Improving Probabilistic Latent Semantic Analysis with Principal Component Analysis

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

Francine Chen
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304
chen@fxpal.com

Abstract

Probabilistic Latent Semantic Analysis (PLSA) models have been shown to provide a better model for capturing polysemy and synonymy than Latent Semantic Analysis (LSA). However, the parameters of a PLSA model are trained using the Expectation Maximization (EM) algorithm, and as a result, the trained model is dependent on the initialization values so that performance can be highly variable. In this paper we present a method for using LSA analysis to initialize a PLSA model. We also investigated the performance of our method for the tasks of text segmentation and retrieval on personal-size corpora, and present results demonstrating the efficacy of our proposed approach.

1 Introduction

In modeling a collection of documents for information access applications, the documents are often represented as a "bag of words", i.e., as term vectors composed of the terms and corresponding counts for each document. The term vectors for a document collection can be organized into a term by document co-occurrence matrix. When directly using these representations, synonyms and polysemous terms, that is, terms with multiple senses or meanings, are not handled well. Methods for smoothing the term distributions through the use of latent classes have been shown to improve the performance of a number of information access tasks, including retrieval over smaller collections (Deerwester et al., 1990), text segmentation (Brants et al., 2002), and text classification (Wu and Gunopulos, 2002).

The *Probabilistic Latent Semantic Analysis* model (PLSA) (Hofmann, 1999) provides a probabilistic framework that attempts to capture polysemy and synonymy in text for applications such as retrieval and segmentation. It uses a mixture decomposition to model the co-occurrence data, and the probabilities of words and documents are obtained by a convex combination of the aspects. The mixture approximation has a well defined probability distribution and the factors have a clear probabilistic meaning in terms of the mixture component distributions.

The PLSA model computes the relevant probability distributions by selecting the model parameter values that maximize the probability of the observed data, *i.e., the likelihood function*. The standard method for maximum likelihood estimation is the Expectation Maximization (EM) algorithm. For a given initialization, the likelihood function increases with EM iterations until a local maximum is reached, rather than a global maximum, so that the quality of the solution depends on the initialization of the model. Additionally, the likelihood values across *different* initializations are not comparable, as we will show. Thus, the likelihood function computed over the training data cannot be used as a predictor of model performance across different models.

Rather than trying to predict the best performing model from a set of models, in this paper we focus on finding a good way to initialize the PLSA model. We will present a framework for using Latent Semantic Analysis (LSA) (Deerwester et al., 1990) to better initialize the parameters of a corresponding PLSA model. The EM algorithm is then used to further refine the initial estimate. This combination of LSA and PLSA leverages the advantages of both.

This paper is organized as follows: in section 2, we review related work in the area. In section 3, we summarize related work on LSA and its probabilistic interpretation. In section 4 we review the PLSA model and in section 5 we present our method for initializing a PLSA model using LSA model parameters. In section 6, we evaluate the performance of our framework on a text segmentation task and several smaller information retrieval tasks. And in section 7, we summarize our results and give directions for future work.

2 Background

A number of different methods have been proposed for handling the non-globally optimal solution when using EM. These include the use of Tempered EM (Hofmann, 1999), combining models from different initializations in postprocessing (Hofmann, 1999; Brants et al., 2002), and trying to find good initial values. For their segmentation task, Brants et al. (2002) found overfitting, which Tempered EM helps address, was not a problem and that early stopping of EM provided good performance and faster learning. Computing and combining different models is computationally expensive, so a method that reduces this cost is desirable. Different methods for initializing EM include the use of random initialization e.g., (Hofmann, 1999), k-means clustering, and an initial cluster refinement algorithm (Fayyad et al., 1998). K-means clustering is not a good fit to the PLSA model in several ways: it is sensitive to outliers, it is a hard clustering, and the relation of the identified clusters to the PLSA parameters is not well defined. In contrast to these other initialization methods, we know that the LSA reduces noise in the data and handles synonymy, and so should be a good initialization. The trick is in trying to relate the LSA parameters to the PLSA parameters.

