BatchRank: A Novel Batch Mode Active Learning Framework for Hierarchical Classification

Shayok Chakraborty[1], Vineeth Balasubramanian[2], Adepu Ravi Sankar[2], Sethuraman Panchanathan[3] and Jieping Ye[4]
[1]Electrical and Computer Engineering, Carnegie Mellon University
[2]Department of Computer Science and Engineering, Indian Institute of Technology, Hyderabad
[3]School of Computing, Informatics and Decision Systems Engineering, Arizona State University, AZ
[4]Department of Computational Medicine and Bioinformatics and Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor

ABSTRACT

Active learning algorithms automatically identify the salient and exemplar instances from large amounts of unlabeled data and thus reduce human annotation effort in inducing a classification model. More recently, Batch Mode Active Learning (BMAL) techniques have been proposed, where a batch of data samples is selected simultaneously from an unlabeled set. Most active learning algorithms assume a flat label space, that is, they consider the class labels to be independent. However, in many applications, the set of class labels are organized in a hierarchical tree structure, with the leaf nodes as outputs and the internal nodes as clusters of outputs at multiple levels of granularity. In this paper, we propose a novel BMAL algorithm (BatchRank) for hierarchical classification. The sample selection is posed as an NP-hard integer quadratic programming problem and a convex relaxation (based on linear programming) is derived, whose solution is further improved by an iterative truncated power method. Finally, a deterministic bound is established on the quality of the solution. Our empirical results on several challenging, real-world datasets from multiple domains, corroborate the potential of the proposed framework for real-world hierarchical classification applications.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; I.5.4 [**Pattern Recognition**]: Applications

General Terms

Algorithms

Keywords

Active Learning, Hierarchical Classification, Optimization

KDD'15 August 11 - 14, 2015, Sydney, NSW, Australia

DOI: http://dx.doi.org/10.1145/2783258.2783298.

1. INTRODUCTION

Active learning algorithms have recently gained popularity to reduce human annotation effort in training a classification/regression model. When exposed to large amounts of unlabeled data, such algorithms automatically identify the informative samples for manual labeling. Of late, there have been research attempts towards batch mode active learning (BMAL), where a batch of unlabeled samples is selected simultaneously for manual annotation.

Most BMAL algorithms have been developed for a flat label space, that is, they treat all the category labels independently [14, 11]. However, in many applications, the class labels are organized in the form of a tree hierarchy, where the leaf nodes contain the output classes and the internal nodes denote a super-set of the outputs at different levels of granularity. For instance, text documents are often represented as a hierarchy of classes, based on their contents [16, 21]; medical images can be annotated using a class hierarchy for efficient classification [7]. Due to the tremendous increase in the generation of digital data, taxonomies and hierarchies are becoming increasing popular to adequately organize and interpret them. Thus, there is a pronounced need for an active batch selection framework in the context of hierarchical classification.

In this paper, we propose a novel BMAL algorithm, which exploits the hierarchical structure of the label tree to select the informative unlabeled samples for manual annotation. The sample selection is expressed as an NP-hard integer quadratic programming (IQP) problem. We then derive a convex relaxation, based on linear programming, and show that the batch selection task reduces to a score ranking problem (hence the name BatchRank). Finally, a deterministic bound is derived on the quality of the solution, obtained using the relaxation. To the best of our knowledge, this is the first research effort to derive a concrete mathematical guarantee on the solution quality of batch mode active learning in hierarchical classification. We note here that the purpose of this work is not to derive a bound on the number of queries required to achieve a given generalization error in active learning. This problem has been extensively studied in the literature [12, 1]. Our objective is to derive a mathematical guarantee on the solution quality of the convex relaxation of BMAL in hierarchical classification, which, to the best of our knowledge, has not been addressed till date.

2. RELATED WORK

We start with a survey of active learning in general and then present the few existing work in active learning for hierarchical classification.

Active Learning: Active learning is a well-studied problem in the machine learning literature. Several techniques have been developed over the last few years and a review of these can be found in [23]. In a typical pool-based active learning setting, the learner is exposed to a pool of unlabeled samples and it iteratively selects samples for manual annotation. Pool-based active learning is further classified into *single instance selection* (where a single unlabeled sample is selected in each iteration) and *batch selection* (where a batch of samples is selected simultaneously and is effective in utilizing the presence of multiple labeling oracles). Since the focus of this research is on batch mode active learning (BMAL), we present existing work in this field below.

Initial BMAL techniques were largely based on extending the pool-based approaches to the batch setting using greedy heuristics [2, 22]. More recently, optimization based strategies have been proposed which have been shown to outperform the heuristic approaches. Hoi *et al.* [15, 13] used the Fisher information matrix as a measure of model uncertainty and proposed to query the set of points that maximally reduced the Fisher information. The same authors [14] proposed a BMAL scheme based on SVMs where a kernel function was first learned from a mixture of labeled and unlabeled samples, which was then used to identify the informative and diverse examples through a min-max framework. Guo and Schuurmans [11] proposed a discriminative strategy that selected a batch of points which maximized the log-likelihoods of the selected points with respect to their optimistically assigned class labels and minimized the entropy of the unselected points in the unlabeled pool. Recently, Guo [10] proposed a batch mode active learning scheme which maximized the mutual information between the labeled and unlabeled sets and was independent of the classification model.

Active Learning in Hierarchical Classification: Owing to the explosive growth in the amount of digital data, hierarchies are becoming increasingly popular to efficiently organize and categorize data. This has led to the development of several hierarchical classification algorithms. Such algorithms usually associate a vector w_i with each node i in the label tree and can be broadly categorized into two groups: *recursive* and *non-recursive*. The recursive algorithms start with the root node and label a data sample sequentially by selecting the category for which the associated vector outputs the largest score among its siblings, until a leaf node is reached [27, 8]. Specifically, given a training set $\{(x_1, y_1), (x_2, y_2), \ldots (x_N, y_N)\}$, a vector $w_i \in \Re^n$ is associated with each node $i \in Y$, where Y is the set of all labels in the tree. Let $C(i)$ denote the set of all children of node i. The recursive procedure uses classifiers $f(x)$ that are parameterized by $w_1, w_2, \ldots w_m$ through the following recursive procedure:

$$f(x) = \begin{cases} \textbf{initialize} : i = 0 \\ \textbf{while} \quad \textbf{C(i)} \quad \textbf{is} \quad \textbf{not} \quad \textbf{empty} \\ \quad i = \arg\max_{j \in C(i)} w_j^T x \\ \textbf{return}\ i \end{cases} \tag{1}$$

