Batch Mode Active Learning with Hierarchical-Structured Embedded Variance

Yu Cheng[*] [†] Zhengzhang Chen [*] Hongliang Fei[†] Fei Wang[†] Alok Choudhary[*]

Abstract

We consider the problem of active learning when the categories are represented as a tree with leaf nodes as outputs and internal nodes as clusters of the outputs at multiple granularity. Recent work has improved the traditional techniques by moving beyond "flat" structure through incorporation of the label hierarchy into the uncertainty measure. However, these methods have two major limitations when used. First, these methods roughly use the information in the label structure but do not take into account the training samples, which may lead to a sampling bias due to their crude approximation of the class relations. Second, none of these methods can work in a batch mode to reduce the computational time of training. We propose a batch mode active learning scheme that exploits both the hierarchical structure of the labels and the characteristics of the training data to select the most informative data for human labeling. We achieve this goal by first using an approach based on graph embedding that embeds the relationships between the labels and data points in a transformed low-dimensional space. Then, we compute uncertainty by calculating the variance among the points and the labels in the embedding space. Finally, the selection criterion is designed to construct batches and incorporate a diversity measure. Experimental results indicate that our technique achieves a notable improvement in performance over the state-of-the-art approaches.

1 Introduction

Multi-class classification is a problem that arises in many applications of data mining, such as document and web categorization. Large scale classification problems usually involve hundreds or even thousands of possible classes and need a lot of training data that requires expensive and time-consuming labeling work. To make the learning task more efficient, it is imperative to intelligently choose specific unlabeled instances to be labeled by an oracle/user. The general problem of optimally choosing these instances is known as active learning [1]. Within an active learning framework, instances that maximize model uncertainty are the most informative ones and are the ones that are selected. The uncertainty is usually quantified by entropy of class posterior probabilities or distances from the decision hyper-plane, with the assumption that all the class priori probabilities are known.

Previous active learning techniques modeled the label relationship with a "flat" structure [2,3], in which each class/category was independently tackled by a binary active learning algorithm. However, it is often the case that the categories are not just discrete classes, but organized in a hierarchical structure, or a category tree. For example, web pages can be categorized into the Yahoo! web hierarchical taxonomy, and images can be organized according to the WordNet hierarchy. When considering the hierarchical multi-class classification problems, most of these approaches treat all the class labels independently and largely ignore the rich inherent relationships among the multi-class labels.

Most recent approaches exploit information in the hierarchy structure [4,5] into the active learning setting. For example, [4] estimated the cost of misclassification based on the geodesic distance of two class labels in the category tree. And [5] proposed a variance-based measure with incorporating the label hierarchy. Although modeling the label relationship using the "hierarchy tree" can boost the performance compared to the "flat" structure, these methods have two main limitations when deployed to hierarchical classification. First, in reality, it is not sufficient to estimate the uncertainty information using only the information embedded in the label structure. The class label is usually a very crude approximation of the class relations created by an editor with knowledge of the true underlying semantic space of the class [6]. And it would be even worse when the prior knowledge is wrong. It may lead to sampling bias by exploiting the label information. Second, instead of selecting a batch of samples simultaneously, these active learning methods are designed to select a single sample for each learning iteration which requires too much computational effort. Directly applying the none-batch methods may also result in selecting multiple examples that are similar (or even identical) to each other. We refer to these two problems as "label bias" problem and

[*] EECS Department, Northwestern University, IL 60208, USA
[†] IBM T.J. Watson Research Center, NY 10582, USA

"batch sampling" problem, respectively.

To address the "label bias" problem, in this paper, we propose a graph embedding based active learning approach for hierarchical classification, named Hierarchical-Structured Embedded Variance (HSEV). It handles the "label bias" problem by exploiting both the externally provided class hierarchy and the training data into active learning selections. To solve the "batch sampling" problem, we propose the Batch Mode Hierarchical-Structured Embedded Variance ($\text{HSEV}_{\text{Dive}}$), which can select multiple samples simultaneously via a combined strategy incorporating diversity measure. We show that the proposed approach results in notable improvement upon the learning rate (and performance) of the state-of-the-art active learning methods. In summary, our contributions are:

1. We propose a graph embedding based approach exploiting both the hierarchical structure of the label tree and characteristics of the training data to measure the uncertainty. The structure information of the labels is induced into the learning algorithm.
2. We demonstrate that the proposed sequential mode Hierarchical-Structured Embedded Variance sampling is a generalization of some other active learning methods for hierarchical classification.
3. We present a framework for actively selecting a batch of unlabeled examples simultaneously. It has low computational requirements making it feasible for large scale problems with thousands of samples.

