A Latent Semantic Structure Model for Text Classification

Mingwen Wang[1,2], Jian-Yun Nie[1]

[1] DIRO, Université de Montréal
C.P. 6128, succursale Centre-ville, Montreal
Quebec, H3C 3J7 Canada
{wangming, nie}@iro.umontreal.ca

[2] School of Computer Science and Technology
Jiangxi Normal University
330027, Nanchang, Jiangxi, China
mwwang@jxnu.edu.cn

Abstract Latent Semantic Indexing (LSI) has been successfully applied to information retrieval and classification. LSI can deal with the problems of polysemy and synonymy, and can reduce noise in the raw document-term matrix. However, LSI may ignore important features for some small categories because they are not the most important features for all the document collection.

In this paper, we describe a new approach which extends LSI by incorporating also the classification information of the training documents. In our model, we consider two matrices: document-term and document-class. This model may better capture the latent semantic structure behind the classification examples than LSI.

1. Introduction

Text classification (categorization) aims to assign test to one or more predefined classes based on their content. A number of statistical and machine learning techniques has been developed for text classification, including regression model, k-nearest neighbor, decision tree, Naïve Bayes, support vector machines, and so on (Sebastiani, 2002). However, many current classifiers are based on keyword (that are usually single words) extracted from the documents. In many of the classification approaches, a keyword is assumed to be a unique representative of a distinctive concept or semantic unit. However, the reality is different: A word may represent several different meanings, and different words may refer to the same meaning. These are the problems of polysemy and synonymy. For example, the word *bank* can be a section of computer memory, a financial institution, a steep slope, a collection of some sort, an airplane maneuver, or even a billiard shot. It is hard to distinguish those meanings automatically. Similarly, in medical literature, *myocardial infarction* has the same meaning as *minor heart attack*.

Polysemy and synonymy are two main problems that any conceptual indexing scheme must cope with. These are the very problems that motivated the utilization of Latent Semantic Indexing (LSI) in IR (Deerwester et al., 1990). LSI tries to transform the original vector space to a latent semantic space. Documents and queries are represented and compared in this new space. The experiments have shown that this method allows dealing with the problems of polysemy and synonymy to some extent: the newly obtained dimensions correspond to linear combinations of the original dimensions in the document-term matrix. It is believed that they can capture "latent" semantic structure of the vocabulary used in the corpus.

Since the first application of LSI to IR in (Deerwester et al., 1990), LSI has been applied to other tasks such as text classification. The work of (Wiener et al., 1995) is very representative. Wiener et al. used LSI for classification in two different ways. Their first method uses LSI to reduce dimensionality of the original vector space: by cutting the least weighted dimensions in the latent semantic space, they can keep only a subset of the dimensions and remove a part of the noise. The second way consists of separating documents into different categories, and an LSI representation is created for each category. Their experiments showed the second approach performs better than the first one.

One of the reasons that motivated the use of separate LSI representations for different categories is that some terms may be very important in a particular category, while when all the categories are considered together, they may become unimportant. This is the case for ambiguous terms: the term *bank* may be important in the category of *finance*. However, if all the categories are merged, this term may be ignored (or becomes less important) in the latent semantic space because of its ambiguity (i.e. it is not helpful to determine the correct category of a text).

We observe that while the creation of separate LSI representations for different categories may correctly assign a degree of importance to terms (or features) within each category, it is difficult to consider them facing with a set of categories. Indeed, while a term is important in some particular categories, it may be

useless for determining to which category a document should belong to because of its ambiguity. The key problem is to put all the categories together, so that a global competition takes place to select the most appropriate category for a document. This competition cannot be made on a fair basis if the categories have been considered completely separately.

In this paper, we propose an extend LSI method. This method extracts the semantic features reflecting both the structure between the terms and the document classification information. Instead of creating several LSI representations for different categories, we create only one LSI representation which integrates the classification structure.

This paper is organized as follows. In the section 2, we will describe some basic notions and related work. In section 3, our method will be described, and compared with the existing approaches. Finally, some conclusions will be given in section 4.

