N/A	0.18		
ADI					

Table 1: Performance comparison of term matching (TM), LSA, $GLSA_{MDS}$ with and without stemming, “I” and “II”, respectively. We show the best performance over different numbers of dimensions, where appropriate. We used the AND operator for $GLSA_{MDS}$.

Data	AND(II)	NEAR(II)
Med	0.59	0.63
CRAN	0.31	0.40
CACM	0.19	0.23
CISI	0.14	

Table 2: Performance comparison for preprocessing modus “II” with stemming and operators AND and NEAR.

technical. We are currently working on including a back-off to the Web in cases when we need counts for a word that is not present in our Web-based collection. Since we only need ratios of counts, the information about the terms co-occurrence from multiple sources can be combined easily.

4.5 Computational Complexity

The main part of the GLSA algorithms is to compute the eigenvalue of the terms similarity matrix. We used the PLAPACK package ¹ to do this computation. Bientinesi et al. [1] report that they could compute an eigenvalue decomposition of a 128,000x128,000 dense matrix on a 256 processor machine with 2 Gbytes of memory per processor in 8 hours and 24 minutes. While this amount of computational resources may be available for example through TeraGrid ², the vocabulary size of many real-world collections is still much larger and might include over a million of words (TREC collections).

One of the possible solutions to this problem is to use a subset of the vocabulary that contains semantically most informative words for the dimensionality reduction, for example as it was done recently by Griffiths et al. [15].

5. DISCUSSION

Table 1 shows the average precision for term matching (TM) as baseline, for LSA, $GLSA_{MDS}$ using the best values over a range of embedding dimensions. We also experimented with the operators AND and NEAR for obtaining the term counts. GLSA shows a competitive performance compared to LSA. The results we obtain for LSA are very similar to those reported elsewhere for log-entropy weight-

¹PLAPACK package is available at <http://www.cs.utexas.edu/users/plapack/>

²<http://www.teragrid.org/>

Data	PMI	χ^2	Log L
Med(NEAR)	0.63	0.49	
Med(AND)	0.59	0.53	
CRAN(NEAR)	0.40	0.26	
CRAN(AND)	0.31	0.28	

Table 3: Performance comparison for different association measures.

Data	$GLSA_{MDS}$		$GLSA_{LE}$	
	AND	NEAR	AND	NEAR
Med	0.59	0.63	0.60	0.60
CISI	0.14		0.22	
CACM	0.19	0.23	0.19	0.23
CRAN	0.31	0.40	0.39	0.39

Table 4: Performance comparison of $GLSA_{MDS}$ and $GLSA_{LE}$ using preprocessing with stemming, AND and NEAR operators.

ing scheme used here [12]. These results show that GLSA’s performance trend is similar to that of LSA. LSA achieves a large improvement for the MED collection, but is not better than term matching for CISI. These experiments also showed us what extensions of our system would be most valuable for GLSA performance. We discuss some of them below.

Since term matching and LSA use only one document collection, the use of stemming does not make much difference and we only report the results when stemming was used.

These results show that the data preprocessing is very important when two collections are used. It seems that the performance without stemming is better. However, it is clear that in both cases the counts are biased and we are currently working different preprocessing formats for the Web-based collection to ensure that we obtain unbiased counts with stemming.

We are also working on the problem of out-of-vocabulary terms. We had a number of terms in each of the standard collections for which we could not obtain co-occurrence counts. In many cases these are technical terms such as “benzpyrene”, “homotransplantation”. We believe that the performance of GLSA will improve when we implement the back-off to the Web (Google returned 8,340 documents that contain “benzpyrene” and 7,150 for “homotransplantation”).

Using the preprocessing with stemming, we evaluated the influence of AND and NEAR operators for obtaining the terms co-occurrence counts. Table 2 contains the results. Similar to findings of Turney [28] and Terra and Clarke [27], the use of NEAR operator leads to an improved performance.

Table 3 shows the results when different association mea-

Data	AND(I)	NEAR(I)
Med	0.65	0.60
CRAN	0.37	0.39
CACM	0.24	
CISI	0.19	

Table 5: Performance comparison for preprocessing modus “I” without stemming and operators AND and NEAR.

asures are used. As expected, GLSA with PMI shows the best performance. Interestingly, χ^2 has the opposite performance trend to PMI, the use of the NEAR operator hurts the performance.

Table 4 shows the result when MDS and Laplacian Embedding algorithms are used to perform dimensionality reduction. It can be seen that the Laplacian Embedding term vectors have very similar performance to MDS. However, the Laplacian Embedding offers big computation advantage over MDS. When metric MDS is used, the eigenvalue decomposition is done for the dense similarities matrix whereas in the Laplacian Embedding algorithm the eigenvalue decomposition is done for a very sparse graph adjacency matrix. We used normalized Laplacian, see [5] for details, in our experiments. Due to the relation between the Laplacian Embedding and MDS, this can be seen as a principled why to sparsify the originally dense matrix of pair-wise similarities by setting the entries for all but the top n most similar terms to zero. This embedding algorithm also offers a good way of using such association measures as χ^2 and log likelihood ratios where not the actual computed values are useful but the indication of whether the hypothesis of independence is rejected.