2 Related Work

2.1 Active Hierarchical Classification

The goal of active learning is to choose informative instances to label, so as to achieve a low classification error with few iterations [7,8]. Given a large set of unlabeled instances $(\mathbf{x}_1, ..., \mathbf{x}_n) \in \mathcal{U}$, in each step, the learner is allowed to use the query function Q and select an unlabeled instance $\mathbf{x} \in \mathcal{U}$ to be labeled. The most commonly used query function is entropy based sampling. Given the posterior $p(y_i|\mathbf{x}_i)$ of every point $\mathbf{x}_i$ in $\mathcal{U}$, the query instance is found by maximizing the entropy function:

$$\text{Entropy}(\vec{y}, \mathbf{x}) = -\sum_{\alpha=1}^{m} p_\alpha \log p_\alpha \tag{2.1}$$

where $p_\alpha = p(y = \alpha|\mathbf{x})$ and y ranges over all possible labels. Nader *et al.* [9] found that using variance to measure uncertainty has a very similar performance to entropy. The variance-based measure is to select the instance which maximizes the variance function:

$$\text{Var}(\vec{y}, \mathbf{x}) = \sum_{\alpha=1}^{m} p_\alpha (1 - p_\alpha) \tag{2.2}$$

Relatively few approaches exploit the relationship of the labels to incorporate label structure for active learning. One way to incorporate the structural relationship of the labels is to use cost-based measure [4], which estimates the cost of misclassification using the geodesic distance of two class labels in the category tree to calculate the uncertainty:

$$\text{Cost}(\vec{y}, \mathbf{x}) = \sum_{\alpha=1}^{m}\sum_{\beta=1}^{m} C_{\alpha\beta} p_\alpha p_\beta \tag{2.3}$$

where $C_{\alpha\beta}$ is a cost measure of misclassification, and set to be the geodesic distance between node α and node β in the hierarchical label tree. This method is referred as cost-based sampling in the paper. Yu *et al.* [5] proposed a variance-based measure with incorporating the label structure, denoted as hierarchy active sampling (HAS):

$$\text{HAS}(\vec{y}, \mathbf{x}) = \text{Tr}(\sum_{\alpha=1}^{m} p_\alpha {\mathbf{v}_\alpha}^T \mathbf{v}_\alpha - d_{\vec{y}}^T d_{\vec{y}}) \tag{2.4}$$

where $\mathbf{v}_\alpha$ is transformation of label space $\mathbf{y}$, which is approximately estimated using the geodesic distance of the labels, and $d_{\vec{y}} = \sum_{\alpha=1}^{m} p_\alpha \mathbf{v}_\alpha$ is the mean vector of each label vector. However, these methods either model the label relationship as "flat" or roughly take the label hierarchy into consideration with ignoring the training data information. In contrast, the proposed method, Hierarchical-Structured Embedded Variance, exploits both the structure of the label tree and the training data in measuring the uncertainty, and is able to achieve higher sampling efficiency.

2.2 Batch Mode Active Learning

Pool-based active learning methods can be sub-categorized as two sets: one is sequential mode, where a single point is queried at a time; another is batch mode, where a batch of points is queried simultaneously before updating the classifier. Amidst batch mode techniques, Hoi *et al.* [10] used the Fischer score matrix as a measure of model uncertainty and proposed to select a batch of points which reduced the Fischer score. Brinker [11] proposed a strategy in SVM setting which queried a diverse batch of points where diversity was measured as the distance between the candidate point to all other selected points. Xu *et al* [12] combined the relevance, density and diversity measure to select a batch of candidate queries. Our batch mode strategy is similar to Brinker's and Xu's work, which exploits the diversity function into the active batch selection. But different to their diversity functions which are based on single point distance measure, our proposed graph-based diversity function, is more robust due to the k-NN graph structure.

3 Sequential Mode Hierarchical-Structured Embedded Variance

In this section, we propose the sequential mode Hierarchical-Structured Embedded Variance method HSEV, for active learning to select new instances.

3.1 Hierarchical Structured Embedding One solution for combining both label structure and data information is using the embedding-based methods [6,13,14], which discover a representation of the data points where distances reflect the underlying dissimilarities and proximities based on topic membership. In this paper, we employ the graph embedding scheme to perform the embedding and propose a quadratic programming framework to transform the graph, including training data points and class labels, into a low-dimensional space that preserves the relationships among the labels and training data. The overall framework of the proposed embedding is illustrated as in Figure 1. After embedding, 1) data points within the same or related classes should be close; 2) data points within highly different classes should be well separated; and 3) the embedded label prototype should be able to characterize the label tree structure. Then the label variance, which is computed by treating each label as a point in the transformed space, is utilized to measure the uncertainty, which can well estimate informativeness of a new data point.

We assume that the input data consists of instances, represented as a set of vectors $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ in the training dataset ζ. In addition, each instances $\mathbf{x}_i$ is accompanied by a single topic label $\alpha \in \{1, 2, ..., m\}$ organized into a label tree T with m total topics. A label tree [15] is a tree $T = (V_T, E)$ with nodes V_T and edges E. Each node $v \in V_T$ is associated with a set of class labels $l(v) \subseteq \{1, \cdots, m\}$ so that the set of labels $\{1, \cdots, m\}$ has a one-to-one mapping to the set of leaf nodes, and each non-root node's label set should be a subset of its parent's label set.