2. Related work

2.1. Classification Using Vector Space Model

In vector space model, a document is represented by a vector in a linear space of *n* words or terms (word vector space). These words or terms are indexes extracted from the documents. Similarly, a word is represented by a vector in a linear space of *m* documents. The documents form a document vector space. These dual representation are best captured by the document-term association matrix X, where each column $X_{\bullet j}$ represents a word (term), and each row $X_{i\bullet}$ represents a document:

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix} = (X_{\bullet 1}, X_{\bullet 2}, \cdots, X_{\bullet n}) = \begin{pmatrix} X_{1\bullet} \\ X_{2\bullet} \\ \vdots \\ X_{m\bullet} \end{pmatrix}$$

The matrix element x_{ij} contains the term frequency (*tf*) of term *j* occurring in the document *i*, properly weighted by other factors (Salton and Buckly, 1988), for example, by the inverse document frequency (*idf*):

$$x_{ij} = tf_{ij} * \log(m / df_j)$$

where the document frequency df_j is the number of documents the word (term) *j* occurs in. Note that most matrix elements of X are zero because an index word usually occurs in only a few documents.

Information retrieval on the document collection is typically handled by keywords matching. A user query **q,** consisting of a set of keywords (terms), is treated as a document. The keyword matching is equivalent to a dot product between the query vector and a document vector (variable document length is accounted for by normalizing document vectors to unit length). Relevance scores for all the *m* documents form a row vector **s**, which is determined as follows:

$$\mathbf{s} = \mathrm{X}\mathbf{q}^{\mathrm{T}}$$

Documents are then sorted according to their relevance score and returned to the user.

Text categorization tries to classify an incoming news item or email into predefined categories. This may be done in a number of ways. A simple method is to calculate a centroid vector $\mathbf{c}_i$ (i.e., the average of all documents in the category) of each category *i* (Dumais,1995). The centroid of a category is considered as the representation of the category. All the *k* centroid vectors for *k* categories form a $k \times n$ matrix C=($\mathbf{c_1}$, $\mathbf{c_2}$,…, $\mathbf{c_k}$)^T. The category chosen for a document is the one which has a centroid the most similar to the document's vector.

Another method is to solve the problem by determining the C that represents the least square of classification errors (Yang,1995), that is:

$$C = \arg\min_C \| CX^T - B \|$$

where B is a $k \times m$ matrix which defines correct categories for each document in the training set. An incoming document is then projected onto these mapping vectors to get similarity scores for each category with appropriate thresholding. Note that a document may belong to several categories.

2.2. Latent Semantic Indexing for classification

In the initial vector space, the word-to-document relations contain redundancy, ambiguity, and noise. We want to create a subspace that contains only meaningful semantic associations and is much smaller than the initial space. One method to achieve this is to perform a dimensionality reduction so as to select a semantic subspace that contains essential and meaningful associative relations. In the subspace, redundant information can be grouped together, ambiguity can be coped with by combining with other contextual information, and noise can be partly removed by trimming the least important dimensions of the subspace.

LSI is one such dimension reduction method. It automatically computes a subspace containing meaningful semantic associations which is much smaller than the initial space. This is done through the singular value decomposition (SVD) (Golub and Van Loan, 1989) of the term-document matrix:

$$\mathbf{X} = \sum_{i=1}^{r} \mathbf{u}_i \sigma_i \mathbf{v}_i = (\mathbf{u}_1, \cdots, \mathbf{u}_r) \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{pmatrix} \begin{pmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_r \end{pmatrix}$$

where r is the rank of the matrix X, $\sum_r \equiv diag(\sigma_1 \quad \cdots \quad \sigma_r)$ is a diagonal matrix of singular values, $\mathbf{U_r} = (\mathbf{u}_1, \cdots, \mathbf{u}_r)$ and $\mathbf{V_r} = (\mathbf{v}_1, \cdots, \mathbf{v}_r)$ are left and right singular vectors. Typically the rank r is in the order of min(n, m). However, if we truncate the singular values, i.e., keep only the first k largest terms, the following resulting matrix is a good approximation:

$$\mathbf{X} = \mathbf{U}_r \Sigma_r \mathbf{V}_r^{\mathbf{T}} \approx \mathbf{U}_k \Sigma_k \mathbf{V}_k^{\mathbf{T}}$$

