5th IEEE Int. Conf. on Data Mining (ICDM'05), Houston, TX, 2005.

Hierarchy-Regularized Latent Semantic Indexing

Yi Huang[1], Kai Yu[2], Matthias Schubert[1], Shipeng Yu[1], Hans-Peter Kriegel[1]
[1]Institute for Computer Science, University of Munich
[2]Siemens Corporate Technology
huang@cip.ifi.lmu.de {kai.yu}@siemens.com
{schubert,spyu,kriegel}@dbs.ifi.lmu.de

Abstract

Organizing textual documents into a hierarchical taxonomy is a common practice in knowledge management. The given class hierarchy does not only express the similarity *between the classes, but can also be interpreted as a model of the* specificity *of the classes. In this paper, we propose a hierarchy-regularized framework, which encodes the hierarchy into a similarity graph of documents and then formulates an optimization problem mapping each node of the graph into a low dimensional vector space. The new feature space preserves the intrinsic structure of the original taxonomy and thus provides a meaningful basis for various learning tasks like visualization, regression and classification. Our approach employs the information about class proximity and class specificity, and can naturally cope with multi-labeled documents. Our empirical studies show very encouraging results on two real-world data sets, the new Reuters (RCV1) benchmark and the Swissprot protein database.*

1 Introduction

A characteristic of textual documents is the high dimensionality (typically tens of thousands). Thus, dimensionality reduction plays an important role for reducing computational costs and for improving the performance of the employed text mining algorithms. Another typical characteristic of text applications is that often a document is allowed to belong to more than one class, i.e. the documents are multi-labeled. For example, a news article about a football team could belong to both categories "sports" and "business". Large text databases usually contain large amounts of classes. To allow easy navigation and express the inheritance relationships between these classes, the classes are often organized in a class hierarchy or taxonomy.

The taxonomy is an intrinsic structure of categories and documents. Each node in a class hierarchy represents a subclass of the father node. The leaf nodes describe basic classes that are not distinguishable any further, while the root corresponds to the most general class, comprising all documents. An example is the large topic hierarchy of a web directory service like Yahoo! that allows us to navigate to any category among several thousands topics by just a few clicks. Additional examples for large topic trees are the library of congress catalogue or biological class systems like Gene Ontology [5].

In this paper, we describe a novel approach to exploit a given class hierarchy for text indexing. The idea is to directly integrate the information that is contained in the class hierarchy, into a new highly descriptive feature space. We interpret the classes in the hierarchy as "bridges" connecting the documents and introduce a new hierarchy-regularized framework. Our method naturally incorporates the *similarity* between the classes into feature transformation. Thus, large distances between the objects belonging to very similar classes are penalized, while large distances between the objects belonging to dissimilar classes are encouraged. Furthermore, the generated output space considers the *specificity* of classes, i.e. very general classes are considered to be less informative than very specific classes. Thus, classes that are close to the leaf nodes and classes being characterized by a small number of documents play a more important role. Our method employs both the *similarity* and the *specificity* of the classes, an aspect that has not been sufficiently addressed by previous methods. Finally, our method handles multi-labeled documents in a natural way, while other approaches (e.g. see [14]) often need to involve constraints having combinatoric complexity.

Our proposed hierarchy-regularized framework is used to develop a novel textual feature reduction technique, called hierarchy-regularized latent semantic indexing (HLSI). The resulting feature space offers the possibility to integrate a class hierarchy into a variety of text mining and retrieval tasks. Furthermore, it increases the efficiency of these techniques due to the smaller dimensionality of the output space. Our experimental evaluation demon-

strates that the proposed framework is capable to derive low dimensional and highly descriptive feature spaces that allow fast and accurate classification on two real-world data sets. The first is the Reuters Corpus Volume 1 benchmark (RCV1) and the second is the Swissprot [2] protein database.

The rest of the paper is organized as follows: Section 2 describes our hierarchy-regularized framework. In section 3, we briefly survey related work in the area of feature reduction and hierarchical classification of multi-labeled objects. Section 4 presents our experimental results on two real-world data sets and section 5 conclude the paper.

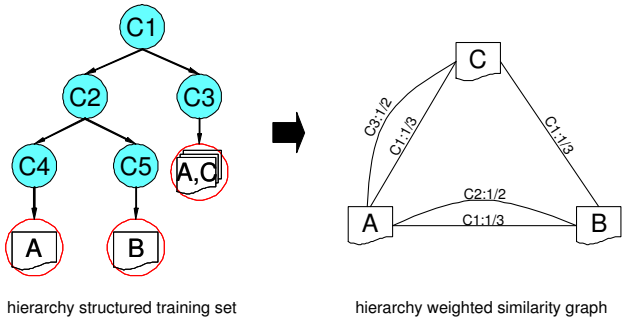


Figure 1. A hierarchy structured training set and the corresponding hierarchy-induced similarity graph.

2 Hierarchy-Regularized Approach

2.1 Training Data

The usual setting of supervised algorithms in data mining and machine learning is: Given a set of predefined classes $\mathcal{C} = \{c_1, \ldots, c_l\}$ and a set of observed *labeled examples* (e.g. documents) $\{(\mathbf{x}_i, y_i)\}$, where $\mathbf{x}_i \in \mathcal{X}$ is an n-dimensional feature vector and $y_i \in \mathcal{C}$ the class label, we want to learn a function that can predict the labels for new test examples. In hierarchical classification, the training data does not simply consist of "feature-label" pairs, but also a hierarchical structure of classes, which offers some additional information about the characteristics of data. Formally, we define the training data in the following way.