In order to use the graph embedding algorithm to embed the label and training data, we first construct the intrinsic graph G. Let $G = (V, \mathbf{W})$ be a weighted graph with the set vertex V and the edge matrix $\mathbf{W}$. V contains all the category labels $\alpha \in \{1, 2, ..., m\}$ and all training data points $\mathbf{x}_i \in \zeta$. The edge matrix $\mathbf{W} \in R^{N \times N}$ $(N = n + m)$ is a similarity matrix and w_{ij} defines the similarity coefficient of vertices v_i and v_j in V. The edges of the intrinsic graph include: (1) edges joining each data point vertex to its nearest neighbor in the data space, and (2) edges joining each label vertex to the data point vertices of the corresponding class. Two vertices connected by an edge in the intrinsic graph means that their embedded points should be close to each other. In other words, the larger the similarity coefficient w_{ij} is, the smaller the distance between two vertices should be. Formally, the edge matrix $\mathbf{W}$ of the intrinsic graph is defined as follows:

(1) For data points $\mathbf{x}_i$ and $\mathbf{x}_j$, the weight of the edge connecting them is:

$$w_{ij} = \begin{cases} 1, & \text{if } i \in N_{k_1}(j) \text{ or } j \in N_{k_1}(i) \\ 0, & \text{otherwise} \end{cases} \tag{3.5}$$

where $N_{k1}(j)$ indicates the index set of the k_1 nearest neighbors of the sample $\mathbf{x_i}$ with the same label. The data points should be near to its k_1-nearest neighbors that has the same label.

(2) For a data point $\mathbf{x}_i$ with label y_i and a label point $\mathbf{x}_\alpha$, the weight of the edge connecting them is:

$$w_{i\alpha} = \begin{cases} 1, & \text{if } \alpha = y_i \\ 0, & \text{otherwise} \end{cases} \tag{3.6}$$

These edges enforce each label prototype to be near to all the training data with that label, so that the label prototype is representative of the training data with this label. The diagonal matrix $\mathbf{D}$ and the Laplacian matrix $\mathbf{L}$ of the graph G are defined as:

$$\mathbf{L} = \mathbf{D} - \mathbf{W}, D_{ii} = \sum_{j \neq i} W_{ij} \tag{3.7}$$

In addition, an intra-class penalty graph $G^p = (V, \mathbf{W}^p)$ is defined as a graph whose vertices V are the same as those of G, but the weighted edge matrix $\mathbf{W}^p$, corresponding to the similarity characteristics is opposite to $\mathbf{W}$ in the embedding feature space, i.e., the larger the edge element w_{ij}^p is, the larger the distance between v_i and v_j should be. The edges of the intra-class penalty graph include: (1) edges joining two data point vertices with different classes, and (2) edges joining two label vertices. The $\mathbf{W}^p$ of the penalty graph is constructed as follows:

(1) For data points $\mathbf{x}_i$ and $\mathbf{x}_j$, the weight of the edge connecting them is:

$$w_{ij}^p = \begin{cases} 1, & \text{if } (i,j) \in P_{k_2}(c_i) \text{ or } (i,j) \in P_{k_2}(c_j) \\ 0, & \text{otherwise} \end{cases} \tag{3.8}$$

where c_i is the class of data $\mathbf{x_i}$, $P_{k_2}(c_i)$ is a set of data point pairs that are the k_2 nearest pairs among the set $\{(i,j), y_i = c, y_j \neq c\}$. These edges maintain the neighborhood structure of the nearest data point pairs by pushing the data points that have different labels further away.

(2) For a label point $\mathbf{x}_\alpha$ and a label point $\mathbf{x}_\beta$, the weight of the edge connecting them is

$$w_{\alpha\beta}^p = C_{\alpha\beta} \tag{3.9}$$

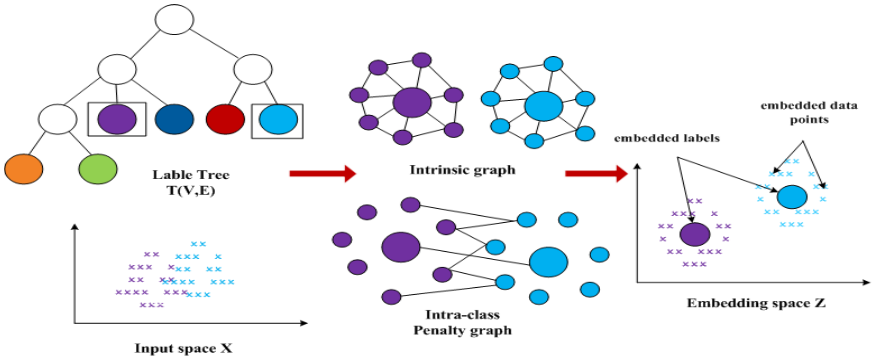


Figure 1: A schematic layout of the proposed graph embedding approach. An intrinsic graph and penalty graph are constructed to reflect the structure of the data and the categories. These graphs are then used to embed the all training data as well as the category labels to a low-dimensional space.

where $C_{\alpha,\beta}$ is the class distance between classes α and β ($C_{\alpha,\alpha} = 0$), which is the length of the shortest path between the corresponding nodes in the label tree. In this way, the distance of the labels that are further away in the label tree will be penalized more heavily in the graph embedding.