In an LSI k-dimension subspace, a document $X_{i\bullet}$ is represented as its projection to the subspace, i.e. $\mathbf{X}_{i\bullet}\mathbf{V}_k$, and all the m documents are projected to $\mathbf{X}\mathbf{V}_k = \mathbf{U}_k \Sigma_k$. Queries are transformed in the same way as documents. Therefore, in text retrieval, the score vector for a query **q** in LSI subspace is evaluated as

$$\mathbf{s} = (\mathbf{X}\mathbf{V}_k)(\mathbf{q}\mathbf{V}_k)^{\mathbf{T}} = (\mathbf{U}_k \Sigma_k)(\mathbf{V}_k^{\mathbf{T}} \mathbf{q}^{\mathbf{T}}).$$

The greatly reduced dimensionality can reduce both the complexity and noise in text categorization and retrieval. For example, the centroid matrix or mapping matrix is reduced from $m \times n$ to $m \times k$ (Dumais,1995). The same dimensionality reduction has also proven to be effective in Naïve Bayes categorization (Baker and McCallum, 1998), KNN classification and Support Vector Machine classification (Park et al., 2003). All these lead to more accurate categorization results.

The success of LSI is attributed to the fact that the LSI subspace captures the essential associative semantic relationships, better than the original document space, and thus partially resolves the word choice (synonyms) problem in information retrieval, and redundant semantic relationships in text classification.

Mathematically, LSI with a truncated SVD is the best approximation of X in the reduced k-dimension subspace. From a statistical point of view, LSI amounts to an effective dimensionality reduction, similar to principal component analysis in statistics. Dimensions with small singular values are often viewed as representing semantic noises and thus are ignored.

However, LSI also ignores the class structure while reducing the dimensionality. In fact, when a global LSI representation is created for all the documents, the classification information is usually not considered. The LSI representation is the one that contain the most important global principal components. One has to notice that some components may be important in some particular categories, while be marginal in the whole corpus. These components are likely to be removed in the dimension reduction step. As a consequence, the algorithm may ignore these components when trying to classify a document into these particular categories. To solve this problem, the classification information should be taken into account when the LSI representation is created.

Another problem with the traditional utilization of LSI in IR and dimensionality reduction is that there is no theoretical optimum value for the number of the dimensions to be kept. A large amount of experiments may be required to determine it. Classification results with LSI may vary depending upon the reduced dimensions (Park et al., 2003).

3. A latent semantic structure model for classification

The LSI method is based on the assumption that there is some underlying latent semantic structure in the term-document matrix that is corrupted by the wide variety of words used in documents. One of the problems using LSI for classification is that important characteristics of some small classes may be ignored. In fact, infrequent classes are usually characterized by infrequent terms (in comparison with the other terms in the whole corpus), and infrequent terms may be projected out of LSI's representations which utilize only a fraction of the original dimensions. Thus some terms which are important for discriminating between two classes of documents may be considered to be noise. Let us consider the following simple example to illustrate how an important feature of a category may be cut out.

Suppose that the documents collection include 3 documents $\{d_1, d_2, d_3\}$. The words occur in the collection are $\{t_1, t_2, t_3\}$, and documents $\{d_1, d_2\}$ belong to class 1, $\{d_3\}$ to class 2. The document-word matrix is as follows:

$$\mathbf{X} = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Through the singular value decomposition, we obtain

$$\mathbf{X} = \begin{pmatrix} 1/\sqrt{2} & 0 & 0 \\ 1/\sqrt{2} & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \sqrt{2} & & \\ & 1 & \\ & & 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

$$= \begin{pmatrix} 1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \sqrt{2} & \\ & 1 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Now if we cut the second singular value from the diagonal matrix, then the resulting LSI representation will only reflect the characteristic of one class. Class 2 will be completely ignored.

This simple example shows a rather drastic case. In the dimensionality reduction used in LSI, cut is less drastic. However, the problem is still present, even it is to a lesser degree. The essential point is that some important characteristics of small categories may be cut off because they are not considered to be strong characteristics for the whole document corpus.