Definition 2.1. *A hierarchy structured training set $\mathcal{T}(\mathbf{X}, \mathcal{C}, g, s)$ consists of (1) a set of N* labeled examples *$\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\intercal$, where $\mathbf{x}_i \in \mathbb{R}^n$; (2) a set of* classes *$\mathcal{C} = \{c_1, \ldots, c_l\}$; (3) a function $g : \mathcal{X} \times \mathcal{C} \to \{-1, 1\}$ with $g(\mathbf{x}_i, c_k) = 1$ if $\mathbf{x}_i$ belongs to c_k and -1 otherwise; (4) a function $s : \mathcal{C} \backslash \{c_r\} \to \mathcal{C}$, such that $s(c_k)$ gives the direct father class of c_k, where c_r is the root class.*

Because the classes are organized in a tree structure, the following condition should be fulfilled: $c_k \neq s(c_k)$, and $g(\mathbf{x}_i, c_k) = 1 \Rightarrow g(\mathbf{x}_i, s(c_k)) = 1$. Note that the definition allows the multi-label case where one document can be assigned to multiple leaf classes.

In addition, we define some operators on the tree: (1) $h(\mathbf{x}_i)$ returns the corresponding classes as well as all of their ancestors containing $\mathbf{x}_i$; (2) $H(\mathbf{x}_i, \mathbf{x}_j) = h(\mathbf{x}_i) \cap h(\mathbf{x}_j)$ returns the common classes of $\mathbf{x}_i$ and $\mathbf{x}_j$; (3) $|c_k|$ is the number of examples in class c_k.

2.2 Hierarchy-Induced Similarity Graph

A class hierarchy is not just a notion of class proximity, but also a way to describe the similarity between the examples (e.g., a document). A class is like a "bridge" connecting all the examples within this class. Therefore, we define a *hierarchy-induced similarity graph* as follows:

Definition 2.2. *A hierarchy-induced similarity graph $\mathcal{G}|_{\mathcal{T}}(V, E)$ consists of (1) a set V of* vertices *with a bijective function to $\mathbf{X}$; (2) a set $E \subseteq V \times \mathcal{C} \times V$ of* edges, *where $[i, k, j] = [j, k, i] \in E$ is the edge between examples $\mathbf{x}_i$ and $\mathbf{x}_j$ via class $c_k \in H(\mathbf{x}_i, \mathbf{x}_j)$, $i \neq j$.*

In this graph, each vertex corresponds to an example $\mathbf{x}_i$. For each class $c_k \in H(\mathbf{x}_i, \mathbf{x}_j)$ two examples $\mathbf{x}_i, \mathbf{x}_j$ have in common, the graph induces an edge $[i, k, j]$. Since every example is part of the root class, there is at least one edge connecting any pair of documents and thus the graph is fully connected (cf. figure 1).

The hierarchy can be used to derive implications about the connections between the examples. The similarity of two examples naturally depends on the number of edges connecting them. If two examples share a common leaf class, the number of edges tends to be rather big because each of the predecessor classes provides an additional edge as well. Thus, examples sharing specific classes are connected by more edges than examples that only share very general common classes. However, the specificity of a class is not exclusively dependent on its level in the hierarchy, but also on the number of documents belonging to the class. For example, if two documents are the only examples belonging to a particular class, then the class is very specific and the connection between both documents is very strong.

In order to express the strength of these connections, we define the *edge weight* $w : E \to \mathbb{R}_+$ as follows:

$$w([i, k, j]) = \frac{1}{|c_k|} \tag{1}$$

where the weights of edges from $\mathbf{x}_i$ via c_k to $\mathbf{x}_j$ evenly divided by the size of c_k. This is consistent with the intuition

that more popular classes are less informative for indicating examples' similarities. Accordingly, by summing over all the shared classes, the induced connection strengths between two examples is computed as

$$w([i,j]) = \sum_{c_k \in H(\mathbf{x}_i, \mathbf{x}_j)} w([i,k,j]) \tag{2}$$

Then the *vertex degree* $d : V \to \mathbb{R}_+$ is defined as:

$$d(i) = \sum_j w([i,j]) \tag{3}$$

which is the total strength of all of edges connected with $\mathbf{x}_i$. Our definition of edge weights and vertex degrees can also be justified from a *random walk* point of view. Suppose a reader is browsing documents in a hierarchical directory. The transition probability from document $\mathbf{x}_i$ to document $\mathbf{x}_j$ via category c_k should be

$$p(j,k|i) = p(j|i,k)p(k|i) = \frac{w([i,k,j])}{d(i)} \tag{4}$$

Then the *expected* transition probability from $\mathbf{x}_i$ to $\mathbf{x}_j$ is

$$p(j|i) = \sum_{c_k \in H(\mathbf{x}_i, \mathbf{x}_j)} p(j,k|i) = \frac{w([i,j])}{d(i)} \tag{5}$$

$Eq.\ (5)$ indicates that the transition probability depends not only on the number of classes shared by $\mathbf{x}_i$ and $\mathbf{x}_j$, but it is also dependent on the size of these classes. Therefore, transitions across high-level branches are considered to be rather unlikely while transitions within deep or small branches occur with a rather high probability.