Non-recursive classifiers have also been proposed in the context of hierarchical classification [3, 6]. A popular approach is:

$$f(x) = \arg\max_{i \in Y} w_i^T x \tag{2}$$

Even though hierarchical classification has been extensively studied, active learning in hierarchical classification is less explored. Researchers have begun to study this problem only recently (in the last 2-3 years). Li *et al.* [18, 19] used uncertainty sampling to select informative samples for active learning in hierarchical classification, in the context of text mining. The fundamental idea of their approach was to intelligently subdivide the unlabeled pool so as to minimize the so-called out-of-domain queries. Cheng *et al.*[5] proposed a variance based uncertainty measure to query a set of informative unlabeled samples. The fundamental idea was to identify a continuous semantic space underlying the hierarchical structure. All the labels in the category tree were then embedded into the latent semantic space. The variance was computed by considering each label as a point and the uncertainty was computed using this variance. The same authors also introduced an active batch selection framework in the hierarchical setting using a diversity criterion together with the uncertainty measure [4]. To compute the diversity, a graph structure was used to identify the highly connected unlabeled points. A k nearest neighbor graph was built for every unlabeled point, which was weighted by a Gaussian kernel; the weighted matrix was used to rank all the unlabeled samples according to their diversities. In each iteration, the unlabeled sample furnishing the maximum weighted uncertainty and diversity score was selected for manual annotation. This algorithm, Hierarchical-Structured Embedded Variance (HSEV) was shown to outperform the previous approaches and is the state-of-the-art. Even though they exhibit good empirical performance, these techniques are heuristic in nature and also lack any quantitative guarantees on the solution quality.

In this paper, we present a novel batch mode active learning framework for hierarchical classification. Our algorithm has a concrete mathematical foundation and also provides strong theoretical guarantees on the quality of the obtained solution. We now describe the proposed framework.

3. PROPOSED FRAMEWORK

The recursive classifiers of the form in Equation (1) have been shown to outperform the non-recursive counterparts both in terms of accuracy and computational efficiency, in hierarchical classification [27]. We therefore use a recursive model as the underlying classifier in our framework.

3.1 Problem Set-up

Consider a batch mode active learning problem, where we are given a training set L_t and an unlabeled set U_t at time t. Let w^t be the classifier trained on L_t and let Y denote the set of all labels in the hierarchical label tree of depth d. The objective is to select a batch B (containing k points) from U_t in such a way that the future learner w^{t+1}, trained on $L_t \bigcup B$, has maximum generalization capability.

Batch Selection Criterion: We quantify the quality of a batch of selected samples based on their informativeness and diversity, that is, each point in the selected batch should furnish valuable information and the selected samples should have minimal redundancy among them.

Formally, we compute an information vector $c \in \Re^{|U_t| \times 1}$ where $c(i)$ denotes the information furnished by the point x_i in the unlabeled pool. The uncertainty of the trained model on a given sample x_i is used as a measure of information of that sample; higher uncertainty values denote higher degrees of information. Let y_l denote the set of labels in level l of the label tree. The uncertainty of an unlabeled sample x_i at level l is given by the entropy $S(y_l|x_i)$ of the distribution $P(y_l|x_i)$, such that:

$$S(y_l|x_i) = -\sum_{y \in y_l} P(y|x_i) \log P(y|x_i) \tag{3}$$

The posterior probability at a given node is obtained from the corresponding classification model trained at the node. The aggregate uncertainty of an unlabeled sample x_i is computed as the summation of the entropy values at all levels in the label tree.

$$c(i) = \sum_{l=1}^{d} S(y_l|x_i) \tag{4}$$

Intuitively, the uncertainty of an unlabeled sample is computed as the sum of the entropy values at each level while classifying the sample using a recursive classifier.

Further, to maximize the contribution of the selected unlabeled samples, diversity based criteria have been proposed [24] which ensure that the selected samples have maximal divergence (minimal redundancy) among them. To this end, we compute a divergence matrix $R \in \Re^{|U_t| \times |U_t|}$, whose $(i,j)^{th}$ entry is a measure of redundancy between unlabeled points x_i and x_j (higher the value of R_{ij}, lower the redundancy). In our formulation, we compute the diversity between a pair of unlabeled samples as their kernelized distance. Then, the $(i,j)^{th}$ entry in the matrix R is given by:

$$R(i,j) = \phi(x_i, x_j) \tag{5}$$

Computational Considerations: Let $A \in \Re^{n \times p}$ denote the matrix of n unlabeled samples, each with dimension p. The redundancy matrix R involves computation of the pairwise distance matrix AA^T, which has a complexity of $O(n^2p)$. This may be expensive (and often prohibitive) for large n and p, which are frequently encountered in modern applications.

We first note that the matrix R needs to be computed just once in our framework. Once we have the pairwise distance matrix of all the unlabeled samples, we can keep deleting the corresponding rows and columns from the distance matrix (as batches of unlabeled samples are selected for annotation) to get an updated distance matrix for the new set of unlabeled samples. This matrix can be computed offline before the active learning iterations commence. Further, we use the concept of random projections to reduce the computational overhead. Random projections have been successfully used to speed up computations, where the original data matrix $A \in \Re^{n \times p}$ is multiplied by a random projection matrix $X \in \Re^{p \times d}$ to obtain a projected matrix $B \in \Re^{n \times d}$ in the lower dimensional space d:

$$B = \frac{1}{\sqrt{d}} AX \tag{6}$$

where $d << min(n,p)$. X is typically populated using the entries from the standard normal distribution $N(0,1)$, leading to many well-known theoretical results [25]. In our work, we adopt the idea of *Very Sparse Random Projections* proposed by Li *et al.* [17]. The random matrix X is computed as:

$$X(j,i) = \sqrt{s} \begin{cases} 1, & \text{with prob } \frac{1}{2s} \\ 0, & \text{with prob } 1 - \frac{1}{s} \\ -1 & \text{with prob } \frac{1}{2s} \end{cases} \tag{7}$$

We use $s = \sqrt{p}$ to significantly speed up the computation, by a factor of $\sqrt{p}$ or more [17]. Let $\{u_i\}_{i=1}^n \in \Re^p$ denote the rows in the original data matrix A and $\{v_i\}_{i=1}^n \in \Re^d$ be the rows in the projected data matrix B, such that $v_i = \frac{1}{\sqrt{d}} X^T u_i$. Also, let u_1, u_2, v_1, v_2 denote the two leading rows respectively. Then, it can be proved that [17]:

$$E(||v_1 - v_2||^2) = ||u_1 - u_2||^2 \tag{8}$$

Very sparse random projections therefore preserve pairwise 2-norm distances in expectations while offering significant computational speed-ups. Please refer [17] for more results and detailed derivations.