To solve this problem, Wiener et al. (Wiener et al., 1995) use local LSI instead of global LSI. They create category-specific LSI representations, each for a category. The local LSI representations of categories are compared separately with an incoming document for the classification of the latter. This method can partly solve the above problem: the important features of each category can be kept in the local LSI representations. However, there are other problems in this method:

1. The incoming document is compared with each local LSI representation separately. As each local LSI has been created separately, the resulting similarities with the local LSI representations are hardly comparable. It may be the case that the document is considered more similar to one category, while it should be classified in another category.
2. This method cannot cope with ambiguous words correctly. In fact, within one particular category (e.g. finance), an ambiguous word (e.g. bank) may become unambiguous. The resulting LSI representation may attribute a strong importance to the word. However, if we consider such words in a global context

of classification, we should not attribute a high importance to it because its ambiguity may lead to wrong classes. By separating the LSI representations for each category, we cannot consider the global ambiguity of words in the LSI representations.

In order to solve these problems, we propose a method that considers the document-word matrix and document-class matrix at the same time. The method is similar to LSI, based on the projection of the words (terms) to the latent semantic space. However, our method creates the latent semantic relationship between words and classes by modeling the relationship between them while maintaining most of the information in the words (terms). From a statistical point of view, it is similar to Partial Least Square Analysis.

In what follows, **X** will represent the $m \times n$ document-word matrix as before and **Y** will stand for the $m \times r$ document-class matrix as follows:

$$\mathbf{Y} = \begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1r} \\ y_{21} & y_{22} & \cdots & y_{2r} \\ \vdots & \vdots & \ddots & \vdots \\ y_{m1} & y_{m2} & \cdots & y_{mr} \end{pmatrix}$$

where $y_{ij} = \begin{cases} 1 & \text{doument } i \text{ belong to class } j \\ 0 & \text{otherwise} \end{cases}$

Both **X** and **Y** are assumed to be centered. We can use the following diagram to illustrate the latent semantic relationship between **X** and **Y.**