In our approach, multi-labeled documents are naturally handled. More importantly, these multi-labeled documents are connected to the documents from different branches and somehow inform a closeness of these branches. Thus the similarity of documents and the similarity of classes are further informing to each other, which is similar to the *hub-authority* idea in web search [11].

2.3 Regularization on the Similarity Graph

We seek for a mapping function $\Phi : \mathcal{X} \to \mathbb{R}^m, m \ll n$, that maps feature vector $\mathbf{x}$ into a new m-dimensional space. It is desired to ensure the mapping functions $\Phi(\mathbf{x})$ to be *consistent* with respect to the structure of $\mathcal{G}|_{\mathcal{T}}(V,E)$. Let the mapping function Φ contain m elementary functions, and each of them $\phi : \mathcal{X} \to \mathbb{R}$ map examples into a one-dimensional space. Intuitively, a tight connection between two examples should induce similar outputs in the new space. Similar to the idea of spectral clustering [15], the *cost* induced by an one-dimensional mapping function is defined as:

$$\Gamma(\phi) = \sum_i \sum_j w([i,j]) \big[\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\big]^2 \tag{6}$$

The cost function emphasizes the variations of $\phi(\mathbf{x})$ between tightly connected examples. In the following, we call $\Gamma(\phi)$ the *smoothness functional*, since it measures the non-smoothness of ϕ with respect to the hierarchy structure. Furthermore, Eq. (6) can now be rewritten into the following form:

$$\Gamma(\phi) = \boldsymbol{\phi}^\intercal \triangle \boldsymbol{\phi} \tag{7}$$

where $\boldsymbol{\phi} = [\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_N)]^\intercal$ and $\triangle$ is an $N \times N$ matrix:

$$(\triangle)_{[i,j]} = \begin{cases} d(i), & \text{if } i = j \\ -w([i,j]), & \text{otherwise} \end{cases} \tag{8}$$

In this paper, we mainly consider linear functions $\phi(\mathbf{x}) = \mathbf{w}^\intercal \mathbf{x}$. Then we replace $\Gamma(\phi)$ by $\Gamma(\mathbf{w})$ and write the smoothness functional as:

$$\Gamma(\mathbf{w}) = \mathbf{w}^\intercal \mathbf{X}^\intercal \triangle \mathbf{X} \mathbf{w} \tag{9}$$

The cost can be easily plugged into a formalism of latent semantic indexing (LSI), to ensure the derived features consistent with the structure of hierarchies (see Sec.2.4).

2.4 Hierarchy-Regularized Latent Semantic Indexing

The high dimensionality (typically tens of thousands) of text data always hampers the generalization of learning machines and seriously increases the computational costs. However, in general, the *effective subspace* responsible for the document labels has often a lower dimensionality. Latent semantic indexing (LSI)[6] is a popular feature-reduction technique for text data that identifies such a subspace. The method is however unsupervised and cannot incorporates additional information.

In this section, we employ the hierarchical structure to identify the effective subspace of text data. Various algorithms (e.g., clustering, classification and retrieval) can then be efficiently and effectively based on the new low dimensional feature space.

First, we derive a formalism of LSI such that the hierarchy-induced cost Eq. (7) can be easily plugged in. Let $\Phi : \mathcal{X} \to \mathbb{R}^m$ be the feature mapping consisting of m linear functions $\phi_j(\mathbf{x}) = \mathbf{w}_j^\intercal \mathbf{x}$, $j = 1, \ldots, m$. LSI finds the projections of data $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\intercal$ by applying singular value decomposition (SVD):

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^\intercal$$

where $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_N]$ is an $N \times N$ matrix, $\mathbf{D}$ is an $N \times n$ diagonal matrix with diagonal entries sorted non-increasingly, and $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$ an $n \times n$ matrix. Then the results of mapping Φ on $\mathbf{X}$ are given by the first m columns of $\mathbf{U}$.

In the following theorem, we interpret SVD from a different point of view.

Theorem 2.3. *Let $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^\intercal$ be the singular value decomposition of $\mathbf{X}$. Then $\mathbf{u}_j = \mathbf{X}\mathbf{w}_j$ where $\mathbf{w}_j$ are the solutions to*

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{R}^n} \quad & \|\mathbf{w}\|^2 \\ \textit{s.t.} \quad & \mathbf{w}^\intercal\mathbf{X}^\intercal\mathbf{X}\mathbf{w} = 1, \\ & \mathbf{X}\mathbf{w} \perp \mathbf{X}\mathbf{w}_1, \ldots, \mathbf{X}\mathbf{w}_{j-1} \end{aligned}$$