Active Batch Selection Framework: By definition, all the entries in c and R are non-negative, that is, $c_i \geq 0$ and $r_{ij} \geq 0$. Also, $R_{ii} = 0, \forall i$. Given c and R, our objective is to select a batch of points having high information scores and high divergence (or minimal redundancy) among them. For notational simplicity, we combine c and R into a single matrix $D \in \Re^{|U_t| \times |U_t|}$ as follows:

$$D(i,j) = \begin{cases} R(i,j), & \text{if } i \neq j \\ \lambda c(i), & \text{if } i = j \end{cases} \tag{9}$$

where each entry in the matrix D is non-negative, that is $d_{ij} \geq 0, \forall i,j$. λ is a trade-off parameter. We note that the matrix D can be defined suitably based on the application at hand. Without any loss of generality, we proceed with the criterion based on entropy and diversity and explain our framework.

We now formulate the batch selection task as an explicit mathematical optimization problem, where the objective is to select a batch of points with high aggregate uncertainty scores and high divergences among them. Specifically, we define a binary vector m with $|U_t|$ entries ($m \in \{0,1\}^{|U_t| \times 1}$) where each entry m_i denotes whether the corresponding unlabeled point x_i will be included in the batch ($m_i = 1$) or not ($m_i = 0$). Our batch selection criterion (with given batch size k) can thus be expressed using the following integer quadratic programming (IQP) problem:

$$\max_m m^T D m$$

$$\text{s.t.} \quad m_i \in \{0,1\}, \forall i \quad \text{and} \quad \sum_{i=1}^{|U_t|} m_i = k \tag{10}$$

The binary constraint on m_i makes this IQP problem NP-hard. We now discuss an efficient relaxation to solve this NP-hard problem [1].

3.2 An Efficient Convex Relaxation

We first show that the IQP in Equation (10) is equivalent to an Integer Linear Programming (ILP) problem.

[1]The problem is presumed to be NP-hard as D is just a symmetric matrix, without any other special properties. A formal proof will be taken up as part of future research.

LEMMA 1. *The Integer Quadratic Programming batch mode active learning formulation in Equation (10) can be simplified into an equivalent Integer Linear Programming (ILP) problem.*

PROOF. We introduce a binary matrix $Z = (z_{ij})$ with $z_{ij} = m_i.m_j$. Thus, the optimization problem in (10) reduces to:

$$\max_{m,Z} \sum_{i,j} d_{ij} z_{ij}$$

$$\text{s.t.} \quad z_{ij} = m_i m_j, \quad \sum_{i=1}^{|U_t|} m_i = k, \quad \textit{and} \quad m_i \in \{0,1\}, \forall i \quad (11)$$

The quadratic equality constraint $z_{ij} = m_i m_j$ makes this problem difficult to solve. We can show that this quadratic constraint, in fact, allows itself to be represented as a simpler linear inequality $-m_i - m_j + 2z_{ij} \leq 0, \forall i, j$. This ensures that the value of z_{ij} is 0 if either m_i or m_j (or both) is equal to 0. When both m_i and m_j are equal to 1, z_{ij} is free to be either 0 or 1. However, the maximization criterion in (11) forces the value of z_{ij} to be 1 since $d_{ij} \geq 0$. Hence, the problem can now be written as:

$$\max_{m,Z} \sum_{i,j} d_{ij} z_{ij}$$

$$\text{s.t.} \quad -m_i - m_j + 2z_{ij} \leq 0, \forall i, j$$

$$\textit{and} \quad \sum_{i=1}^{|U_t|} m_i = k, m_i, z_{ij} \in \{0,1\}, \forall i, j \quad (12)$$

This is an integer LP problem, proving Lemma 1. □

Although a global maximum exists for the ILP, it is computationally expensive to compute. To solve such an ILP, a standard approach is to employ the LP relaxation.

LEMMA 2. *The convex LP relaxation of the above ILP (Equation 12) in Lemma* 1 *is equivalent to a ranking formulation based on the entries in the matrix D.*

PROOF. We consider the following linear program relaxation:

$$\max_{m,Z} \sum_{i,j} d_{ij} z_{ij}$$

$$\text{s.t.} \quad -m_i - m_j + 2z_{ij} \leq 0, \forall i, j, \quad \sum_{i=1}^{|U_t|} m_i = k$$

$$\textit{and} \quad m_i, z_{ij} \in [0,1], \forall i, j \quad (13)$$

Since this is a maximization problem with $d_{ij} \geq 0$, at optimality, $z_{ij} = \frac{m_i + m_j}{2}$ (from the inequality constraint $-m_i - m_j + 2z_{ij} \leq 0$). Hence, (13) is equivalent to:

$$\max_{m} \frac{1}{2} \sum_{i,j} d_{ij}(m_i + m_j)$$

$$\text{s.t.} \quad \sum_{i=1}^{|U_t|} m_i = k \quad \textit{and} \quad m_i \in [0,1], \forall i \quad (14)$$

This formulation admits an analytical (as well as an integer) solution for m by a simple ranking based on the entries in the matrix D. The objective in (14) can be written as $\sum_{i,j} d_{ij} m_i + \sum_{i,j} d_{ij} m_j$. Since the matrix D is symmetric, the maximization problem essentially becomes equivalent to ranking the column sums of D (hence the name BatchRank). This proves Lemma 2. □

3.3 Solution Bound of BatchRank

In this section, we prove a bound on the solution to the convex LP relaxation in (14) with respect to the solution of the original NP-hard integer quadratic programming problem. To this end, we transform the original maximization problem in Equation (10) into an equivalent minimization problem through the following objective function:

$$f(m) = ||D||_1 - m^T D m \quad (15)$$

where $||D||_1 = \sum_{i,j} d_{ij}$. We note that since $||D||_1$ is constant for a given matrix D, maximizing $m^T D m$ as in Equation (10) is equivalent to minimizing the function $f(.)$ defined above, that is, maximizing $m^T D m$ and minimizing $f(.)$ as defined above will fetch the same solution to the variable m. Since we are interested in the solution quality of m, we prove an upper bound on the minimization problem in Equation (15). Since the solution to m is the same, it is essentially equivalent to proving a bound on the original maximization problem in Equation (10). The main result is summarized in the following theorem:

THEOREM 1. *Let m^* and $\widehat{m}$ be the optimal solutions of the original NP-hard IQP in Equation (10) and the convex relaxation in Equation (14) respectively. Then,*

$$f(\widehat{m}) \leq 2 f(m^*)$$

PROOF. The optimization in (14) is an LP relaxation of the quadratic formulation in (10) and thus the objective value of (14) is larger than that of (10). That is,

$$m^{*T} D m^* \leq \frac{1}{2} \sum_{i,j} d_{ij}(\widehat{m_i} + \widehat{m_j})$$

$$= \frac{1}{2} \sum_{i,j:\widehat{m_i}+\widehat{m_j}=1} d_{ij} + \sum_{i,j:\widehat{m_i}+\widehat{m_j}=2} d_{ij} \quad (16)$$