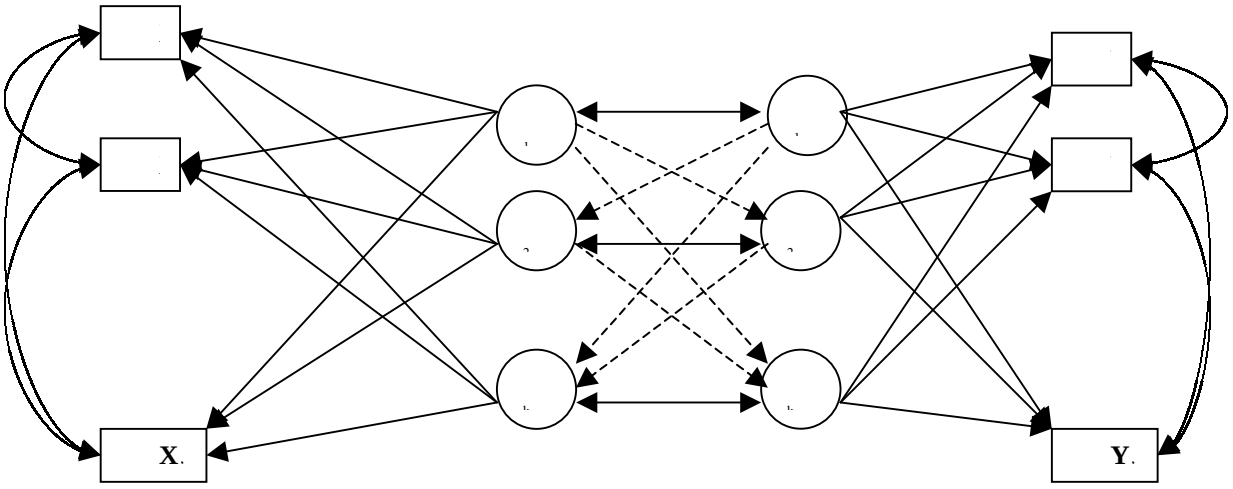

Figure 1. diagram for latent semantic relationship

In figure 1, enclosed in cycles are latent variables ξ and ω, enclosed in rectangles are term ($\mathbf{X_{\bullet i}}$) and class ($\mathbf{Y_{\bullet i}}$) variables. The double arrow between ξ and ω indicates a non-zero correlation. The single arrow from the latent variables to the terms/classes indicates that there are non-zero coefficients for the latent variables in the equations for the term/class. The lack of an arrow or edge between variables, e.g. between ξ and $\mathbf{Y_{\bullet i}}$, means that any dependence between them can occur only through other variables. The double-headed arrow between the term variables means that they are not conditionally independent given ξ. Between the class variables, the links are similar.

The dotted arrows mean that the variables are indeed dependent, but they are considered implicitly. In fact, as we will see later, when we determine the lower level variables ξ_i and ω_i, we only consider the remainder information that has not been captured by the higher-level ξ's and ω's. This is what implicit dependency means.

Given the diagram, we are now interested, not in the covariance matrix **X** of the term variables, but in the cross-covariance of **X** and **Y**. And we wish to model the cross-covariance by pair of latent variables, i.e. (ξ_1, ω_1), (ξ_2, ω_2), ..., (ξ_k, ω_k). ξ_i represents the latent semantic information of **X**, and ω_i represents the latent semantic information of **Y**. (ξ_i, ω_i) are in the decreasing order of their importance to the **X** and **Y** matrices. That is, (ξ_1, ω_1) captures the most important features, (ξ_2, ω_2) the next most important features, and so on.

The principle of determining these variables is as follows:

- (ξ_1, ω_1) is the pair of variables that approximates the best **X** and **Y**;
- (ξ_2, ω_2) is the pair of variables that approximates the best the remainder of **X** and **Y**, i.e. the parts that have not been captured by (ξ_1, ω_1);
- (ξ_3, ω_3) and other pairs are determined successively in the same way to best approximate the remainder of the previous variable pairs.

Let us explain the determination of these variables.

For the first pair of latent variable (ξ_1, ω_1), they have to meet the following requirements:

(a). ξ_1 must represent the latent semantic information of **X** as best as possible;

(b). ω_1 must represent the latent semantic information of **Y** as best as possible.

(c). (ξ_1, ω_1) must represent as best as possible the correlation between **X** and **Y**;

From a statistical point of view, the condition (a) is equivalent to a variable that has the maximal variance, i.e. Var(ξ_1)→max. Recall that a latent variable can represent the most information of a data matrix if and only if the variance is maximal. The condition (b) is equivalent to Var(ω_1)→max. The condition (c) is equivalent to r(ξ_1, ω_1)→max, where $r(\bullet,\bullet)$ represent the correlation coefficient between two stochastic variables.

If we consider ξ_1 as a term component, it may be expressed as follows:

$$\xi_1 = \mathbf{Xu}$$

where **u** is a vector to be determined. This also means that ξ_1 is a linear combination of some terms in the documents that corresponds to an important semantic unit.

In the same way, we can also express the variable $\omega_1 = \mathbf{Yv}$ (where **v** is also a vector to be determined) as a linear combination of elements from **Y**. It is an intermediate variable connecting the content information (terms of documents) and the classification information.

Then the above three conditions can be translated to:

$$\max(\mathrm{Var}(\xi_1)) = \max_{\|\mathbf{u}\|=1} (\mathrm{Var}(\mathbf{Xu}))$$

$$\max(\mathrm{Var}(\omega_1)) = \max_{\|\mathbf{v}\|=1} (\mathrm{Var}(\mathbf{Yv}))$$

$$\max(\mathrm{r}(\xi_1, \omega_1)) = \max_{\|\mathbf{u}\|=\|\mathbf{v}\|=1} r(\mathbf{Xu}, \mathbf{Yv})$$

where ||**u**|| and ||**v**|| represents the lengths of vectors **u** and **v**.