Proof. we give the sketch. $\mathbf{U} = \mathbf{X}\mathbf{V}\mathbf{D}^{-1}$ apparently suggests $\mathbf{u}_j = \mathbf{X}\mathbf{w}_j$. Let $\mathbf{K} = \mathbf{X}\mathbf{X}^\intercal$, then it is known that $\mathbf{u}_j$ are the directions maximizing $\mathbf{u}^\intercal\mathbf{K}\mathbf{u}$ with constraints $\mathbf{u}^\intercal\mathbf{u} = 1$ and $\mathbf{u} \perp \mathbf{u}_1, \ldots, \mathbf{u}_{j-1}$. The objective can be replace by $\min_{\mathbf{u}} \mathbf{u}^\intercal\mathbf{K}^{-1}\mathbf{u}$. Inserting $\mathbf{u}_j = \mathbf{X}\mathbf{w}_j$ to the optimization problem completes the proof. □

Theorem 2.3 gives the formalism of LSI that enables us to easily perform *hierarchy-regularized LSI* (HLSI). Since we wish to have the mapping functions to be consistent with the class hierarchy, the optimization problem for HLSI is denoted as follows:

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{R}^n} \quad & \gamma\|\mathbf{w}\|^2 + \mathbf{w}^\intercal\mathbf{X}^\intercal\triangle\mathbf{X}\mathbf{w} \\ \textit{s.t.} \quad & \mathbf{w}^\intercal\mathbf{X}^\intercal\mathbf{X}\mathbf{w} = 1, \\ & \mathbf{X}\mathbf{w} \perp \mathbf{X}\mathbf{w}_1, \ldots, \mathbf{X}\mathbf{w}_{j-1} \end{aligned} \tag{10}$$

where $\gamma \in \mathbb{R}^+$, $\mathbf{w}^\intercal\mathbf{X}^\intercal\triangle\mathbf{X}\mathbf{w}$ is the cost induced by the hierarchy structure, and γ determines how much the projections should tend to follow the structure of input features. From the regularization point view, γ prevents the mappings from being over-fitted by the hierarchy structure. When $\gamma \to \infty$, HLSI becomes identical to LSI.

By setting the derivatives of its Lagrange formalism to be zero, it turns out that the linear weights are the solutions to a generalized eigenvalue problem:

$$(\gamma\mathbf{I} + \mathbf{X}^\intercal\triangle\mathbf{X})\mathbf{w} = \lambda\mathbf{X}^\intercal\mathbf{X}\mathbf{w}$$

The m generalized eigenvectors with the smallest eigenvalues are the linear weights $\mathbf{w}_j$, $j = 1, \ldots, m$, of the feature mapping functions.

Since text data is usually very high-dimensional, it is very expensive to solve the large scale generalized eigenvalue problem. The following theorem enables the algorithm to work in the dual space where the dimensionality depends on the number of examples.

Theorem 2.4. *The solutions $\mathbf{w}_j$, $j = 1, \ldots, m$, to the HLSI problem have the form*

$$\mathbf{w}_j = \sum_i (\boldsymbol{\alpha}_j)_i \mathbf{x}_i = \mathbf{X}^\intercal\boldsymbol{\alpha}_j$$

Proof. Let S be the space $\mathrm{span}\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and P the projection onto it. Then $\mathbf{w} = P\mathbf{w} + (\mathbf{I} - P)\mathbf{w} = \mathbf{w}_\parallel + \mathbf{w}_\perp$. Since $\mathbf{w}_\perp$ does not affect $\Gamma(\mathbf{w})$ but only increases $\|\mathbf{w}\|^2$, $\mathbf{w}_\perp$ must be zero at the optimum. Therefore $\mathbf{w} \in S$ which completes the proof. □

Then, the HLSI problem has the dual form, $\phi_j(\mathbf{x}) = \sum_i (\boldsymbol{\alpha}_j)_i \langle \mathbf{x}_i, \mathbf{x}\rangle$, $j = 1, \ldots, m$, where $\boldsymbol{\alpha}_j$ are solved by

$$\begin{aligned} \min_{\boldsymbol{\alpha}\in\mathbb{R}^N} \quad & \gamma\boldsymbol{\alpha}^\intercal\mathbf{K}\boldsymbol{\alpha} + \boldsymbol{\alpha}^\intercal\mathbf{K}\triangle\mathbf{K}\boldsymbol{\alpha} \\ \textit{s.t.} \quad & \boldsymbol{\alpha}^\intercal\mathbf{K}\mathbf{K}\boldsymbol{\alpha} = 1, \\ & \mathbf{K}\boldsymbol{\alpha} \perp \mathbf{K}\boldsymbol{\alpha}_1, \ldots, \mathbf{K}\boldsymbol{\alpha}_{j-1} \end{aligned}$$

where $\mathbf{K} = \mathbf{X}\mathbf{X}^\intercal$. The problem is also equivalent to a generalized eigenvalue problem:

$$(\gamma\mathbf{K} + \mathbf{K}\triangle\mathbf{K})\boldsymbol{\alpha} = \lambda\mathbf{K}\mathbf{K}\boldsymbol{\alpha}$$

Finally, the learned mapping functions transform a high-dimensional feature vector $\mathbf{x}$ to a m-dimensional space. In the new space, data mining and retrieval tasks can be efficiently done.