Since all entries in D are non-negative, the following holds:

$$\begin{aligned} ||D||_1 &= \sum_{ij} d_{ij} \\ &= \sum_{i,j:\widehat{m_i}+\widehat{m_j}=2} d_{ij} + \sum_{i,j:\widehat{m_i}+\widehat{m_j}=1} d_{ij} + \sum_{i,j:\widehat{m_i}+\widehat{m_j}=0} d_{ij} \\ &\geq \sum_{i,j:\widehat{m_i}+\widehat{m_j}=2} d_{ij} + \sum_{i,j:\widehat{m_i}+\widehat{m_j}=1} d_{ij} \end{aligned}$$

Thus,

$$\sum_{i,j:\widehat{m_i}+\widehat{m_j}=1} d_{ij} \leq ||D||_1 - \sum_{i,j:\widehat{m_i}+\widehat{m_j}=2} d_{ij} \quad (17)$$

Combining the above two, we have

$$
\begin{aligned}
f(m^*) &= ||D||_1 - m^{*T} D m^* \\
&\geq ||D||_1 - \frac{1}{2} \sum_{i,j:\widehat{m_i}+\widehat{m_j}=1} d_{ij} - \sum_{i,j:\widehat{m_i}+\widehat{m_j}=2} d_{ij} \\
&\geq ||D||_1 - \frac{1}{2} \left(||D||_1 - \sum_{i,j:\widehat{m_i}+\widehat{m_j}=2} d_{ij} \right) \\
&- \sum_{i,j:\widehat{m_i}+\widehat{m_j}=2} d_{ij} \\
&= \frac{1}{2} \left(||D||_1 - \sum_{i,j:\widehat{m_i}+\widehat{m_j}=2} d_{ij} \right) = \frac{1}{2} f(\widehat{m})
\end{aligned}
$$

The first inequality follows from Equation (16) and the second from Equation (17). The last equality is true because in the evaluation of $m^T Dm$, only the indices where both m_i and m_j are 1 will survive, others will vanish. This completes the proof of the theorem. We thus note that the convex relaxation of the original NP-hard problem in BatchRank has a guaranteed bound on the solution quality. □

3.4 The Iterative Truncated Power Algorithm

As evident from Lemma 2, the LP relaxation of the NP-hard IQP in Equation (10) reduces to selecting a set of k unlabeled points producing the k largest column sums of the matrix D. To further improve the solution, we use the iterative truncated-power algorithm proposed by Yuan and Zhang [26]. This solution was proposed in the context of the sparse eigenvalue problem and was also shown to be applicable to the densest k-subgraph problem (DkS). Mathematically, DkS can be expressed as a binary quadratic programming problem, equivalent to Equation (10). Starting with an initial approximation x_0, the algorithm generates a sequence of solutions $x_1, x_2, \ldots$. At each time step t, the vector x_{t-1} is multiplied by the weight matrix D and then the entries are truncated to zeros except for the k largest entries, which becomes the new solution x_t. This process is repeated until convergence. This simple, yet efficient algorithm has a guaranteed monotonic convergence for a positive semi-definite weight matrix D. When the matrix D is not psd, the algorithm can be run on the shifted quadratic function (with a positive scalar added to the diagonal elements) to guarantee a monotonic convergence [26].

We use the BatchRank solution as the initial approximation x_0 followed by the iterative truncated power method to derive the final set of unlabeled samples to be selected for manual annotation. Since the convergence is monotonic, the quality of the solution can only improve over the iterations and the bound established in Theorem 1 on the quality of the solution still holds. Moreover, the running time increases only marginally due to the iterative process as it involves minimal computational overhead and converges fast. The pseudo-code for the BatchRank algorithm is given in Algorithm 1. The complexity of the algorithm is $O(n^2)$, where n is the number of unlabeled samples.

Algorithm 1 BatchRank algorithm for Batch Mode Active Learning in Hierarchical Classification

Require: Training set L_t, Unlabeled set U_t and batch size k

```
Train a classifier w^t on the training set L_t
Compute information vector c (Equation 4) and the divergence matrix R (Equation 5)
Compute the matrix D, as described in Equation 9
Compute a vector v ∈ ℜ^{|U_t|×1} containing the column sums of D
Identify the k largest entries in v and derive the initial solution x_0
t = 1
repeat
    Compute x_t' = D.x_{t-1}
    Identify F_t as the index set of x_t' with top k values
    Set x_t to be 1 on the index set F_t and 0 otherwise
    t = t + 1
until Convergence
Select a batch of k unlabeled samples based on the final solution x_t
```

4. EXPERIMENTS AND RESULTS

4.1 Datasets and Experimental Setup

We used 9 datasets from different application domains, with varied feature dimensions and label tree sizes, in our experiments, to corroborate the generalizibility of our framework. Each dataset was divided into an initial training set, an unlabeled set and a test set. For a given batch size k, each algorithm selected k instances from the unlabeled pool to be labeled in each iteration. After each iteration, the selected points were removed from the unlabeled set, appended to the training set and the performance was evaluated on the test set. The goal was to study the improvement in performance on the test set with increasing sizes of the training set. The setup is similar to previous work [4]. All the results were averaged over 5 runs, to mitigate the effects of randomness. The dataset details together with the training, unlabeled and test splits are summarized in Table 1 (we only consider data samples that belong to a single leaf node in the label tree; multi-label samples, belonging to multiple leaf nodes simultaneously were discarded). The algorithms were implemented in MATLAB on an Intel Core processor with 2.60 GHz CPU and 6 GB RAM. The weight parameter λ was selected as 4 and a Gaussian kernel with parameter 1 was used to compute the kernelized distances between the unlabeled samples (based on preliminary experiments).

The entropy term in the objective function necessitates a classifier which can provide concrete posterior probability estimates of its output classes. We therefore used the hierarchical Logistic Regression (LR) as the base classification model in our experiments. As depicted in Equation (1), the hierarchical LR traverses the label tree from the root until it reaches a leaf node, and at each node, follows the child that has the largest classification score. The label at the final leaf node is the predicted label. Since most datasets used in our experiments are high dimensional and sparse (Table 1), we used the SLEP package [20] to train the models.