According to the definition of covariance, we have:

$$\mathrm{Cov}(\xi_1, \omega_1) = \sqrt{Var(\xi_1)Var(\omega_1)} \times r(\xi_1, \omega_1)$$

Therefore, the above three maximization problems can be integrated into one maximization problem (Wold, 1985):

$$\mathrm{Cov}(\xi_1, \omega_1) \rightarrow \max;$$

Let $< \bullet,\bullet >$ represent dot product. Because $<\xi_1, \omega_1> = \mathrm{Cov}(\xi_1, \omega_1)$, the problem of determining (ξ_1, ω_1) can be translated into the following maximization problem:

$$<\xi_1, \omega_1> = \xi_1^{\mathbf{T}}\omega_1 = \max_{\|\mathbf{u}\|=\|\mathbf{v}\|=1} < \mathbf{Xu}, \mathbf{Yv} >$$

Let $\mathbf{u}_1$ and $\mathbf{v}_1$ be the solution vectors of the above maximize problem. Then by the well-known property of singular value decomposition, we know that $d_1\mathbf{u}_1\mathbf{v}_1^{\mathbf{T}}$ is the best one-dimension approximation of $\mathbf{X}^{\mathbf{T}}\mathbf{Y}$ in the least square sense (Harvill, 1997) (where d_1 is the largest singular value, $\mathbf{u}_1$ and $\mathbf{v}_1$ are the left and right singular vectors).

Once the first pair of latent variables (ξ_1, ω_1) has been exacted, we can obtain the information represented by these variables by regressing the matrix **X** on ξ_1 and **Y** on ω_1 as follows:

$$\hat{\mathbf{X}}_1 = \xi_1(\xi_1^{\mathbf{T}}\xi_1)^{-1}\xi_1^{\mathbf{T}}\mathbf{X}$$
$$\hat{\mathbf{Y}}_1 = \omega_1(\omega_1^{\mathbf{T}}\omega_1)^{-1}\omega_1^{\mathbf{T}}\mathbf{Y}$$

Then the remaining information corresponds to $\mathbf{X} = \mathbf{X} - \hat{\mathbf{X}}$ and $\mathbf{Y} = \mathbf{Y} - \hat{\mathbf{Y}}$.

From $\hat{\mathbf{X}}_1$ and $\hat{\mathbf{Y}}_1$, we go through the same process to determine (ξ_2, ω_2) and other pairs of variables in turn.

The above procedure is the similar to the Partial Least Square (PLS) algorithm proposed by Wold (Wold, 1985). This algorithm has been modified by (Tenenhaus, 1998) in order to avoid the computing of SVD. The modified algorithm is as follows:

ALGORITHM-1(LSR/PLS2):

$\mathbf{E}_0 = \mathbf{X}$; $\mathbf{F}_0 = \mathbf{Y}$;
FOR k=1,…,s DO
$\mathbf{u}_k$= first column of $\mathbf{F}_{k-1}$;
DO until convergence in ξ_k //computing pairs of latent variables
$\xi_k = \mathbf{E}_{k-1}^{\mathbf{T}}\mathbf{u}_k / \mathbf{u}_k^{\mathbf{T}}\mathbf{u}_k$;
$\xi_k = \frac{\xi_k}{\| \xi_k \|}$;
$\mathbf{t}_k = \mathbf{E}_{k-1}\xi_k$;
$\omega_k = \mathbf{F}_{k-1}^{\mathbf{T}}\mathbf{t}_k / \mathbf{t}_k^{\mathbf{T}}\mathbf{t}_k$
$\mathbf{u}_k = \mathbf{F}_{k-1}\omega_k / \omega_k^{\mathbf{T}}\omega_k$;
ENDDO
$\mathbf{p}_k = \mathbf{E}_{k-1}^{\mathbf{T}}\mathbf{t}_k / \mathbf{t}_k^{\mathbf{T}}\mathbf{t}_k$;
$\mathbf{E}_k = \mathbf{E}_{k-1} - \mathbf{t}_k\mathbf{p}_k^{\mathbf{T}}$; //remaining information of document-term matrix
$\mathbf{F}_k = \mathbf{F}_{k-1} - \mathbf{t}_k\omega_k^{\mathbf{T}}$ //remaining information of document-class matrix
ENDFOR

In this algorithm, the number *s* can be determined by cross-validation (Helland, 2002).