2.5 Hierarchy-Regularized Classifier

HLSI also suggests a direct optimization approach to handle the *multi-label* hierarchical categorization problem. Clearly, the simplest solution is to train binary classifiers for each leaf class c_k. Given the training examples $\mathbf{X} = [\mathbf{x}_i, \ldots, \mathbf{x}_N]^\intercal$ with labels $(\mathbf{y}_k)_i \in \{+1, -1\}$ for class c_k, a linear classifier[1] $\phi_k(\mathbf{x}) = \mathbf{w}_k^\intercal\mathbf{x}$ can be learned by

$$\mathbf{w}_k = \arg\min_{\mathbf{w}} \sum_i \ell(\mathbf{w}^\intercal\mathbf{x}_i, (\mathbf{y}_k)_i) + \beta\|\mathbf{w}\|^2 \tag{11}$$

where $\ell(\cdot,\cdot)$ is the loss function, $\beta \in \mathbb{R}^+$, and $\|\mathbf{w}\|^2$ is the regularizer preventing overfitting. The optimization in Eq. (11) treats all negative examples identically. There is no bigger penalty if an example is miss-classified into a leaf class which is faraway in the tree from the correct class. Therefore, we insert $\Gamma(\mathbf{w}) = \mathbf{w}^\intercal\mathbf{X}^\intercal\triangle\mathbf{X}\mathbf{w}$ into the optimization problem Eq. (11) and get the following objective function

$$J_k(\mathbf{w}) = \sum_i \ell(\mathbf{w}^\intercal\mathbf{x}_i, (\mathbf{y}_k)_i) + \xi\Gamma(\mathbf{w}) + \beta\|\mathbf{w}\|^2 \tag{12}$$

[1] Linear support vector machine is the state of the art method for text categorization (see [20, 17]).

where $\beta, \xi \in \mathbb{R}^+$ and $\mathbf{I} \in \mathbb{R}^{N \times N}$ is an identity matrix. In Eq. (12) there are two parts of loss based on empirical data: one is the conventional classification loss $\ell(\mathbf{w}^\intercal \mathbf{x}_i, (\mathbf{y}_k)_i)$, the other is the hierarchy-induced loss $\Gamma(\mathbf{w})$.

The square error loss $\ell(\mathbf{w}^\intercal \mathbf{x}_i, (\mathbf{y}_k)_i) = (\mathbf{w}^\intercal \mathbf{x}_i - \mathbf{y}_k)^2$ was often reported to achieve superior performance in text categorization [20, 19]. For this case, the estimate of $\mathbf{w}_k$ has closed form:

$$\mathbf{w}_k = (\mathbf{X}^\intercal \mathbf{X} + \xi \mathbf{X}^\intercal \triangle \mathbf{X} + \beta \mathbf{I})^{-1} \mathbf{y}_k. \qquad (13)$$

which is derived by setting $\frac{\partial J_k(\mathbf{w})}{\partial \mathbf{w}} = 0$.

Let us note that the methods suggested in this section have a close connection to HLSI as suggested in Sec. 2.4. Here, the feature projection is implicitly done via the regularization. Despite its equivalence, explicit feature mappings enable learning methods to work on a low dimensional feature space and greatly improve the efficiency. This advantage is very important for real-world applications.

2.6 Generalization of the Approach

The basic idea of HLSI is to extract principal components of the original input space while one defines a similarity concept on the output space using class hierarchy and encodes data set into a hierarchy-induced similarity graph. The data points that are similar in the output space should be also similar in the input space. This idea can be used not only for a hierarchical structure of classes, but also for an arbitrary intrinsic character of a data set. The main point is the definition of similarity as in $Eq.\ (2)$. For complex data sets with multiple characters one can define several similarity concepts and combine them.

3 Related Work

Dimensionality reduction is a well established approach in data mining and information retrieval. One sort of the most well-known techniques is feature selection, like mutual information, information gain and χ^2 statistic [18]. In general, established feature selection methods ignore the co-occurrence between features, which exists obviously in textual data sets. Furthermore it is difficult for feature selection to deal with multi-label problems. Another sort of dimensionality reduction techniques is feature transformation/mapping. A representative approach is latent semantic indexing [6], which uses singular value decomposition (SVD) to find the principal components of term-document matrices. However, this method is unsupervised and thus the found dimensions are not necessarily relevant when employing labels of data.

Our approach is a supervised feature mapping method. Our framework considers the co-occurrence between features and handles multi-labeled problems in a natural way. Similar methods are the canonical correlation analysis (CCA) [9], partial least square (PLS) and linear discriminant analysis (LDA) [16]. LDA aims to find transformation directions that maximize distances between class means and minimize variances within classes. However, LDA can only handle the single-label problems. PLS and particularly CCA are classical statistical methods and measure the linear correlation between two multidimensional data sets (e.g. inputs and outputs). The difference of both methods is that in CCA the correlation is normalized by variances within two data sets. However, for these two methods the number of transformation directions are limited to the smallest dimensionality of two data sets. This limitation is not in our approach.

The resulting feature representations with our framework are usable for any problem settings in data mining and information retrieval. In particular for hierarchical classification one can add the information of the class taxonomy to the loss function as suggested in Sec. 2.5. However, none of the former hierarchical classification approaches [12, 13, 8, 1, 14, 4, 7, 3] does directly influence the employed feature space using a class hierarchy like our new framework. Another conceptual difference is that we use the information about the specificity of the classes as well as the information about the similarity between classes.