LSA is based on singular value decomposition (SVD) of a term by document matrix and retaining the top K singular values, mapping documents and terms to a new representation in a *latent semantic space*. It has been successfully applied in different domains including automatic indexing. Text similarity is better estimated in this low dimension space because synonyms are mapped to nearby locations and noise is reduced, although handling of polysemy is weak. In contrast, the PLSA model distributes the probability mass of a term over the different latent classes corresponding to different senses of a word, and thus better handles polysemy (Hofmann, 1999). The LSA model has two additional desirable features. First, the word document co-occurrence matrix can be weighted by any weight function that reflects the relative importance of individual words (e.g., tf-idf). The weighting can therefore incorporate external knowledge into the model. Second, the SVD algorithm is guaranteed to produce the matrix of rank k that minimizes the distance to the original word document co-occurrence matrix.

As noted in Hofmann (1999), an important difference between PLSA and LSA is the type of objective function utilized. In LSA, this is the L2 or Frobenius norm on the word document counts. In contrast, PLSA relies on maximizing the likelihood function, which is equivalent to minimizing the cross-entropy or Kullback-Leibler divergence between the empirical distribution and the predicted model distribution of terms in documents.

A number of methods for deriving probabilities from LSA have been suggested. For example, Coccaro and Jurafsky (1998) proposed a method based on the cosine distance, and Tipping and Bishop (1999) give a probabilistic interpretation of principal component analysis that is formulated within a maximum-likelihood framework based on a specific form of Gaussian latent variable model. In contrast, we relate the LSA parameters to the PLSA model using a probabilistic interpretation of dimensionality reduction proposed by Ding (1999) that uses an exponential distribution to model the term and document distribution conditioned on the latent class.

3 LSA

We briefly review the LSA model, as presented in Deerwester et al. (1990), and then outline the LSA-based probability model presented in Ding (1999).

The term to document association is presented as a term-document matrix

$$\mathbf{X} = \begin{pmatrix} x_1^1 & \cdots & x_n^1 \\ \vdots & \ddots & \vdots \\ x_1^m & \cdots & x_n^m \end{pmatrix} = (\mathbf{x_1} \cdots \mathbf{x_n}) = \begin{pmatrix} \mathbf{t_1} \\ \vdots \\ \mathbf{t_m} \end{pmatrix} \quad (1)$$

containing the frequency of the m index terms occurring in n documents. The frequency counts can also be weighted to reflect the relative importance of individual terms (e.g., Guo et al., (2003)). $\mathbf{x_i}$ is an m dimensional column vector representing

document i and $\mathbf{t_j}$ is an n dimensional row vector representing term j. LSA represents terms and documents in a new vector space with smaller dimensions that minimize the distance between the projected terms and the original terms. This is done through the truncated (to rank k) singular value decomposition $\mathbf{X} \approx \mathbf{X_k} = \mathbf{U_k \Sigma_k V_k^T}$ or explicitly

$$\mathbf{X_k} = (\mathbf{u_1} \cdots \mathbf{u_k}) \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{pmatrix} \begin{pmatrix} \mathbf{v_1} \\ \vdots \\ \mathbf{v_k} \end{pmatrix}. \tag{2}$$

Among all $m \times n$ matrices of rank k, $\mathbf{X_k}$ is the one that minimizes the Frobenius norm $||\mathbf{X} - \mathbf{X_k}||_F^2$.

3.1 LSA-based Probability Model

The LSA model based on SVD is a dimensionality reduction algorithm and as such does not have a probabilistic interpretation. However, under certain assumptions on the distribution of the input data, the SVD can be used to define a probability model. In this section, we summarize the results presented in Ding (1999) of a dual probability representation of LSA.

Assuming the probability distribution of a document $\mathbf{x_i}$ is governed by k characteristic (normalized) document vectors, $\mathbf{c_1} \cdots \mathbf{c_k}$, and that the $\mathbf{c_1} \cdots \mathbf{c_k}$ are statistically independent factors, Ding (1999) shows that using maximum likelihood estimation, the optimal solution for $\mathbf{c_1}, \cdots \mathbf{c_k}$ are the left eigenvectors $\mathbf{u_1} \cdots \mathbf{u_k}$ in the SVD of $\mathbf{X}$ used in LSA:

$$p(\mathbf{x_j}|\mathbf{u_1} \cdots \mathbf{u_k}) = \frac{e^{(\mathbf{x_j} \cdot \mathbf{u_1})^2 + \cdots + (\mathbf{x_j} \cdot \mathbf{u_k})^2}}{Z(\mathbf{u_1} \ldots \mathbf{u_k})} \tag{3}$$