Given all the category labels $\alpha \in \{1, 2, ..., m\}$ and the labeled dataset $\zeta = [(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_n, y_n)]$, we would like to embed both the labels and the data into the same space $\mathbf{Z} \in R^{N\times d}$. We represent the embedded space as a vector $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_N]^T$, where $\mathbf{z}_i$ is the d-dimensional representation of data point $\mathbf{x_i}$ or a label α. The goal of graph embedding for the intrinsic graph G is to find the desirable low dimensional vector representation $\mathbf{z}_i$ that best characterizes the similarity relationship between the vertex pairs in G, where the criterion formulated based on G^p is also preserved [14]. By setting the intrinsic graph $\mathbf{W}$ and the penalty graph $\mathbf{W}^p$ as inputs, the graph-preserving embedding algorithm works by minimizing the following objective function:

$$\mathbf{Z} = \arg\min(\mathrm{Tr}(\mathbf{Z^T}(\mathbf{L} - \mathbf{B})\mathbf{Z}) + \lambda||\mathbf{Z}||_F^2) \tag{3.10}$$

where Tr() is the trace function, $\lambda||\mathbf{Z}||_F^2$ is a regularization term to bound objective function, and $\mathbf{B}$ is the matrix determined by the penalty graph $\mathbf{G}^p$:

$$\mathbf{B} = \mathbf{D}^p - \mathbf{W}^p, D_{ii}^p = \sum_{j\neq i} W_{ij}^p \tag{3.11}$$

where $\mathbf{D}^p$ is the diagonal matrix. The i-th row of the matrix $\mathbf{Z}$ is the embedded vector representation $\mathbf{z}_i$ of the graph. The optimization of Eq. 3.10 is a Quadratic Programming (QP) problem [16], and the objective function is strictly convex when $\lambda \geq \mathrm{abs}(\psi), \forall\psi \in \Psi$, where Ψ is eigenvalues set of $(\mathbf{L} - \mathbf{B})$ (it is easy to prove the positive definiteness). It has a unique local minimum which is also a global minimum. Ellipsoid method [17] can find global minimum of the objective function in polynomial time.

3.2 Variance-based Uncertainty Measure

After embedding each class label α into a point $\mathbf{z}_\alpha$ in embedding space $\mathbf{Z}$, the variance can be defined as the trace of the covariance matrix of all the label points:

$$\mathrm{VOI}(\vec{y}, \mathbf{x}) = \mathrm{Tr}(\sum_{\alpha=1}^{m} p_\alpha \mathbf{z}_\alpha{}^T \mathbf{z}_\alpha - d_{\vec{y}}^T d_{\vec{y}}) \tag{3.12}$$

where $\mathrm{VOI}(\vec{y}, \mathbf{x})$ is the uncertainty measure, $d_{\vec{y}} = \sum_{\alpha=1}^{m} p_\alpha \mathbf{z}_\alpha$ is the mean vector of each label vector, and p_α is the estimated probability that x belongs to class α. In each step, the point $\mathbf{x}$ with the greatest uncertainty is selected for labeling. We refer this sequential mode Hierarchical-Structured Embedded Variance method as HSEV.

Here we demonstrate that HSEV is a generalization of the variance-based, cost-based and Hierarchy Active Sampling (HAS) measures. If all the labels are equivalent, they can be embedded uniformly into a high dimensional Euclidean space. Suppose $y \in \{1, ..., m\}$, then we can embed them into a m-dimensional space. Formally, we can embed a label $y = \alpha$ to $\mathbf{e}_\alpha$, ($\mathbf{z}_\alpha = \mathbf{e}_\alpha$), where $\mathbf{e}_\alpha$ is a vector with its α-th element 1 and the rest of elements 0. Then, the uncertainty measure Eq. 3.12 can be derived as:

$$\begin{aligned} \mathrm{VOI}(\vec{y}, \mathbf{x}) &= \mathrm{Tr}(\sum_{\alpha=1}^{m} p_\alpha \mathbf{e}_\alpha^T \mathbf{e}_\alpha - (\sum_{\alpha=1}^{m} p_\alpha \mathbf{e}_\alpha)^T \sum_{\alpha=1}^{m} p_\alpha \mathbf{e}_\alpha) \\ &= \sum_{\alpha=1}^{m} p_\alpha (1 - p_\alpha) \qquad (3.13) \end{aligned}$$

Thus, the HSEV can be viewed as a generalization of the variance-based measure. When considering the distance of two label points in the embedding space $\mathbf{Z}$, if we set $||\mathbf{z}_\alpha - \mathbf{z}_\beta|| = C_{\alpha\beta}$, the HSEV is equivalent to cost-based measure:

$$\begin{aligned}\mathrm{VOI}(\vec{y}, \mathbf{x}) &= \mathrm{Tr}(\sum_{\alpha=1}^{m} p_\alpha \mathbf{z}_\alpha^T \mathbf{z}_\alpha - (\sum_{\alpha=1}^{m} p_\alpha \mathbf{z}_\alpha)^T \sum_{\beta=1}^{m} p_\beta \mathbf{z}_\beta) \\ &= \sum_{\alpha=1}^{m} \sum_{\beta=1}^{m} ||\mathbf{z}_\alpha - \mathbf{z}_\beta|| p_\alpha p_\beta \\ &= \sum_{\alpha=1}^{m} \sum_{\beta=1}^{m} C_{\alpha\beta} p_\alpha p_\beta\end{aligned}$$

Similarly, if we consider the distance of two label points in the embedding space $\mathbf{Z}$ to be equal to their distance in the transformed label space $\mathbf{V}$ in Eq. 2.4, the HSEV is equivalent to HAS.