4 Empirical Study

4.1 Data Sets

In order to demonstrate the advantages of the introduced framework, we evaluated our methods on two real-world data sets. The first is the Reuters Corpus Volume 1 (RCV1) which consists of 806,791 English news stories. We randomly chose 10,000 documents from this data set having 31,613 class labels. These class labels refer to a class hierarchy of 81 classes, 64 leaf classes and 17 inner classes. The depth of the tree is 4 and each topic is represented by at least 20 documents. In all 10,000 examples occur 9,705 different words. The second data set is derived from the Swiss-Prot[2] protein database that contains textual annotations of proteins. The entries in Swissprot provide links to the class system of Gene Ontology (GO) [5] which is used as a class hierarchy. We selected the subtree "Oxido-reductase" from GO, which contains 125 categories. The corresponding entries in Swissprot comprise 8,335 proteins having 18,955 labels. The class hierarchy has also a maximum depth of 4 and provides 94 leaf classes and 31 inner classes. For each category, there are at least 10 entries available. This document collection contains 10,404 different words. For both data sets, we derived an original feature space of word vectors by dropping the words being contained in less than 5 documents and afterwards applied TFIDF.

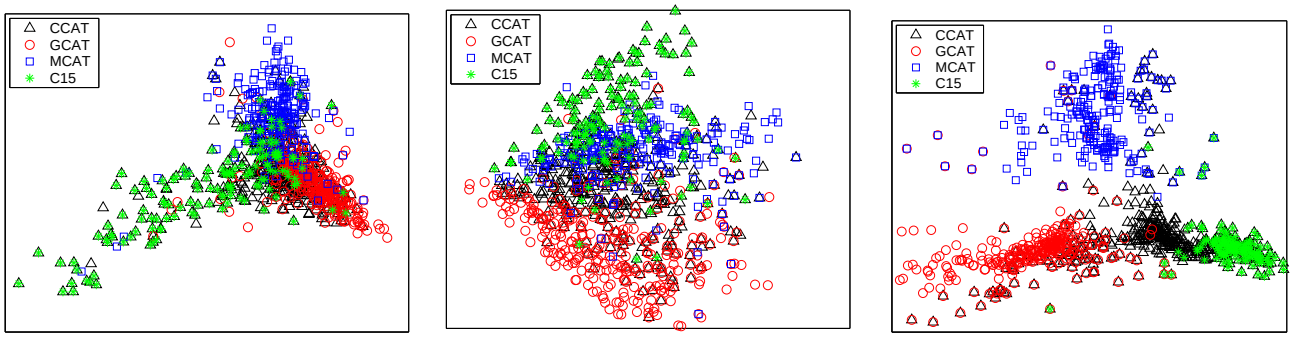


Figure 2. Visualization of the RCV1 on two-dimensional space (left) LSI, (mittle) CCA and (right) HLSI

4.2 Data Visualization

One important application of feature reduction is to visualize data patterns in a 2 or 3-dimensional space, providing an impression about the quality of the underlying feature space. Therefore, we used LSI, CCA and HLSI to project the documents into a two-dimensional space and visualized them. The results for RCV1 data are displayed in Fig. 2. Let us note that we observed a similar visualization for the Swissprot data, but we had to omit it due to the space limitation. The mapping functions are computed based on 2000 documents and were afterwars applied to project another set 1000 documents[2]. We visualized 3 top-level classes and one second-level class C15 which is a subclass of the class CCAT. In Fig. 2 different colors and marks are used to distinguish the classes. The results of LSI and CCA do not present a very meaningful data distribution because the classes are not separated very well. On the other hand, using HLSI provided a visualization which is quite relevant to the class memberships, in the sense that documents from the same class often display a close distance. This result indicates that the method employs the multi-label information to represent the class similarities. Since class C15 is a subclass of CCAT, HLSI mapped the members of C15 into a concentrated subregion of area it mapped the members of CCAT. Thus, the projection preserved the inheritance relationship within the data. The good performance of HLSI on visualization demonstrates that the proposed algorithm effectively detects the meaningful subspaces within hierarchical data. Thus, the resulting feature spaces should allow fast and accurate solutions for various learning and retrieval tasks.

4.3 Classification Performance

Our second set of experiments studied the quality of HLSI in terms of dimensionality reduction for text classifications. The experimental results are evaluated using macro-averaged F_1 and micro-averaged F_1, which are suitable to measure the classification accuracy when the classes are very unbalanced. In particular, micro-averaged F_1 reflects the quality on the classes with a large number of positive examples, while macro-averaged F_1 emphasizes on the minor classes, which correspond to the leaf classes in the hierarchical case. As a comparison, we investigate the quality of 4 different feature spaces: The first was generated by LSI, the second by HLSI, the third by CCA and the last was the original feature space. For classification, we used a linear support vector machines (SVMs) that was implemented in the SVM^{light} package [10]. For each run, we randomly selected 2000 examples with the constraint that each leaf class had at least 5 positive examples. Treating the selected data as training set, we trained classifiers for all inner classes and leaf classes. Then, the trained models were used to predict the class of the remaining 8000 examples. For LSI, CCA and HLSI, the same set of 2000 examples were employed for learning the feature mapping. We changed the dimensionality of projections and compared change of performance. The experiment was randomized for 10 times and the mean and error bar of the results were computed.