where $Z(\mathbf{u_1} \cdots \mathbf{u_k})$ is a normalization constant. The dual formulation for the probability of term $\mathbf{t}$ in terms of the tight eigenvectors (*i.e.,* the document representations $\mathbf{v_1} \cdots \mathbf{v_k}$) of the matrix $\mathbf{X_k}$ is:

$$p(\mathbf{t_i}|\mathbf{v_1} \cdots \mathbf{v_k}) = \frac{e^{(\mathbf{t_i} \cdot \mathbf{v_1})^2 + \cdots + (\mathbf{t_i} \cdot \mathbf{v_k})^2}}{Z(\mathbf{v_1} \cdots \mathbf{v_k})} \tag{4}$$

where $Z(\mathbf{v_1} \cdots \mathbf{v_k})$ is a normalization constant. Ding also shows that $\mathbf{u_j}$ is related to $\mathbf{v_j}$ by:

$$\mathbf{u_j} = \frac{1}{\sigma_j} \mathbf{X} (\mathbf{v_j})^T \quad j = 1, \cdots, k \tag{5}$$

We will use Equations 3-5 in relating LSA to PLSA in section 5.

4 PLSA

The PLSA model (Hofmann, 1999) is a generative statistical latent class model: (1) select a document d with probability $p(d)$ (2) pick a latent class z with probability $p(z|d)$ and (3) generate a word w with probability $p(w|z)$, where

$$p(w|d) = \sum_z p(w|z) p(z|d). \tag{6}$$

The joint probability between a word and document, $p(d, w)$, is given by

$$\begin{aligned} p(d, w) &= p(d) p(w|d) \\ &= p(d) \sum_z p(w|z) p(z|d) \end{aligned}$$

and using Bayes' rule can be written as:

$$p(d, w) = \sum_z p(w|z) p(d|z) p(z). \tag{7}$$

The likelihood function is given by

$$\mathcal{L} = \sum_d \sum_w n(d, w) \log p(d, w). \tag{8}$$

Hofmann (1999) uses the EM algorithm to compute optimal parameters. The *E-step* is given by

$$p(z|d, w) = \frac{p(z) p(d|z) p(w|z)}{\sum_{z'} p(z') p(d|z') p(w|z')} \tag{9}$$

and the *M-step* is given by

$$p(w|z) = \frac{\sum_d n(d, w) p(z|d, w)}{\sum_{d, w'} n(d, w') p(z|d, w')} \tag{10}$$

$$p(d|z) = \frac{\sum_w n(d, w) p(z|d, w)}{\sum_{d', w} n(d', w) p(z|d', w)} \tag{11}$$

$$p(z) = \frac{\sum_{d, w} n(d, w) p(z|d, w)}{\sum_{d, w} n(d, w)}. \tag{12}$$

4.1 Model Initialization and Performance

An important consideration in PLSA modeling is that the performance of the model is strongly affected by the initialization of the model prior to training. Thus a method for identifying a good initialization, or alternatively a good trained model, is needed. If the final likelihood value obtained after training was well correlated with accuracy, then one could train several PLSA models, each with a different initialization, and select the model with the largest likelihood as the best model. Although, for a given initialization, the likelihood

Table 1: Correlation between the negative log-likelihood and Average or BreakEven Precision

Data	# Factors	Average Precision	BreakEven Precision
Med	64	-0.47	-0.41
Med	256	-0.15	0.25
CISI	64	-0.20	-0.20
CISI	256	-0.12	-0.16
CRAN	64	0.03	0.16
CRAN	256	-0.15	0.14
CACM	64	-0.64	0.08
CACM	256	-0.22	-0.12

increases to a locally optimal value with each iteration of EM, the final likelihoods obtained from different initializations after training do not correlate well with the accuracy of the corresponding models. This is shown in Table 1, which presents correlation coefficients between likelihood values and either average or breakeven precision for several datasets with 64 or 256 latent classes, i.e., factors. Twenty random initializations were used per evaluation. Fifty iterations of EM per initialization were run, which empirically is more than enough to approach the optimal likelihood. The coefficients range from -0.64 to 0.25. The poor correlation indicates the need for a method to handle the variation in performance due to the influence of different initialization values, for example through better initialization methods.