4.2 Competing Algorithms

We compared the proposed framework against the following three algorithms: (*i*) *Random Sampling*, where a batch of samples is queried at random (used for baseline comparison), (*ii*) *Uncertainty Sampling*, which computes the entropy of every unlabeled sample on the leaf-node labels and the top

Dataset	Dimensionality	Label Tree Size	Initial Training	Unlabeled	Testing	Batch Size
20 Newsgroups	26,214	25	100	2,900	2,000	30
Astronomy	54,632	34	100	700	845	10
Biology	148,644	99	100	1,900	1,151	30
CLEF	80	47	100	2,900	2,000	30
Earth Sciences	71,756	52	100	1,900	1,102	30
Math	108,559	104	100	1,900	1,862	30
OHSUMED	18,143	87	100	2,900	2000	30
Reuters	47,236	97	100	2,900	2000	30
WIPO	74,437	188	100	900	700	20

Table 1: Dataset Details

k uncertain points are queried for annotation (k being the batch size) [5] and (iii) Hierarchical-Structured Embedded Variance (*HSEV*) (proposed by Cheng *et al.* [4]), which uses an uncertainty and diversity based criterion for sample selection and is the state-of-the-art for BMAL in hierarchical classification. However, the HSEV is based on a heuristic sample selection strategy, where at every iteration, the unlabeled sample with the maximum weighted sum of uncertainty and diversity values is selected for annotation. Our method, on the other hand, is based on a concrete mathematical formulation and also provides theoretical guarantees on the quality of the solution.

4.3 Evaluation Metrics

We used the zero-one error (error rate) and the tree-loss error (the graph distance between the predicted and actual categories) to study the performance of the algorithms. These are commonly used evaluation metrics in hierarchical classification [27].

4.4 Active Learning Performance

The results are depicted in Figure 1 (0/1 error) and Figure 2 (tree-loss error). In each graph, the x-axis denotes the number of iterations (rounds of active learning) and the y-axis denotes the error (0/1 or tree-loss). The proposed framework outperforms Random Sampling on all the datasets; both the zero-one error and the tree loss error drop at a faster rate with increasing size of the labeled set. The BatchRank algorithm therefore identifies the salient and exemplar instances for manual annotation and attains a given level of performance with much reduced human labeling effort. The Uncertainty Sampling and HSEV methods depict better performance than Random Sampling, but are not as good as BatchRank. The proposed framework depicts the best performance in terms of both the evaluation metrics. The results unanimously lead to the conclusion that our algorithm outperforms HSEV and the other baselines consistently across all datasets. For the Biology and WIPO, however, the performance of all the algorithms are close.

We conducted a 2-sided paired t-test at the significance level of $p < 0.05$, to compare the performance of the proposed BatchRank algorithm against its closest competitor across all datasets (such a test has been previously used to analyze the performance of active learning algorithms [10]). Our analysis revealed that the proposed framework outperforms its closest competitor on 8 out of the 9 datasets used in our study (except the OHSUMED dataset). The performance improvement achieved by BatchRank is therefore statistically significant.

Dataset	Uncertain	HSEV	BatchRank
20 News	2.82±1.83	3.37±2.03	5.93±1.87
Astronomy	0.90±0.57	1.17±0.71	3.77±0.82
Biology	9.80±2.62	19.32±2.08	27.78±4.87
CLEF	0.84±0.61	1.22±0.79	3.56±1.07
Sciences	3.13±1.19	6.32±2.78	7.47±2.26
Math	8.56±2.27	15.90±3.16	17.28±3.11
Ohsumed	3.79±2.31	6.06±3.66	9.28±2.94
Reuters	5.72±3.83	12.75±3.92	16.23±3.36
WIPO	3.17±2.07	13.56±3.59	19.22±3.89

Table 2: Average time taken (in seconds) to query a batch of samples from the unlabeled set.

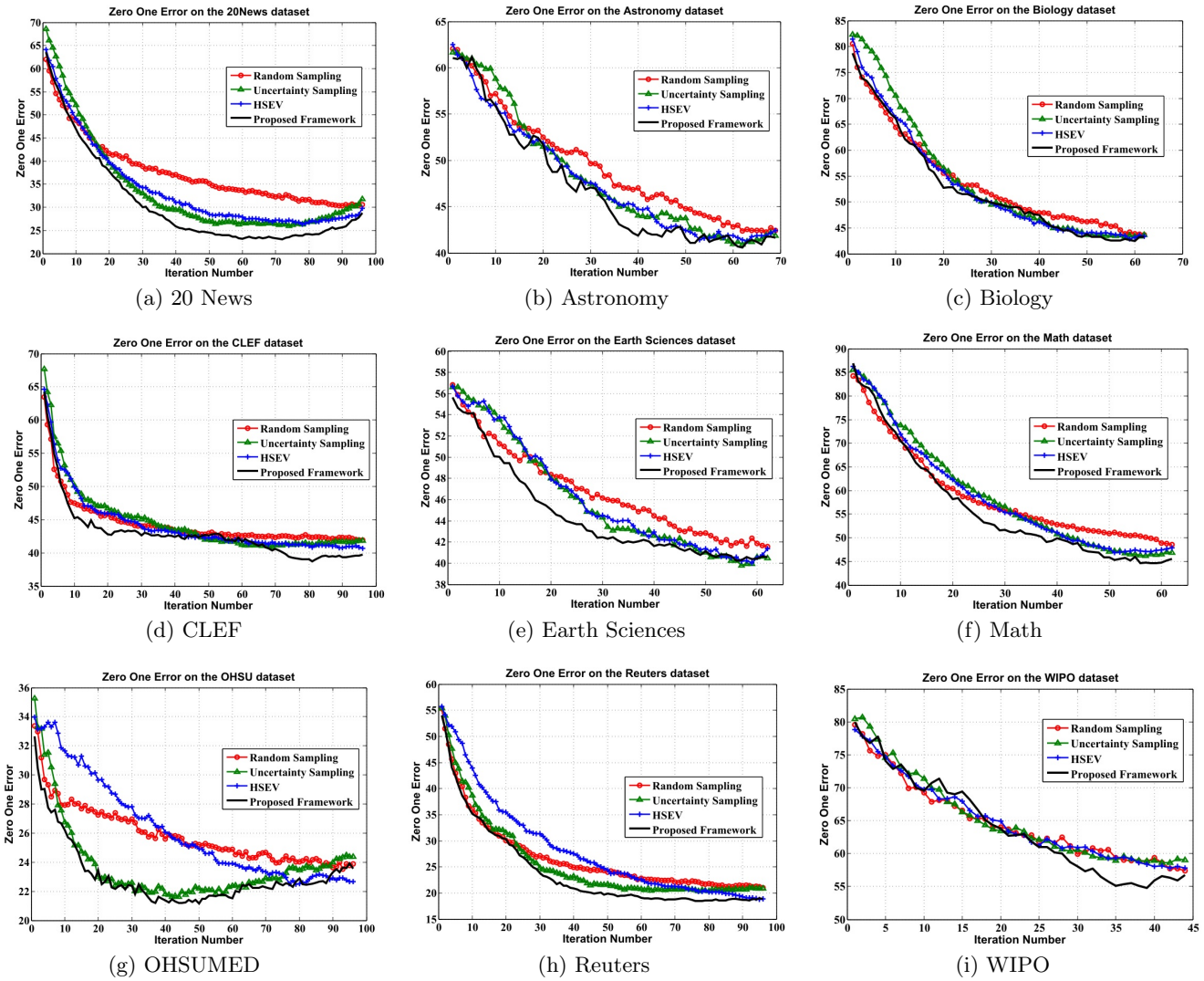
4.5 Computation Time Analysis

In this section, we study the computation time of the BMAL algorithms. Random Sampling does not take any time practically, since it does not involve any computations; we hence exclude it from our analysis. For the proposed BatchRank algorithm, a pairwise distance matrix needs to be computed, as detailed in Section 3.1. This needs to be computed just once and can be done offline before the active learning process starts and the selected unlabeled data samples are passed to the human annotators for labeling. We therefore do not include the time taken for the distance matrix computation in our run-time analysis.