Once these variables have been determined, one can use the latent semantic structure to classify documents as follows. Suppose a new document $\mathbf{d}_0 = (x_{01}, x_{02}, \cdots, x_{0n})$ arrives. Let

$\mathbf{e}_0 = (x_{01} - \bar{x}_1, x_{02} - \bar{x}_2, \cdots, x_{0n} - \bar{x}_n)$, where $\bar{\mathbf{x}}_i = \frac{1}{m}\sum_{k=1}^{m} x_{ki}$, be the vector representing the difference between the document and the centroid vector of all the training documents. Then we can calculate the following elements for $k = 1, \ldots, s$;

$$\mathbf{t}_{k0} = \mathbf{e}_{k-1}\xi_k$$
$$\mathbf{e}_k = \mathbf{e}_{k-1} - \mathbf{t}_{k0}\mathbf{p}_k$$

$\mathbf{t}_{k0}$ is an intermediate variable. $\mathbf{p}_k$ is the normalized principal component determined at step k (see the algorithm). $\mathbf{t}_{k0}\mathbf{p}_k$ represents the semantic information captured by the k-th principal component. Therefore, the element $\mathbf{e}_k$ represents the remainder of the document at step k.

Finally, the classification of the document can be calculated as follows:

$$\begin{aligned}\hat{\mathbf{y}} &= \bar{\mathbf{y}} + \sum_{k=1}^{s} t_{k0}\omega_k^{\mathbf{T}} \\ &= \bar{\mathbf{y}} + \sum_{k=1}^{s} t_{k0}(\mathbf{F}_{k-1}^{\mathbf{T}}\mathbf{t}_k / \mathbf{t}_k^{\mathbf{T}}\mathbf{t}_k)^{\mathbf{T}} \\ &= \bar{\mathbf{y}} + \sum_{k=1}^{s} t_{k0}(\mathbf{F}_{k-1}^{\mathbf{T}}\mathbf{E}_{k-1}\xi_k / (\mathbf{E}_{k-1}\xi_k)^{\mathbf{T}}\mathbf{E}_{k-1}\xi_k)^{\mathbf{T}}\end{aligned}$$

where $\bar{\mathbf{y}}$ is determined in the same way as $\bar{\mathbf{x}}$. The vector $\hat{\mathbf{y}}$ stores the similarities of the new document to each class. We notice that this formula contains both the contents of the documents and the classification information (**F**).

Notice that our approach take into account both the contents of the documents (**E** in the above formula) and the classification information of the documents (**F** in the above formula). In fact, the former is captured by variables ξ_k, and the latter by variables ω_k. The element $t_{k0}\omega_k^{\mathbf{T}}$ in the above expression combines both of them.

In comparison with the approach based on global LSI, our approach adds the classification information. In comparison with the approach based on local LSIs, our approach does not consider the classes separately. This has the advantage that the degree of classification to each class can be easily compared.

The algorithm ELSI/PLS2 is relatively slow because of the complexity of the computation of ξ_k. This complexity is required because **Y** is a matrix that stores the classification information, whereby a document can be classified into several classes. If we assume that each document is classified into only one class, then the matrix **Y** can be simplified to a vector $\mathbf{y} = (y_1, y_2, \cdots, y_m)^{\mathbf{T}}$, where $y_k = i$, if document k belong to class i. In the case, the algorithm can be simplified to the following one (ELSI/PLS1).

ALGORITHM-1 (LSR/PLS1):

$\mathbf{E}_0 = \mathbf{X}$; $\mathbf{f}_0 = \mathbf{y}$;
FOR k=1,…,s DO
$\xi_k = \mathbf{E}_{k-1}^{\mathbf{T}}\mathbf{f}_{k-1}$;
$\mathbf{t}_k = \mathbf{E}_{k-1}\xi_k$;
$\mathbf{p}_k = \mathbf{E}_{k-1}^{\mathbf{T}}\mathbf{t}_k / \mathbf{t}_k^{\mathbf{T}}\mathbf{t}_k$;
$\omega_k = \mathbf{f}_{k-1}^{\mathbf{T}}\mathbf{t}_k / \mathbf{t}_k^{\mathbf{T}}\mathbf{t}_k$
$\mathbf{E}_k = \mathbf{E}_{k-1} - \mathbf{t}_k\mathbf{p}_k^{\mathbf{T}}$;