Hofmann (1999) and Brants (2002) averaged results from five and four random initializations, respectively, and empirically found this to improve performance. The combination of models enables redundancies in the models to minimize the expression of errors. We extend this approach by replacing one random initialization with one reasonably good initialization in the averaged models. We will empirically show that having at least one reasonably good initialization improves the performance over simply using a number of different initializations.

5 LSA-based Initialization of PLSA

The EM algorithm for estimating the parameters of the PLSA model is initialized with estimates of the model parameters $p(z), p(w|z), p(d|z)$. Hofmann (1999) relates the parameters of the PLSA model to an LSA model as follows:

$$\mathbf{P} = \mathbf{U}_{plsa}\Sigma_{plsa}\mathbf{V}_{plsa}^T \quad (13)$$

$$\mathbf{U}_{plsa} = (P(d_i|z_k))_{i,k} \quad (14)$$

$$\mathbf{V}_{plsa} = (P(w_j|z_k))_{j,k} \quad (15)$$

$$\Sigma = diag(P(z_k))_k. \quad (16)$$

Comparing with Equation 2, the k LSA factors, $\mathbf{u_i}$ and $\mathbf{v_j}$ correspond to the factors $p(w|z)$ and $p(d|z)$ of the PLSA model and the mixing proportions of the latent classes in PLSA, $p(z)$, correspond to the singular values of the SVD in LSA. Note that we can not directly identify the matrix $\mathbf{U}_k$ with $\mathbf{U}_{plsa}$ and $\mathbf{V}_k$ with $\mathbf{V}_{plsa}$ since both $\mathbf{U_k}$ and $\mathbf{V_k}$ contain negative values and are not probability distributions. However, using equations 3 and 4, we can attach a probabilistic interpretation to LSA, and then relate $\mathbf{U}_{plsa}$ and $\mathbf{V}_{plsa}$ with the corresponding LSA matrices. We now outline this relation.

Equation 4 represents the probability of occurrence of term $\mathbf{t_j}$ in the different documents conditioned on the SVD right eigenvectors. The j, k^{th} element in equation 15 represent the probability of term w_j conditioned on the latent class z_k. As in the analysis above, we assume that the latent classes in the LSA model correspond to the latent classes of the PLSA model. Making the simplifying assumption that the latent classes of the LSA model are conditionally independent on term $\mathbf{t_j}$, we can express the $p(\mathbf{t_j}|\mathbf{v_1} \dots \mathbf{v_k})$ as:

$$\begin{aligned} p(\mathbf{t_j}|\mathbf{v_1} \dots \mathbf{v_k}) &= \frac{p(\mathbf{v_1} \dots \mathbf{v_k}|\mathbf{t_j})p(\mathbf{t_j})}{p(\mathbf{v_1} \dots \mathbf{v_k})} \\ &= \frac{p(\mathbf{t_j})p(\mathbf{v_1}|\mathbf{t_j}) \cdots p(\mathbf{v_k}|\mathbf{t_j})}{p(\mathbf{v_1}) \cdots p(\mathbf{v_k})} \\ &= p^{1-k}(\mathbf{t_j}) \prod_{p=1}^{k} p(\mathbf{t_j}|\mathbf{v_p}). \quad (17) \end{aligned}$$

And using Equation (4) we get:

$$p^{1-k}(\mathbf{t_j}) \prod_{p=1}^{k} p(\mathbf{t_j}|\mathbf{v_p}) = \frac{\prod_{p=1}^{k} e^{(\mathbf{t_j} \cdot \mathbf{v_p})^2}}{Z(\mathbf{v_1} \dots \mathbf{v_k})} \quad (18)$$

Thus, other than a constant that is based on $p(\mathbf{t_j})$ and $Z(\mathbf{v})$, we can relate each $p(\mathbf{t_j}|\mathbf{v_p})$ to a corresponding $e^{(\mathbf{t_i} \cdot \mathbf{v_j})^2}$. We make the simplifying assumption that $p(\mathbf{t_j})$ is constant across terms and normalize the exponential term to a probability:

$$p(\mathbf{t_i}|\mathbf{v_j}) = \frac{e^{(\mathbf{t_i} \cdot \mathbf{v_j})^2}}{\sum_{k=1}^{d} e^{(\mathbf{t_k} \cdot \mathbf{v_j})^2}}$$