4 Batch Mode Hierarchical-Structured Embedded Variance

4.1 Combined Strategy with k-NN Graph-based Diversity The serial-based HSEV suffers from one main limitation: the training process requires the construction of the similarity graph matrix and solving quadratic programming, which is computationally expensive. Despite working well for selecting a single unlabeled instance, it is difficult to extend it to select multiple instances. Here, we present a batch mode Hierarchical-Structured Embedded Variance, denoted by $\mathrm{HSEV_{Dive}}$, by incorporating a diversity to measure the distance between an instance and the selected set. We first propose a graph-based diversity measure and use a graph structure to identify highly connected points. We build a k nearest neighbor graph with $P_{i,j} = 1$ if $d(\mathbf{x_i}, \mathbf{x_j})$ is one of the k smallest distances of $\mathbf{x_i}$ with Manhattan distance. The graph is symmetric and weighted with a Gaussian kernel by variance σ:

$$\mathrm{Q_{ij}} = \mathrm{P_{ij}}\mathrm{exp}(\frac{-\mathrm{d}(\mathbf{x_i}, \mathbf{x_j})}{2\sigma^2}) \tag{4.14}$$

This weight matrix $\mathbf{Q}$ is used to rank all data points according to their diversities. The intuition behind this is that representative data points for one class are usually well embedded in the graph structure and thus have many edges ($>> k$) with high weights. To distinguish among data points with many small weighted neighbors, we normalize these weights by the number of edges:

$$\mathrm{Gra}(\mathbf{x_i}) = \frac{\sum_\mathrm{i} \mathrm{Q_{i,j}}}{\sum_\mathrm{i} \mathrm{P_{i,j}}} \tag{4.15}$$

We reduce the weights of direct neighbors of the currently selected node $\mathbf{x_i}$ with $\mathrm{Gra}(\mathbf{x_i}) = \mathrm{Gra}(\mathbf{x_i}) - \mathrm{Gra}(\mathbf{x_j})\mathrm{P_{i,j}}$. This avoids selecting the same dense region multiple times. In this way, our method avoids sampling of outliers, and is more robust due to the underlying k-NN graph structure.

Finally, we build a convex combination of both uncertainty and diversity measures and proceed to construct a new training batch and incrementally construct a new training batch as shown in Algorithm 1.

Algorithm 1 Batch Mode Hierarchial-Structured Embedded Variance Sampling

```
Input: trade-off $\theta$ between uncertainty and diversity, batch size $h$, unlabeled instance pool $\mathcal{U}$;
Output: $S$: a selected training instances batch.
Init: Set $S = \emptyset$, train the classifier using initial training data;
Repeat
    Compute $\mathrm{VOI}(\mathbf{x_i})$(Eq. 3.12) given instance $\mathbf{x_i}$;
    Compute $\mathrm{Gra}(\mathbf{x_i})$(Eq. 4.15) given instance $\mathbf{x_i}$;
    $\mathbf{x}^* = \arg\max_\mathbf{x} (\theta)\mathrm{VOI}(\mathbf{x}) + (1-\theta)\mathrm{Gra}(\mathbf{x})$;
    $S = S \cup (\mathbf{x}_i)$, $\mathcal{U} = \mathcal{U}; \backslash(\mathbf{x}_i)$;
Until card(S)=h
return $S$.
```

4.2 Computational Complexity The running time of our $\mathrm{HSEV_{Dive}}$ algorithm depends on the active learners used as subroutines (model training, variance measure, or diversity sampling). Summing up training time and sampling time, the sequential mode requires computational time of order $O(T_{train} + T_{voi})$, where $\mathrm{T_{train}}$ is the time to run model training and $\mathrm{T_{voi}}$ is the time to run the variance-based measure. This time complexity is dominated by the training time part $\mathrm{T_{train}}$ as it is larger than $\mathrm{T_{voi}}$. In comparison to the distance strategy, the combined strategy requires additional computational time of order $O(T_{div})$, where $\mathrm{T_{div}}$ is the time to run diversity sampling ($\mathrm{T_{div}}$ is much smaller than $\mathrm{T_{voi}}$). Specifically, the expected time complexity of the $\mathrm{HSEV_{Dive}}$ is $O(\frac{T_\mathrm{train}}{h} + T_\mathrm{voi} + T_\mathrm{div})$ with batch size h. As a result, the overall time complexity of the $\mathrm{HSEV_{Dive}}$ gets decreased as h increases.

4.3 Parameters Setting The hierarchical SVM [18, 19] is employed in this paper as the default classifier. The hierarchical SVM algorithm travels from the root until it reaches a leaf node, and at each node follows the child that has the largest classifier score. In this case, the label at the final leaf node is output as the classification label. We use a standard SVM with RBF kernels. The proposed framework has some parameters

Table 1: Some statistics of Hglass, RCV1-v2 and GPCR-Interpro.

Dataset	$\|y\|$	$\|\iota\|$	$\|n\|$	Train	Test
Hglass	27	20	9	150	64
RCV1-v2	101	77	47236	600	600
GPCR	201	199	450	2444	5000

and we did not tune our parameters to match the test datasets. The embedding dimension d was set to be half of the number of the leaf labels. The hyper-parameters of SVM classifier were set at standard values. How to set parameters k_1 and k_2 is still an open problem [14]. Therefore, we empirically set the two parameters. Specifically, we sampled five values of k_1 between 4 and $(min\{N_\alpha\} - 1)$ and chose the value of the best performance, where N_α is the number of labeled data with label α. Similarly, we chose the best k_2 by sampling between 10 and $4N_\alpha$. The question of how to choose an optimal value for θ depends on the training process and the datasets. We empirically set it to be 0.7 in our experiments.