Finally the results are shown in Fig. 3. We can see that, the full-feature case is always working very well. HLSI gives the performance significantly better than LSI. In the case of micro-averaged F_1 for RCV1 data, the performance of 50-dimensional HLSI features is almost as good as full features, while the cases of more than 80-dimensional HLSI features are even better than full features. In the meantime LSI needs 200 dimensions to reach almost the same performance. Similar observations can be made in the other 3 subplots. In general, 50-dimensional HLSI features for RCV1 data and 80-dimensional features for Swissprot data are sufficient to give comparable accuracy as the full-feature case, however, the calculation for training the mapping only needs to be done once, summing up over the training of all classifiers, the total cost is much smaller than

[2]More data points make the figures difficult to render without color.

SVMs using full features. In our experiments, we observed on RCV1 data a 10 times improvement of efficiency with 20-dimensional HLSI, and a 4 times improvement for 50-dimensional HLSI.

We also performed top-down classification on the class hierarchy, as the method was mentioned in many papers (e.g. [12]). However, we did not observe any big differences from the setting we just described. All the methods including HLSI showed the same behaviors. Moreover, we test the performance of the proposed hierarchy-regularized classifier, however the method shows close predictive accuracy as SVMs using the full feature. The reason seems to be that in the high-dimensional feature spaces, with 2000 training examples, the classifiers converges to almost the same hyperplane anyway.

4.4 Sensibility of γ

There is a parameter γ in HLSI algorithm. Here we examine how sensitive the performance of HLSI on the setting of γ. Here we run the same setting of experiments described in the last section, with only γ changed. All the evaluations are averaged from 10 random repeats. We only report the mean in Fig. 4, while omit the error bar since some curves stay very close, like the cases of $\gamma = 0.01$ or 0.001 in subplots (c) and (d), which are almost completely overlapped. From the figures, we can see for the RCV1 data, the optimal choice of γ is 0.01 or 0.1, while for the Swissprot data, the optimal setting is 0.001 or 0.01. In general, the methods using the hierarchy information is always better than LSI without this information. However, the setting for RCV1 data seems to be a little bit sensitive—large values (e.g. $\gamma \to \infty$ to become LSI) or small values (e.g. $\gamma = 0.001$) both degrade the performance. In general, the setting of parameter γ depends on the nature of data. In practice, we need a valid set to guide the selection.

5 Conclusions

In this paper, we introduced a new method for using class hierarchies in textual machine learning. Our method is based on a hierarchy-regularized framework that incorporates the proximity of classes within the hierarchy which implies a connection between the documents belonging the same class. Additionally, the framework uses the specificity of classes which can be measured by the number of documents belonging to each class. We use our framework to derive a new hierarchy-regularized method for feature transformation that is based on latent semantic indexing called HLSI. HLSI enables us to integrate the information within a class hierarchy into a variety of learning and retrieval tasks. Additionally, our experiments on two real-world text data sets demonstrate that the proposed methods are capable to derive low dimensional and highly descriptive feature spaces that mirror the structure of the underlying class hierarchy. Thus, they are well-suited for a variety of learning tasks employing hierarchical class systems. For future work, an interesting direction is to develop a global method for feature selection based on hierarchy induced graphs. This is especially interesting for applications demanding human understandable class models.

References

[1] H. Blockeel, M. Bruynooghe, S. Dzeroski, J. Ramon, and J. Struyf. Hierarchical mult-classification. In *MRDM Workshop on Multirelational data mining at SIGKDD'02, Edmonton, Canada*, 2002.

[2] B. Boeckmann, A. Bairoch, R. Apweiler, M.-C. Blatter, A. Estreicher, E. Gasteiger, M. Martin, K. Michoud, C. O'Donovan, I. Phan, S. Pilbout, and M. Schneider. "The SWISS-PROT Protein Knowledgebase and its Supplement TrEMBL in 2003". *Nucleic Acid Research*, 31:365–370, 2003.

[3] L. Cai and T. Hofmann. Hierarchical document categorization with support vector machines. In *Proc. 13th Conf. on Information and Knowledge Management(CIKM'04),Washington D.C., USA*, pages 78 – 87, 2004.

[4] N. Cesa-Bianchi, C. Gentile, A. Tironi, and L. Zaniboni. Incremental algorithms for hierarchical classification. In *Proc. 8th ann. Conf. on Neural Information Processing Systems, Vancouver, BC, Canada*, 2003.

[5] T. G. O. Consortium. "Gene Ontology: Tool for the Unification of Biology". *Nature Genetics*, 25:25–29, 2000.

[6] 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.

[7] O. Dekel, J. Keshnet, and Y. Singer. Large margin hierarchical classification. In *Proc. 21th International Conf. on Machine Learning (ICML'04), Banff, Canada*, 2004.

[8] S. Dumais and H. Chen. "Hierarchical Classification of Web Content". In *Proc. 23rd Int. Conf. on Research and Development in Information Retrieval (SIGIR'00)*, pages 256–263, 2000.