Relating the term w_i in the PLSA model to the distribution of the LSA term over documents, t_i, and relating the latent class z_j in the PLSA model

to the LSA right eigenvector v_j, we then estimate $p(w_i|z_j)$ from $p(\mathbf{t_i}|\mathbf{v_j})$, so that:

$$p(w_i|z_j) \approx \frac{e^{(\mathbf{t_i}\cdot\mathbf{v_j})^2}}{\sum_{k=1}^{d} e^{(\mathbf{t_k}\cdot\mathbf{v_j})^2}} \quad (19)$$

Similarly, relating the document d_j in the PLSA model to the distribution of LSA document over terms, x_j, and using Equation 5 to show that $\mathbf{v_j}$ is related to $\mathbf{u_j}$ we get:

$$p(d_i|z_j) \approx \frac{e^{(\mathbf{x_i}\cdot\mathbf{u_j})^2}}{\sum_{k=1}^{d} e^{(\mathbf{x_k}\cdot\mathbf{u_j})^2}} \quad (20)$$

The singular values, σ_i in Equation 2, are by definition positive. Relating these values to the mixing proportions, $p(z_i)$, we generalize the relation using a function $f()$, where $f()$ is any non-negative function over the range of all σ_i, and normalize so that the estimated $p(z_i)$ is a probability:

$$p(z_i) \approx \frac{f(\sigma_i)}{\sum_{i=1}^{k} f(\sigma_i)} \quad (21)$$

We have experimented with different forms of $f()$ including the identity function and the logarithmic function. For our experiments, we used $f(\sigma) = \log(\sigma)$.

In our LSA-initialized PLSA model, we initialize the PLSA model parameters using Equations 19-21. The EM algorithm is then used beginning with the E-step as outlined in Equations 9-12.

6 Results

In this section we evaluate the performance of LSA-initialized PLSA (LSA-PLSA). We compare the performance of LSA-PLSA to LSA only and PLSA only, and also compare its use in combination with other models. We give results for a smaller information retrieval application and a text segmentation application, tasks where the reduced dimensional representation has been successfully used to improve performance over simpler word count models such as *tf-idf*.

6.1 System Description

To test our approach for PLSA initialization we developed an LSA implementation based on the SVDLIBC package (http://tedlab.mit.edu/∼dr/SVDLIBC/) for computing the singular values of sparse matrices. The PLSA implementation was based on an earlier implementation by Brants et al. (2002). For each of the corpora, we tokenized the documents and used the Xelda morphological analyzer to stem the terms. We used entropy weights (Guo et al., 2003) to weight the terms in the document matrix.

6.2 Information Retrieval

We compared the performance of the LSA-PLSA model against randomly-initialized PLSA and against LSA for four different retrieval tasks. In these tasks, the retrieval is over a smaller corpus, on the order of a personal document collection. We used the following four standard document collections: (i) MED (1033 document abstracts from the National Library of Medicine), (ii) CRAN (1400 documents from the Cranfield Institute of Technology), (iii) CISI (1460 abstracts in library science from the Institute for Scientific Information) and (iv) CACM (3204 documents from the association for computing machinery). For each of these document collections, we computed the LSA, PLSA, and LSA-PLSA representations of both the document collection and the queries for a range of latent classes, or factors.

For each data set, we used the computed representations to estimate the similarity of each query to all the documents in the original collection. For the LSA model, we estimated the similarity using the cosine distance between the reduced dimensional representations of the query and the candidate document. For the PLSA and LSA-PLSA models, we first computed the probability of each word occurring in the document, $p(w|d) = \frac{p(w,d)}{p(d)}$, using Equation 7 and assuming that $p(d)$ is uniform. This gives us a PLSA-smoothed term representation of each document. We then computed the Hellinger similarity (Basu et al., 1997) between the term distributions of the candidate document, $p(w|d)$, and query, $p(w|q)$. In all of the evaluations, the results for the PLSA model were averaged over four different runs to account for the dependence on the initial conditions.

6.2.1 Single Models

In addition to LSA-based initialization of the PLSA model, we also investigated initializing the PLSA model by first running the "k-means" algorithm to cluster the documents into k classes, where k is the number of latent classes and then initializing $p(w|z)$ based on the statistics of word occurrences in each cluster. We iterated over the number of latent classes starting from 10 classes

up to 540 classes in increments of 10 classes.