Table 2 reports the average time taken to query a batch of samples by each algorithm. The Entropy method is the most efficient in terms of computation; however, it is not consistent in its performance, as noted in Figures 1 and 2. The proposed framework depicts comparable run-time as the HSEV method. Our method, therefore, outperforms the HSEV algorithm in terms of learning performance and incurs only marginal increment in computational overhead.

4.6 Solution Quality Analysis

In this section, we empirically validate the quality of the solution obtained using the proposed algorithm, for different values of the number of unlabeled samples n and the batch size k. We plot the ratio $\frac{\widehat{m}^T D \widehat{m}}{m^{*T} D m^*}$, where $\widehat{m}$ is the solution obtained using the BatchRank framework (together with the truncated power iterations) and m^* is the optimal solution (note that the IQP in Equation (10) can be solved exactly for small scale problems, which gives us the optimal solution). We present results for $n = 40, 60$ and $k = 10, 20$, for each value of n, in Figure 3. Each figure depicts the results of 500 random, symmetric matrices for the specific n and k. We note that the ratio of the functional values obtained using our method is very close to 1 (greater than 0.8 in most cases). This corroborates the fact that the proposed framework yields high quality approximations of the NP-



(a) 20 News (b) Astronomy (c) Biology

(d) CLEF (e) Earth Sciences (f) Math

(g) OHSUMED (h) Reuters (i) WIPO

Figure 1: BMAL for Hierarchical Classification - Zero One Error Graphs (Best viewed in color.)

hard IQP (Equation (10)) and the solutions obtained very closely match the optimal.

4.7 Parameter Sensitivity

In this section, we study the effect of batch size (number of samples selected in each iteration of active learning) and the size of the initial training set, on the learning performance. We used the 20 Newsgroups dataset for this study and the zero-one error was used as the evaluation metric. To study the effect of batch size, we used the same training, unlabeled and test splits, as outlined in Table 1. Four different batch sizes were studied - $10, 30, 50$ and 100. The results are presented in Figure 4; the proposed algorithm depicts the best performance across all batch sizes.

We used the following four values of the initial training set to study its effect: $50, 100, 150$ and 200. The sizes of the unlabeled and test sets were the same as in Table 1. The batch size was fixed at 30. The results are shown in Figure 5. From the results, we conclude that the proposed framework outperforms the other algorithms consistently.

5. CONCLUSION AND FUTURE WORK

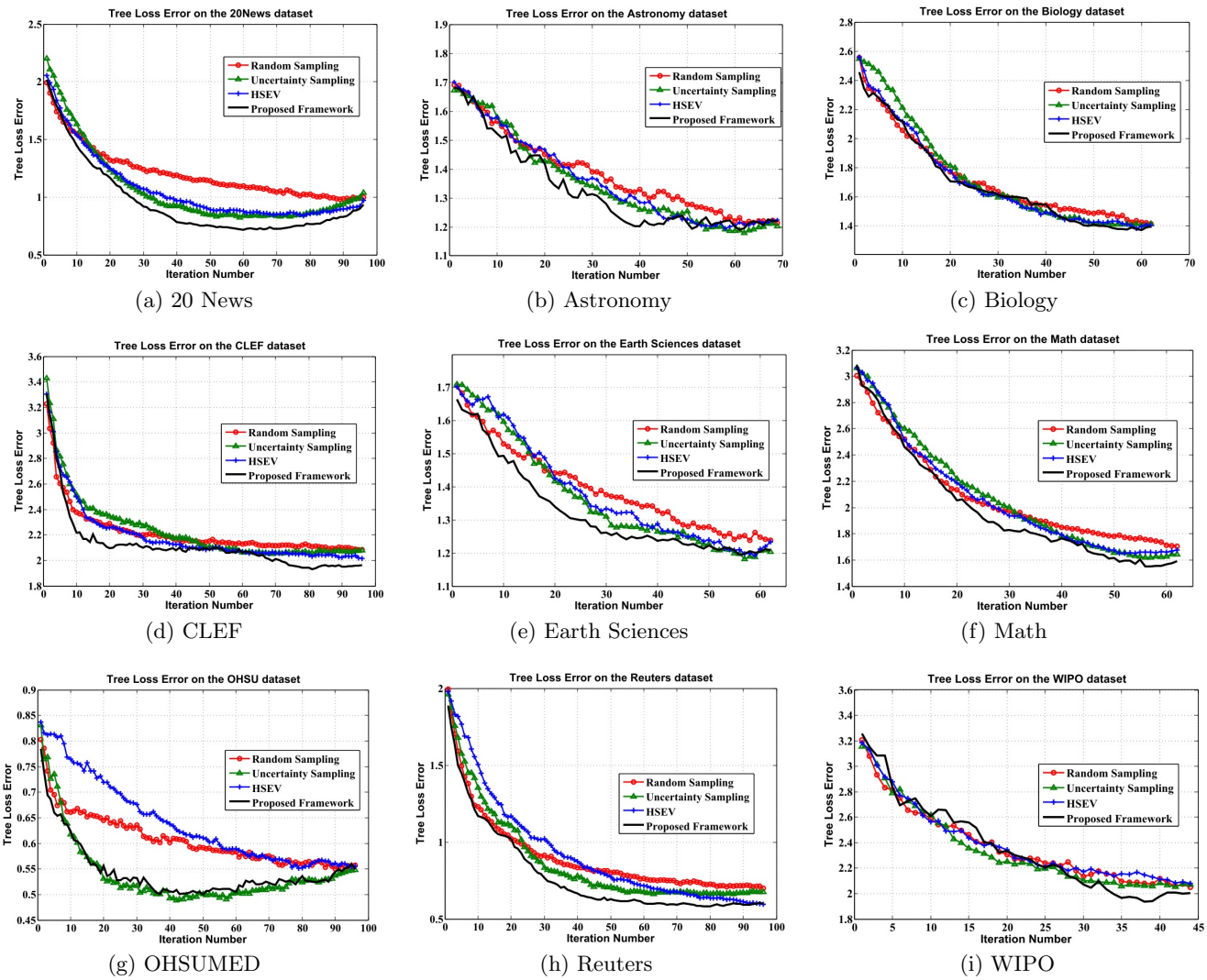
In this paper, we proposed a novel BMAL algorithm for multi-class hierarchical classification. The selective sampling problem was posed as an NP-hard integer quadratic programming; we then derived a convex relaxation to solve the NP-hard IQP and established a bound on the solution quality of the relaxation. Our empirical results on several challenging, real-world datasets corroborate the merit of the proposed approach over the state-of-the-art techniques and also the fact that it delivers high quality solutions. Future work will mainly involve extension of our framework to hierarchical, multi-label active learning.