6. CONCLUSION

We presented an algorithmic framework of Generalized Latent Semantic Analysis for computing a low dimensional vector space representation for documents. This approach combines the insights from Web-based synonymy extraction and dimensionality reduction for information retrieval.

We consider document vectors to be a linear combination of low dimensional term vectors. A low dimensional vector space is computed using the point-wise mutual information or other corpus-based measures of semantic association between terms as similarity measure. Once we have pair-wise similarity information, we can apply MDS or any other dimensionality reduction method to obtain term vectors.

One of the advantages of GLSA over LSA is that linguistically well founded pair-wise similarity information for terms can be obtained without using term-document vectors explicitly. Consistent with other research we found that point-wise mutual information (PMI) is well suited for this application. Semantic relationship between terms is not dependent on a particular document collection, so we can use additional resources such as the Web to refine our PMI computations.

We are planning to carry out a similar performance comparison between our method and LSA, PLSA using large document collections from Trec and CLEF. Since with our method we can combine different approaches to obtaining term similarities and dimensionality reduction, we also want to investigate what term similarity measures other than PMI can be used and use different dimensionality reduction techniques.

7. ACKNOWLEDGMENTS

We are very grateful to Paolo Bientinesi for providing the PLAPACK package and his help to adapt it to our task.

8. REFERENCES

[1]

- [2] R. K. Ando. Latent semantic space: iterative scaling improves precision of inter-document similarity measurement. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 216–223. ACM Press, 2000.
- [3] L. D. Baker and A. K. McCallum. Distributional clustering of words for text classification. In W. B. Croft, A. Moffat, C. J. van Rijsbergen, R. Wilkinson, and J. Zobel, editors, *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, pages 96–103, Melbourne, AU, 1998. ACM Press, New York, US.
- [4] R. Bekkerman, R. El-Yaniv, and N. Tishby. Distributional word clusters vs. words for text categorization.
- [5] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.
- [6] N. J. Belkin and W. B. Croft. Information filtering and information retrieval: two sides of the same coin? *Commun. ACM*, 35(12):29–38, 1992.
- [7] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation, 2002.
- [8] T. Chklovski and P. Pantel. Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP-04)*, 2004.
- [9] T. Cox and M. Cox. *Multidimensional Scaling*. CRC/Chapman and Hall, 2001.
- [10] N. Cristianini, J. Shawe-Taylor, and H. Lodhi. Latent semantic kernels. *J. Intell. Inf. Syst.*, 18(2-3):127–152, 2002.
- [11] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.
- [12] S. Dumais. Enhancing performance in latent semantic indexing, 1990.
- [13] P. W.-H. Freddy Choi and J. Moore. Latent semantic analysis for text segmentation.
- [14] G. Golub and C. Reinsch. *Handbook for Matrix Computation II, Linear Algebra*. Springer-Verlag, New York, 1971.
- [15] T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum. Integrating topics and syntax. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA, 2005.
- [16] X. He and P. Niyogi. Locality preserving projections. In *NIPS*, 2003.
- [17] K. S. Jones. A statistical interpretation of term specificity and its application in retrieval. *J. Documentation*, 28(1):11–21, 1972.
- [18] G.-A. Levow, D. W. Oard, and P. Resnik. Dictionary-based techniques for cross-language information retrieval. (to appear) *Information Processing and Management: Special Issue on Cross-language Information Retrieval*.
- [19] C. Manning and H. Schuetze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, 1999.
- [20] P. Pantel and D. Lin. Document clustering with committees. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 199–206. ACM Press, 2002.
- [21] M. Sahlgren and R. Cster. Using bag-of-concepts to improve the performance of support vector machines in text categorization. In *20th International Conference on Computational Linguistics, COLING*, pages 487–493, 2004.
- [22] G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.
- [23] H. Schuetze. Automatic word sense discrimination. *Computational Linguistics*, 24(21):97–124, 1998.
- [24] N. Slonim and N. Tishby. Document clustering using word clusters via the information bottleneck method. In *Research and Development in Information Retrieval*, pages 208–215, 2000.
- [25] M. Steyvers, R. M. Shiffrin, and D. L. Nelso. Semantic spaces based on free association that predict memory performance. *JEP General*.
- [26] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- [27] E. L. Terra and C. L. A. Clarke. Frequency estimates for statistical word similarity measures. In *HLT-NAACL*, 2003.
- [28] P. D. Turney. Mining the web for synonyms: Pmi-ir versus lsa on toefl. *Lecture Notes in Computer Science*, 2167:491–??, 2001.
- [29] D. Widdows. A mathematical model for context and word-meaning. In *CONTEXT*, pages 369–382, 2003.
- [30] D. Widdows. Unsupervised methods for developing taxonomies by combining syntactic and statistical information. In *HLT-NAACL*, 2003.