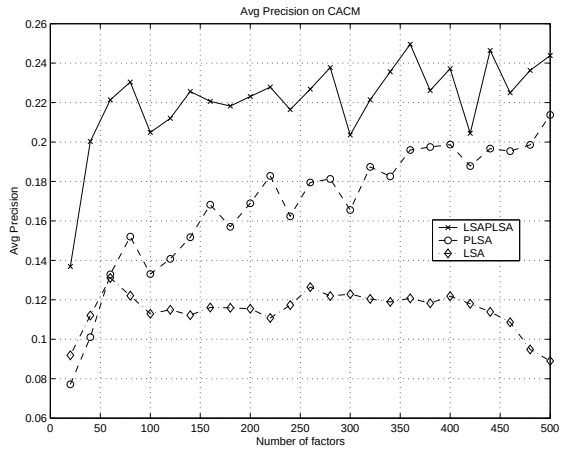

Figure 1: Average Precision on CACM Data set

We evaluated the retrieval results (at the 11 standard recall levels as well as the average precision and break-even precision) using manually tagged relevance. Figure 1 shows the average precision as a function of the number of latent classes for the CACM collection, the largest of the datasets. The LSA-PLSA model performance was better than both the LSA performance and the PLSA performance at all class sizes. This same general trend was observed for the CISI dataset. For the two smallest datasets, the LSA-PLSA model performed better than the randomly-initialized PLSA model at all class sizes; it performed better than the LSA model at the larger classes sizes where the best performance is obtained.

Table 2: Retrieval Evaluation with Single Models. Best performing model for each dataset/metric is in bold.

Data	Met.	LSA	PLSA	LSA-PLSA	kmeans-PLSA
Med	Avg.	**0.55**	0.38	0.52	0.37
Med	Brk.	0.53	0.39	**0.54**	0.39
CISI	Avg.	0.09	0.12	**0.14**	0.12
CISI	Brk.	0.11	0.15	**0.17**	0.15
CACM	Avg.	0.13	0.21	**0.25**	0.19
CACM	Brk.	0.15	0.24	**0.28**	0.22
CRAN	Avg.	0.28	0.30	**0.32**	0.23
CRAN	Brk.	0.28	0.29	**0.31**	0.23

In Table 2 the performance for each model using the optimal number of latent classes is shown. The results show that LSA-PLSA outperforms LSA on 7 out of 8 evaluations. LSA-PLSA outperforms both random and k-means initialization of PLSA in all evaluations. In addition, performance using random initialization was never worse than k-means initialization, which itself is sensitive to initialization values. Thus in the rest of our experiments we initialized PLSA models using the simpler random-initialization instead of k-means initialization.

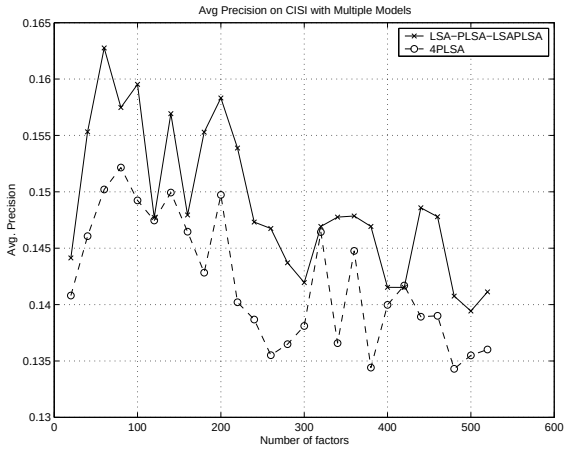

Figure 2: Average Precision on CISI using Multiple Models

6.2.2 Multiple Models

We explored the use of an LSA-PLSA model when averaging the similarity scores from multiple models for ranking in retrieval. We compared a baseline of 4 randomly-initialized PLSA models against 2 averaged models that contain an LSA-PLSA model: 1) 1 LSA, 1 PLSA, and 1 LSA-PLSA model and 2) 1 LSA-PLSA with 3 PLSA models. We also compared these models against the performance of an averaged model without an LSA-PLSA model: 1 LSA and 1 PLSA model. In each case, the PLSA models were randomly initialized. Figure 2 shows the average precision as a function of the number of latent classes for the CISI collection using multiple models. In all class sizes, a combined model that included the LSA-initialized PLSA model had performance that was at least as good as using 4 PLSA models. This was also true for the CRAN dataset. For the other two datasets, the performance of the combined model was always better than the performance of 4 PLSA models when the number of factors was no more than 200-300, the region where the best performance was observed.