5 Experimental Study

5.1 Experimental Setup We evaluate our proposed approach on three real-world datasets: (1) Hglass [20]: a hierarchical classification dataset that contains 214 data points and 20 leaf labels, which uses attributes as features; (2) RCV1-v2 [21]: a widely used benchmark dataset. The documents have been tokenized, stopped and stemmed to 47,236 unique tokens (features) and represented as L2-normalized log TF-IDF vectors; (3) GPCR-Interpro [22]: a protein function dataset whose features are 0/1 "protein signature".

The details of the datasets are described in Table 1, where $|y|$ is the number of all nodes in the class hierarchy, $|\iota|$ is the number of leaf labels, and $|n|$ is the number of features. We employ two evaluation criteria to evaluate the performance. The first is average accuracy. The second is tree-loss [23], which is the sum of the misclassification cost on the test data.

5.2 Sequential Mode Active Learning Comparison We first compare the sequential HSEV to three baselines: 1) entropy-based uncertainty sampling: denoted as ENTROPY; 2) cost-based uncertainty sampling: denoted as COST; 3) Hierarchical Active Sampling, the method proposed in [5], denoted as HAS.

In Figure 2 and 3, we show the average accuracy and tree-loss for the four competing methods at each learning step. Our HSEV achieves the best accuracy among all the methods. The HAS and COST methods get similar performances and their results are better than ENTROPY. Classification on Hglass data is a tough task since the training data is limited and the total number of features is very small. Both HAS and COST perform poorly on this dataset while the proposed HSEV performs well even when only a small amount of data is labeled. This illustrates that our proposed measure is very effective even when the training data is very scarce. From the experimental results on the real-world datasets, we show that the Sequential HSEV does improve the learning rate and the classification performance compared to all the baseline methods.

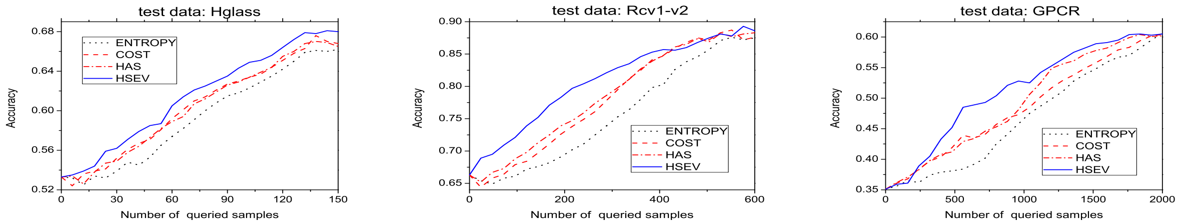
5.3 Batch Model Active Learning Comparison In this experiment, we compare the proposed batch mode HSEV approach, $\text{HSEV}_{\text{Dive}}$ to the following algorithms: 1) Batch mode SVM active learning (SVM_{Dive}): a heuristic modification of SVM active learning by incorporating diversity in the batch sampling procedure [11]. 2) Batch Mode Hierarchical Active Sampling (HAS_{Dive}): a modification of the Hierarchical Structure Sampling by incorporating diversity in the batch sampling procedure. 3) The Sequential Mode Hierarchical-Structured Embedded Variance approach (HSEV): selecting top-h samples with high uncertainty scores.

We conduct experiments with the four active learning approaches by varying the batch size after 30% of the training data is labeled. Tables 2, 3 and 4 show the average accuracy and tree-loss performance on the three datasets separately. "MAR" stands for the mean of average results (accuracy or tree-loss). The proposed algorithm consistently outperforms the other three approaches with significant improvement. The performances of sequential mode HSEV drops when the batch size increases. In contrast, the relative improvement in $\text{HSEV}_{\text{Dive}}$ tends to become more significant and this performance increases when the batch size is increased. These results show that the proposed batch mode active learning algorithm is more effective in selecting a batch of informative unlabeled samples for labeling.

5.4 Sensitivity Analysis of Embedding Dimension We now study how the choice of the embedding dimension d affects the performance. With fixed active learning settings, we vary d, to evaluate the robustness of our method after 30% and 60% of training data are labeled. Since the number of attributes in Hglass is small, the test is only performed on RCV1-v2 and GPCR-Interpro datasets, with d varying from $\frac{|\iota|}{4}$ to $|\iota|$. As shown in Figure 4a and 4b, as d increases from 20 to 80, the performance of both accuracy and tree-loss do

Table 2: The average accuracy and tree-loss performances with different batch sizes on the Hglass.