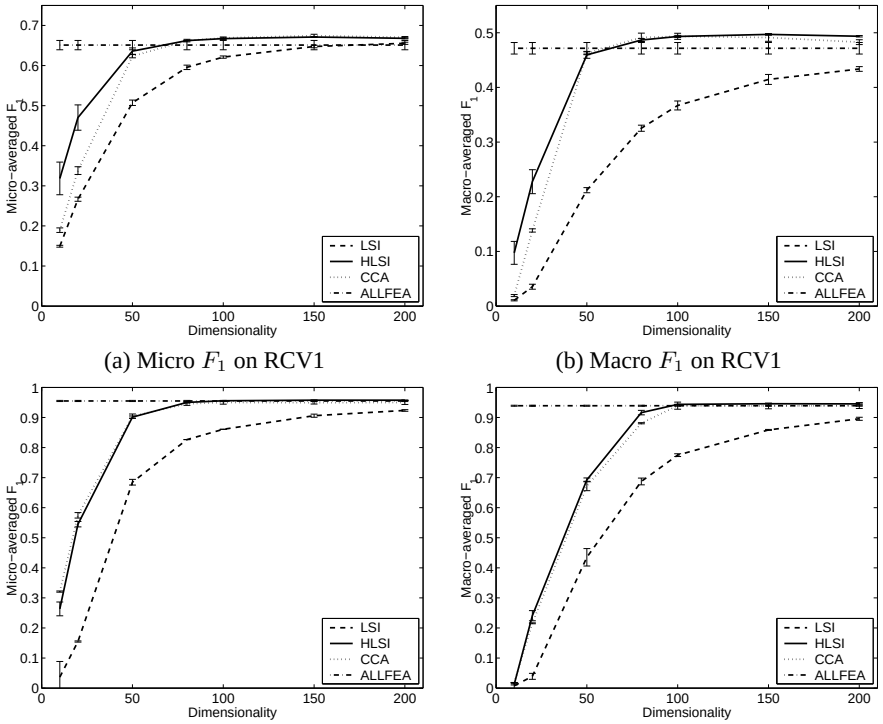
[9] H. Hotelling. Relations between two sets of variables. *Biometrika*, 28:321–377, 1936.

[10] T. Joachims. Text categorization with support vector machine: learning with many relevant features. In *Proceeding of (ECML)-98, 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

[11] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[12] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *Proc. 14th Int. Conf. on Machine Learning (ICML'97), Nashville, TN*, pages 170–178, 1997.

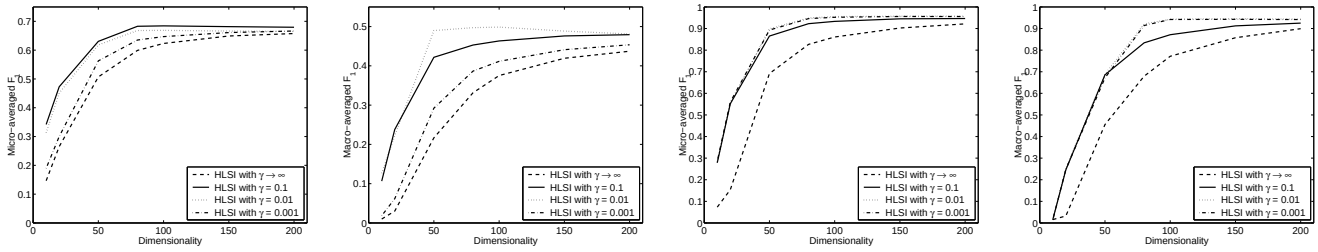
[13] A. McCallum, R. Rosenfeld, T. Mitchell, and A. Ng. "Improving Text Classification by Shrinkage in a Hierarchy of Classes ". In *Proc. 15th Int. Conf. on Machine Learning (ICML'98), Madison, WI*, pages 359–367, 1998.



(a) Micro F_1 on RCV1 (b) Macro F_1 on RCV1

(c) Micro F_1 on Swissprot (d) Macro F_1 on Swissprot

Figure 3. Classification of the RCV1 data set (a,b) and the Swissprot data set(c,d) with different features



(a) (b) (c) (d)

Figure 4. Comparison different γ of the RCV1 data set (a,b) and the Swissprot data set(c,d) with micro-averaged F_1 (a,c) and macro-averaged F_1 (b,d).

[14] J. Rousu, C. Saunder, S. Szedmak, and J. Shawe-Taylor. On maximum margin hierarchical mulitlabel classification. In *Proc. of Workshop on Learning with Structured Outputs at NIPS 2004, Whistler, Canada*, 2004.

[15] J. Shi and J. Malik. Normalized cuts and image segmentation. In *IEEE Conf. Computer Vision and Pattern Recognition(CVPR)*, 1997.

[16] H. Wold. Partial least squares. *Encyclopedia of the Statistical Sciences*, pages 581–591, 1985.

[17] Y. Yang and X. Liu. A re-examination of text categorization methods. In *The 22th Annual International SIGIR Conference (SIGIR'99)*, pages 42–49, 1999.

[18] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In D. H. Fisher, editor, *Proc. 14th International Conference on Machine Learning (ICML'97)*, pages 412–420, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.

[19] J. Zhang and Y. Yang. Robustness of regularized linear classifcation methods in text categorization. In *The 26th Annual International SIGIR Conference (SIGIR'99)*, 2003.

[20] T. Zhang and F. J. Oles. Text categorization based on regularized linear classification methods. *Information Retrieval*, (4):5–31, 2001.