Table 3 summarizes the results and gives the best performing model for each task. Comparing Tables 2 and 3, note that the use of multiple mod-

Table 3: Retrieval Evaluation with Multiple Models. Best performing model for each dataset and metric are in bold. L-PLSA corresponds to LSA-PLSA

Data Set	Met	4PLSA	LSA PLSA L-PLSA	LSA PLSA	L-PLSA 3PLSA
Med	Avg	0.55	**0.620**	0.567	0.584
Med	Brk	0.53	**0.575**	0.545	0.561
CISI	Avg	0.152	**0.163**	0.152	0.155
CISI	Brk	0.18	**0.197**	0.187	0.182
CACM	Avg	0.278	**0.279**	0.249	0.276
CACM	Brk	0.299	0.296	0.275	**0.31**
CRAN	Avg	0.377	**0.39**	0.365	**0.39**
CRAN	Brk	0.358	0.368	0.34	**0.37**

els improved retrieval results. Table 3 also indicates that combining 1 LSA, 1 PLSA and 1 LSA-PLSA models outperformed the combination of 4 PLSA models in 7 out of 8 evaluations.

For our data, the time to compute the LSA model is approximately 60% of the time to compute a PLSA model. The running time of the "LSA PLSA LSA-PLSA" model requires computing 1 LSA and 2 PLSA models, in contrast to 4 models for the 4PLSA model, therefore requiring less than 75% of the running time of the 4PLSA model.

6.3 Text Segmentation

A number of researchers, (e.g., Li and Yamanishi (2000); Hearst (1997)), have developed text segmentation systems. Brants et. al. (2002) developed a system for text segmentation based on a PLSA model of similarity. The text is divided into overlapping blocks of sentences and the PLSA representation of the terms in each block, $p(w|b)$, is computed. The similarity between pairs of adjacent blocks b_l, b_r is computed using $p(w|b_l)$ and $p(w|b_r)$ and the Hellinger similarity measure. The positions of the largest local minima, or dips, in the sequence of block pair similarity values are emitted as segmentation points.

We compared the use of different initializations on 500 documents created from Reuters-21578, in a manner similar to Li and Yamanishi (2000). The performance is measured using error probability at the word and sentence level (Beeferman et al., 1997), P_k^w and P_k^s, respectively. This measure allows for close matches in segment boundaries. Specifically, the boundaries must be within k words/sentences, where k is set to be half the average segment length in the test data. In order to account for the random initial values of the PLSA models, we performed the whole set of experiments for each parameter setting four times and averaged the results.

Table 4: *Single Model* Segmentation Word and Sentence Error Rates (%). PLSA error rate at the optimal number of classes in terms of P_k^w is in italic. Best performing model is in bold without italic.

Num Classes	LSA-PLSA		PLSA	
	P_k^w	P_k^s	P_k^w	P_k^s
64	2.14	2.54	3.19	3.51
100	2.31	2.65	2.94	3.35
128	**2.05**	**2.57**	2.73	3.13
140	2.40	2.69	*2.72*	*3.18*
150	2.35	2.73	2.91	3.27
256	2.99	3.56	2.87	3.24
1024	3.72	4.11	3.19	3.51
2048	2.72	2.99	3.23	3.64

6.3.1 Single Models for Segmentation

We compared the segmentation performance using an LSA-PLSA model against the randomly-initialized PLSA models used by Brants et al. (2002). Table 4 presents the performance over different classes sizes for the two models. Comparing performance at the optimum class size for each model, the results in Table 4 show that the LSA-PLSA model outperforms PLSA on both word and sentence error rate.

Table 5: *Multiple Model* Segmentation Word and Sentence Error Rates (%). Performance at the optimal number of classes in terms of P_k^w is in italic. Best performing model is in bold without italic.