	HSEV$_{\text{Dive}}$		SVM$_{\text{Dive}}$		HAS$_{\text{Dive}}$		HSEV	
Batch Size	accuracy	tree-loss	accuracy	tree-loss	accuracy	tree-loss	accuracy	tree-loss
5	0.522	1.613	0.425	1.727	0.483	1.515	0.529	1.625
10	0.537	1.611	0.423	1.725	0.505	1.624	0.505	1.641
15	0.549	1.597	0.431	1.715	0.522	1.635	0.493	1.629
20	0.556	1.523	0.475	1.678	0.525	1.641	0.485	1.662
30	0.576	1.475	0.513	1.635	0.531	1.649	0.472	1.675
MAR	0.548	1.553	0.465	1.687	0.513	1.626	0.486	1.645



(a)

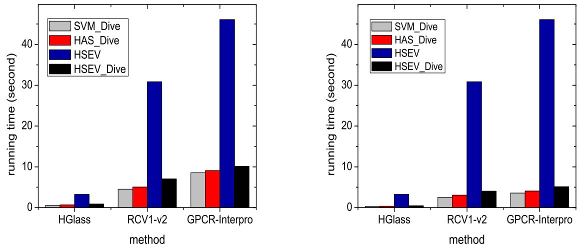
(b)

(c)

Figure 2: Accuracy comparison on different datasets.

not change too much. Similar trends can be observed in Figure 4c and 4d, with d varying from 50 to 200. It is concluded that the trends in accuracy and tree-loss remain almost the same regardless of the choice of d in the range from $\frac{|\iota|}{4}$ to $|\iota|$.

5.5 Running Time Comparison We measure the efficiency of our implementation on the three experimental datasets. Our C++ implementation runs on a dual-core 3.3 GHz machine with 8 G memory. We compare it with the running time of SVM$_{\text{Dive}}$, HAS$_{\text{Dive}}$ and sequential HSEV. Figure 5a and 5b show the result of average running time for selecting one sample while batch size is 5 and 10, separately. Although the complexity of our algorithm is higher than SVM$_{\text{Dive}}$ and HAS$_{\text{Dive}}$, our method achieves comparable running time with the two baselines. Compared with the sequential HSEV method, our method is more efficient, which is consistent with the analysis presented in Subsection 4.2.



(a) Batch size is 5.

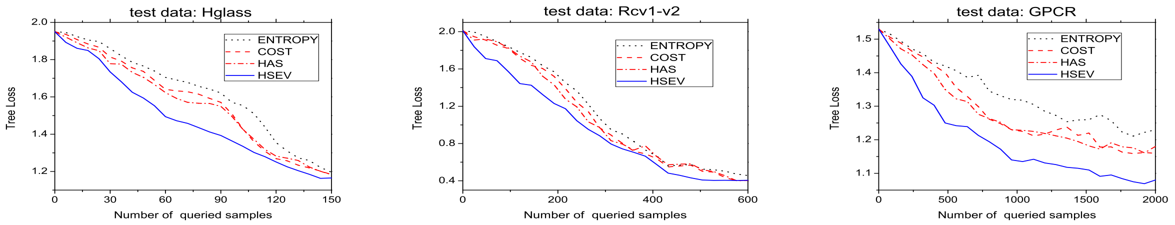
(b) Batch size is 10.

Figure 5: Running time of 4 algorithms on different datasets.

6 Conclusion

We proposed sequential and batch mode active learning schemes for multi-class hierarchical classification. In particular, we presented a unified learning framework for incorporating the hierarchical structure of the label tree as well as the characteristics of the training data to select the most informative data for human labeling, and developed a new batch mode active learning algorithm based on diversity measure. Experimental results have demonstrated that the proposed sequential and batch mode methods can improve the learning rate and performance over the state-of-the-art methods. The results indicate that utilizing both the hierarchical structure of the category labels and information of the training data are very helpful for active learning.

There are several interesting research directions for future work. First, although the combined selection strategy is fairly robust with respect to the trade-off parameter θ, the question of how to choose an optimal value for θ is yet unanswered. Second, the proposed

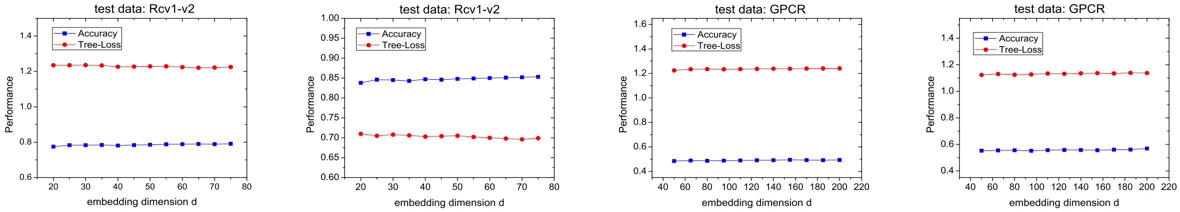


(a)

(b)

(c)

Figure 3: Tree-loss comparison on different datasets.



(a) Rcv1-v2, 30% labeled.

(b) Rcv1-v2, 60% labeled.

(c) GPCR, 30% labeled.

(d) GPCR, 60% labeled.

Figure 4: Sensitivity analysis of d on Rcv1-v2 and GPCR-Interpro datasets.

Table 3: The average accuracy and tree-loss performances with different batch sizes on the Rcv1-v2.

	$\text{HSEV}_{\text{Dive}}$		SVM_{Dive}		HAS_{Dive}		HSEV	
Batch Size	accuracy	tree loss	accuracy	tree loss	accuracy	tree loss	accuracy	tree loss
5	0.752	1.363	0.623	1.427	0.683	1.415	0.725	1.472
10	0.757	1.311	0.625	1.425	0.703	1.362	0.704	1.524
15	0.774	1.259	0.622	1.415	0.721	1.335	0.692	1.563
20	0.796	1.223	0.625	1.378	0.754	1.321	0.686	1.568
30	0.805	1.215	0.631	1.335	0.773	1.294	0.675	1.605
MAR	0.772	1.261	0.623	1.387	0.723	1.315	0.692	1.569

Table 4: The average accuracy and tree-loss performances with different batch sizes on the GPCR-Interpro.