$$\mathbf{f}_k = \mathbf{f}_{k-1} - \mathbf{t}_k \omega_k^{\mathrm{T}};$$

ENDFOR

Although we assumed that the ELSI/PLS1 algorithm deals with single-class classification problem, it can be extended to deal with the multiple-class classification in the following way: If a document belongs to several classes, we extend the document-word matrix and document-class matrix by creating multiple rows for the same document in **X** and **y**, specifying each of the classes for the document. For example, suppose the document $\mathbf{d}_k = (x_{k1}, x_{k2}, \cdots, x_{kn})$ corresponds to two classes *i* and *j*. Then we can form the following matrices:

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ \vdots & \vdots & \vdots & \vdots \\ x_{k1} & x_{k2} & \cdots & x_{kn} \\ x_{k1} & x_{k2} & \cdots & x_{kn} \\ \vdots & \vdots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix} \quad \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ i \\ j \\ \vdots \\ y_m \end{pmatrix}$$

where the document corresponds to two rows. In this way, we can cope with the multiple-class situation.

It is interesting to note that when the vector **y** is uniformly 1, the above model becomes the traditional LSI approach. This means that the latter is a particular case of our model.

4. Concluding remarks

In this paper we have described the framework and algorithms for a new latent semantic structure model for text classification. Instead of using only the document-word matrix as in LSI, we also use the document-class matrix in our approach. The addition of this latter matrix makes it necessary to determine pairs of latent variables (ξ_i, ω_i) instead of single singular values. These pairs of variables are intended to capture not only the matrices **X** and **Y** separately, but also the relationships between them.

One particular case is that all the elements in **Y** are equal. In this case, our model is reduced to the classical LSI.

We proposed two algorithms LSR/PLS2 and LSR/PLS1 to determine the latent semantic structure from a set of training documents. These algorithms are linear. However, we can easy extend them to nonlinear algorithms by transforming **X** into $\Phi(\mathbf{X})$, where $\Phi(\bullet)$ is a nonlinear function.

The model described in this paper only highlight the basic idea of our approach. This idea should be validated in the future with experiments.

References

Baker, L.D., McCallum, A.K. (1998) Distributional clustering of words for text classification, *Proc. ACM-SIGIR-98*, pp. 96—103.

Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R. (1990) Indexing by Latent Semantic Analysis, *Journal of the American Society of Information Science*, 41(6): 391-407.

Dumais, S.T. (1995) Using LSI for information filtering, *The Third Text Retrieval Conference (TREC-3),* D. Harman, Ed., National Institute of Standards and Technology Special Publication.

Golub, G. and Loan, C. V. (1989) *Matrix computation*, Johns-Hopking, Baltimore, 2nd ed.

Harville, D.A. (1997) *Matrix algebra from a statistician's perspective*. Spinger.

Helland, L.S. (2002) Partial least squares regression*, Encyclopedia of statistical science* 2.

Park, H., Howland, P. and Jeon, M (2003) Cluster structure preserving dimension reduction based on the generalized singular value decomposition, *SIAM Journal on Matrix Analysis and Applications*, 25(1):165-179.

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

Sebastiani, F. (2002) Machine learning in automated text categorization, *ACM Computing Surveys,* 34(1):1-47.

Tenenhaus, M. (1998) *La Régreesion PLS. Théorie et Pratique*. Éditions Technip, Paris.

Yang, Y. (1995) Noise, Reduction in a Statistical Approach to Text Categorization, *18th ACM International Conference on Research and Development in Information Retrieval*, pp. 256—263.

Wiener, E., Pedersen, J.O., Weigend, A.S. (1995) A Neural Network Approach to Topic Spotting, *Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval*, pp. 317—332.

Wold, H. (1985) Partial least squares. In *Kotz, S. and Johnson N.L., Encyclopedia of Statistical Science.* Wiley, New York.