Num Class	4PLSA		LSA-PLSA 2PLSA		LSA-PLSA 3PLSA	
	P_k^w	P_k^s	P_k^w	P_k^s	P_k^w	P_k^s
64	2.67	2.93	2.01	2.24	1.59	1.78
100	2.35	2.65	1.59	1.83	**1.37**	**1.62**
128	2.43	2.85	1.99	2.37	1.57	1.88
140	2.04	2.39	*1.66*	*1.90*	1.77	2.07
150	2.41	2.73	1.96	2.21	1.86	2.12
256	2.32	2.62	1.78	1.98	1.82	1.98
1024	*1.85*	*2.25*	2.51	2.95	2.36	2.77
2048	2.88	3.27	2.73	3.06	2.61	2.86

6.3.2 Multiple Models for Segmentation

We explored the use of an LSA-PLSA model when averaging multiple PLSA models to reduce the effect of poor model initialization. In particular, the adjacent block similarity from multiple

models was averaged and used in the dip computations. For simplicity, we fixed the class size of the individual models to be the same for a particular combined model and then computed performance over a range of class sizes. We compared a baseline of four randomly initialized PLSA models against two averaged models that contain an LSA-PLSA model: 1) one LSA-PLSA with two PLSA models and 2) one LSA-PLSA with three PLSA models. The best results were achieved using a combination of PLSA and LSA-PLSA models (see Table 5). And all multiple model combinations performed better than a single model (compare Tables 4 and 5), as expected.

In terms of computational costs, it is less costly to compute one LSA-PLSA model and two PLSA models than to compute four PLSA models. In addition, the LSA-initialized models tend to perform best with a smaller number of latent variables than the number of latent variables needed for the four PLSA model, also reducing the computational cost.

7 Conclusions

We have presented LSA-PLSA, an approach for improving the performance of PLSA by leveraging the best features of PLSA and LSA. Our approach uses LSA to initialize a PLSA model, allowing for arbitrary weighting schemes to be incorporated into a PLSA model while leveraging the optimization used to improve the estimate of the PLSA parameters. We have evaluated the proposed framework on two tasks: personal-size information retrieval and text segmentation. The LSA-PLSA model outperformed PLSA on all tasks. And in all cases, combining PLSA-based models outperformed a single model.

The best performance was obtained with combined models when one of the models was the LSA-PLSA model. When combining multiple PLSA models, the use of LSA-PLSA in combination with either two PLSA models or one PLSA and one LSA model improved performance while reducing the running time over the combination of four or more PLSA models as used by others.

Future areas of investigation include quantifying the expected performance of the LSA-initialized PLSA model by comparing performance to that of the empirically best performing model and examining whether tempered EM could further improve performance.

References

Ayanendranath Basu, Ian R. Harris, and Srabashi Basu. 1997. Minimum distance estimation: The approach using density-based distances. In G. S. Maddala and C. R. Rao, editors, *Handbook of Statistics*, volume 15, pages 21–48. North-Holland.

Doug Beeferman, Adam Berger, and John Lafferty. 1997. Statistical models for text segmentation. *Machine Learning*, (34):177–210.

Thorsten Brants, Francine Chen, and Ioannis Tsochantaridis. 2002. Topic-based document segmentation with probabilistic latent semantic analysis. In *Proceedings of Conference on Information and Knowledge Management*, pages 211–218.

Noah Coccaro and Daniel Jurafsky. 1998. Towards better integration of semantic predictors in statistical language modeling. In *Proceedings of ICSLP-98*, volume 6, pages 2403–2406.

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

Chris H. Q. Ding. 1999. A similarity-based probability model for latent semantic indexing. In *Proceedings of SIGIR-99*, pages 58–65.

Usama M. Fayyad, Cory Reina, and Paul S. Bradley. 1998. Initialization of iterative refinement clustering algorithms. In *Knowledge Discovery and Data Mining*, pages 194–198.

David Guo, Michael Berry, Bryan Thompson, and Sidney Balin. 2003. Knowledge-enhanced latent semantic indexing. *Information Retrieval*, 6(2):225–250.

Marti A. Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.

Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of SIGIR-99*, pages 35–44.

Hang Li and Kenji Yamanishi. 2000. Topic analysis using a finite mixture model. In *Proceedings of Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 35–44.

Michael Tipping and Christopher Bishop. 1999. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B*, 61(3):611–622.

Huiwen Wu and Dimitrios Gunopulos. 2002. Evaluating the utility of statistical phrases and latent semantic indexing for text classification. In *Proceedings of IEEE International Conference on Data Mining*, pages 713–716.