6. ACKNOWLEDGMENT

This research is sponsored in part by NIH LM010730 and ONR N00014-11-1-0108.

7. REFERENCES

[1] M. Balcan, S. Hanneke, and J. Vaughan. The true sample complexity of active learning. In *Machine Learning*, 2010.



(a) 20 News (b) Astronomy (c) Biology

(d) CLEF (e) Earth Sciences (f) Math

(g) OHSUMED (h) Reuters (i) WIPO

Figure 2: BMAL for Hierarchical Classification - Tree Loss Error Graphs (Best viewed in color)

[2] K. Brinker. Incorporating diversity in active learning with support vector machines. *ICML*, 2003.

[3] L. Cai and T. Hofmann. Hierarchical document categorization with support vector machines. In *CIKM*, 2004.

[4] Y. Cheng, Z. Chen, H. Fei, F. Wang, and A. Choudhary. Batch mode active learning with hierarchical-structured embedded variance. In *SDM*, 2014.

[5] Y. Cheng, K. Zhang, Y. Xie, A. Agarwal, and A. Choudhary. On active learning in hierarchical classification. In *CIKM*, 2012.

[6] O. Dekel, J. Keshet, and Y. Singer. Large margin hierarchical classification. In *ICML*, 2004.

[7] I. Dimitrovski, D. Kocev, S. Loskovska, and S. Dzeroski. Hierchical annotation of medical images. In *International Multiconference - Information Society IS*, 2008.

[8] S. Dumais and T. Chen. Hierarchical classification of web content. In *Proceedings of SIGIR*, 2000.

[9] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. In *Journal of the ACM*, 1995.

[10] Y. Guo. Active instance sampling via matrix partition. In *NIPS*, 2010.

[11] Y. Guo and D. Schuurmans. Discriminative batch mode active learning. In *NIPS*, 2007.

[12] S. Hanneke. A bound on the label complexity of agnostic active learning. In *ICML*, 2007.

[13] S. Hoi, R. Jin, and M. Lyu. Batch mode active learning with applications to text categorization and image retrieval. *IEEE TKDE*, 2009.

[14] S. Hoi, R. Jin, J. Zhu, and M. Lyu. Semi-supervised SVM batch mode active learning for image retrieval. In *CVPR*, 2008.

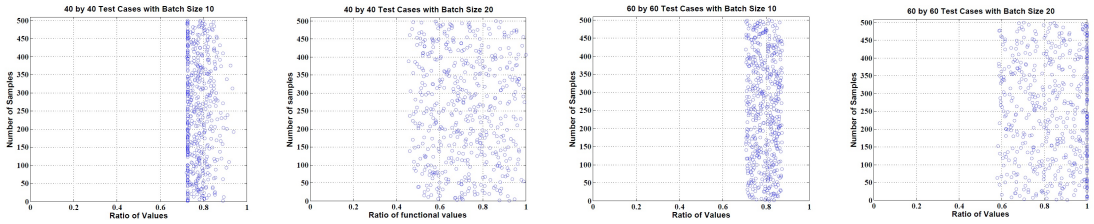
[15] S. C. H. Hoi, R. Jin, and M. R. Lyu. Large-scale text categorization by batch mode active learning. In *WWW*. ACM, 2006.

[16] D. Lewis, Y. Yang, T. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. In *JMLR*, 2004.

[17] P. Li, T. Hastie, and K. Church. Very sparse random projections. In *KDD*, 2006.

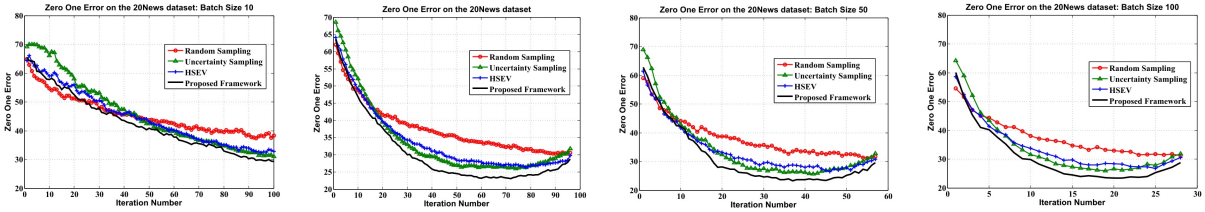
[18] X. Li, D. Kuang, and C. Ling. Active learning for hierarchical text classification. In *PAKDD*, 2012.

[19] X. Li, C. Ling, and H. Wang. Effective top-down active learning for hierarchical text classification. In *PAKDD*,



(a) $n = 40, k = 10$ (b) $n = 40, k = 20$ (c) $n = 60, k = 10$ (d) $n = 60, k = 20$

Figure 3: Analysis of Solution Quality of the Proposed Framework



(a) Batch Size 10 (b) Batch Size 30 (c) Batch Size 50 (d) Batch Size 100

Figure 4: Effect of Batch Size on the 20 Newsgroups dataset - Zero One Error Graphs (Best viewed in color)

2013.

[20] J. Liu, S. Ji, and J. Ye. SLEP: Sparse learning with efficient projections. In *Technical Report, Arizona State University*, 2009.

[21] J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor. Kernel–based learning of hierarchical multilabel classification models. In *JMLR*, 2006.

[22] G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In *ICML*, 2000.

[23] B. Settles. Active learning literature survey. In *Technical Report 1648, University of Wisconsin-Madison*, 2010.

[24] D. Shen, J. Zhang, J. Su, G. Zhou, and C. Tan. Multi-criteria-based active learning for named entity recognition. In *ACL*, 2004.

[25] S. Vempala. The random projection method. In *Americal Mathematical Society*, 2004.

[26] X. Yuan and T. Zhang. Truncated power method for sparse eigenvalue problems. In *JMLR*, 2013.