	$\text{HSEV}_{\text{Dive}}$		SVM_{Dive}		HAS_{Dive}		HSEV	
Batch Size	accuracy	tree loss	accuracy	tree loss	accuracy	tree loss	accuracy	tree loss
5	0.462	1.261	0.385	1.342	0.423	1.256	0.455	1.255
10	0.507	1.211	0.393	1.325	0.457	1.224	0.445	1.281
15	0.519	1.197	0.411	1.341	0.482	1.225	0.413	1.296
20	0.535	1.163	0.475	1.274	0.495	1.192	0.418	1.306
30	0.546	1.135	0.513	1.235	0.501	1.168	0.372	1.357
MAR	0.522	1.191	0.465	1.272	0.473	1.191	0.429	1.295

method provides a general platform to develop new algorithms for active learning with embedding label relationships. It would also be very interesting to consider active learning problems with more complicated label structures [24, 25] (such as directed graphs).

References

[1] B. Settles, "Active learning literature survey," Tech. Rep., 2010.

[2] P. Jain and A. Kapoor, "Active learning for large multiclass problems," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 762–769, 2009.

[3] A. J. Joshi, F. Porikli, and N. Papanikolopoulos, "Multi-class active learning for image classification," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009, pp. 2372–2379.

[4] T. Gao and D. Koller, "Active classification based on value of classifier," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., 2011, pp. 1062–1070.

[5] Y. Cheng, Z. Kunpeng, Y. Xie, A. Agrawal, and A. Choudhary, "On active learning in hierarchical classification," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*. ACM, 2012, pp. 2467–2470.

[6] K. Weinberger and O. Chapelle, "Large margin taxonomy embedding for document categorization," in *Advances in Neural Information Processing Systems*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., 2009, pp. 1737–1744.

[7] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Springer-Verlag New York, Inc., 1994, pp. 3–12.

[8] H. Raghavan, O. Madani, and R. Jones, "Active learning with feedback on features and instances," *J. Mach. Learn. Res.*, vol. 7, pp. 1655–1686, Dec. 2006.

[9] E. M. N. Ebrahimi, "Measuring informativeness of data by entropy and variance," *Slottje(ed.)*, 1999.

[10] S. C. H. Hoi, R. Jin, J. Zhu, and M. R. Lyu, "Batch mode active learning and its application to medical image classification," in *Proceedings of the 23rd International Conference on Machine learning*, ser. ICML '06. New York, NY, USA: ACM, 2006, pp. 417–424.

[11] K. Brinker, "Incorporating diversity in active learning with support vector machines," in *In Proceedings of the 20th International Conference on Machine Learning*. AAAI Press, 2003, pp. 59–66.

[12] Z. Xu, R. Akella, and Y. Zhang, "Incorporating diversity and density in active learning for relevance feedback," in *Proceedings of the 29th European Conference on IR Research*. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 246–257.

[13] S. Bengio, J. Weston, and D. Grangier, "Label embedding trees for large multi-class tasks," in *Advances in Neural Information Processing Systems 23*, J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., 2010, pp. 163–171.

[14] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 40–51, Jan. 2007.

[15] S. Bengio, J. Weston, and D. Grangier, "Label embedding trees for large multi-class tasks." in *Neural Information Processing Systems*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds., 2010, pp. 163–171.

[16] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York: Springer, 2006.

[17] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[18] L. Cai and T. Hofmann, "Hierarchical document categorization with support vector machines," in *Proceedings of the thirteenth ACM International Conference on Information and Knowledge Management*. New York, NY, USA: ACM, 2004, pp. 78–87.

[19] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *J. Mach. Learn. Res.*, vol. 2, pp. 45–66, Mar. 2002.

[20] J. Metz, M. C. Monard, and E. A. Cherman, "A study on the selection of local training sets for hierarchical classification tasks," in *ENIA 2011: Encontro Nacional de Inteligncia Artificial*. Natal, RN, Brasil: Sociedade Brasileira da Computao - SBC, 2011, pp. 572–583.

[21] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "Rcv1: A new benchmark collection for text categorization research," *J. Mach. Learn. Res.*, vol. 5, pp. 361–397, Dec. 2004.

[22] C. N. Silla Jr. and A. A. Freitas, "A global-model naive bayes approach to the hierarchical prediction of protein functions," in *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, ser. ICDM '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 992–997.

[23] O. Dekel, J. Keshet, and Y. Singer, "Large margin hierarchical classification," in *Proceedings of the Twenty-First International Conference on Machine learning*, ser. ICML '04. ACM, 2004, pp. 27–35.

[24] Z. Chen, Y. Xie, Y. Cheng, K. Zhang, A. Agrawal, W. keng Liao, N. F. Samatova, and A. N. Choudhary, "Forecast oriented classification of spatio-temporal extreme events," in *IJCAI*, 2013.

[25] Y. Cheng, Y. Xie, K. Zhang, A. Agrawal, and A. Choudhary, "Cluchunk: clustering large scale user-generated content incorporating chunklet information," ser. BigMine '12, 2012, pp. 12–19.