[27] D. Zhou, L. Xiao, and M. Wu. Hierarchical classification via orthogonal transfer. In *ICML*, 2011.

APPENDIX

A. AN IMPROVED BOUND ON THE SOLUTION QUALITY

In section 3.3, we derived a deterministic bound on the solution quality of the BatchRank algorithm. In this section, we attempt to improve the guarantee on the solution quality. To this end, we present a second convex relaxation to solve the original NP-hard IQP. Starting with Equation (10), we first make the following variable transformation:

$$y_i = 2(m_i - \frac{1}{2}) \Rightarrow m_i = \frac{y_i + 1}{2}$$
$$\Rightarrow \sum_{i=1}^{n} y_i = 2\sum_{i=1}^{n} m_i - n = 2k - n \triangleq p$$

where $n = |U_t|$ is the number of unlabeled data samples in the unlabeled pool. The entire optimization problem in Equation (10) is now rewritten in terms of the new variable y (ignoring the constant $\frac{1}{4}$):

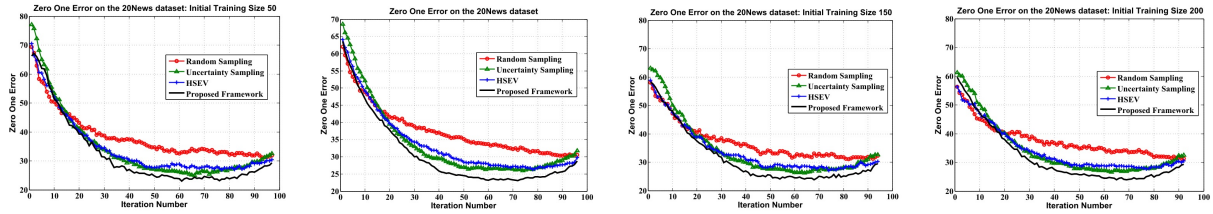
$$\max_{y} \sum_{i,j} d_{ij}(y_i + 1)(y_j + 1)$$
$$\text{s.t.} \quad y_i \in \{-1, 1\}, \forall i \quad \text{and} \quad \sum_{i=1}^{|U_t|} y_i = p \tag{18}$$

A.1 Multi-dimensional Relaxation

Since solving the integer quadratic program in Equation (18) is NP hard, we consider relaxations of the constraints. Specifically, we follow the strategy proposed by Goemans and Williamson [9], where each variable y_i is relaxed to a multidimensional vector v_i belonging to $\Re^n$ of unit Euclidean norm, instead of a one dimensional scalar variable. In other words, we assume that each vector v_i belongs to the n-dimensional unit sphere S_n. The relaxation of the NP hard problem in Equation (18) is therefore given by (ignoring the constant 1):

$$\max_{v} \sum_{i,j} d_{ij}(v_i^T v_j + v_0^T v_i + v_0^T v_j) \tag{19}$$
$$\text{s.t.} \quad v_i \in S_n, \forall i \quad \text{and} \quad \sum_{i=1}^{|U_t|} v_i = p \tag{20}$$

where v_0 is a vector of n-dimensions with all entries 1. (The equality constraint in Equation (20) implies that the sum across all the entries in all the v_i vectors should equal the scalar constant p. This is because each scalar variable y_i was relaxed into a vector variable v_i and the sum of all y_i was constrained to be equal to p). Once we solve for the vectors v from this formulation (we present the solution details below), we select a random unit vector r that is uniformly distributed on the unit sphere and find the dot product of r with every vector v_i. We then select the set of unlabeled points whose corresponding v vectors yield a positive dot



(a) Initial Training Size 50 (b) Initial Training Size 100 (c) Initial Training Size 150 (d) Initial Training Size 200

Figure 5: Effect of Initial Training Set Size on the 20 Newsgroups dataset - Zero One Error Graphs (Best viewed in color)

product with the random unit vector r, as in [9]. However, we note that the number of positive dot products may not exactly equal k, which means that the number of instances selected by this algorithm may not equal the pre-specified batch size.

A.2 Semi-Definite Programming (SDP) Relaxation

Using the decomposition $Q = B^T B$, we note that any positive semi-definite (psd) matrix with diagonal entries 1 corresponds to a set of unit vectors v_i if we correspond the vector v_i to the i^{th} column of the matrix B. Then, $q_{ij} = v_i v_j$, which accounts for the term $v_i^T v_j$ in the objective function of Equation (19). However, to incorporate the terms $v_0^T v_i$ and $v_0^T v_j$ in the objective, the matrix Q is decomposed as

$$Q = \begin{pmatrix} v_0^T \\ B^T \end{pmatrix} \begin{pmatrix} v_0 & B \end{pmatrix}$$

where $B = [v_1\ v_2\ \ldots v_n]$. We can therefore rewrite the entire relaxation in terms of the defined matrix Q (in the previous equation) as follows:

$$\max_{Q} \sum_{i,j} d_{i,j}(Q_{i+1,j+1} + Q_{1,i+1} + Q_{1,j+1}) \qquad (21)$$

$$\text{s.t.} \quad Q_{ii} = 1, \text{for i} = 2 \text{ to n+1},$$

$$\sum_{j=2}^{n+1} Q_{1j} = p, \quad Q_{11} = n \quad \text{and} \quad Q \succeq 0 \quad (\text{Q is psd})$$

This is a semi-definite programming (SDP) problem and can be solved using existing software packages like SeDuMi.

A.3 Probabilistic Solution Guarantee

We first rewrite the objective in Equation (19) in a simplified form as follows:

$$\sum_{i,j} d_{ij}(v_i^T v_j + v_0^T v_i + v_0^T v_j) = \sum_{i,j} \widehat{d_{ij}}(v_i^T v_j)$$

where v_0 is the vector obtained from the first row of the decomposed matrix after solving the SDP problem and $\widehat{d_{ij}}$ is obtained from d_{ij}, to simplify the representation. The main result regarding the solution bound of this algorithm is summarized in the following theorem:

THEOREM 2. *Let W denote the value of the objective function produced using the algorithm and $E(W)$ denote its expectation. Also, let $\widehat{D}_{total}$ denote the sum of all entries in the matrix $\widehat{D}$. Then,the following bound holds:*

$$[E(W) + \widehat{D}_{total}] \geq 0.87856 \left[\sum_{i,j} \widehat{d_{ij}} v_i v_j + \widehat{D}_{total}\right]$$

PROOF. Consider a random unit vector r. By the linearity of expectation, the expectation of the value of the objective function is given by (we drop the sub-scripts i and j from the summation for notational convenience):

$$\begin{aligned} E(W) &= \sum \widehat{d}_{ij}[1.Pr(sgn(v_i.r) = sgn(v_j.r)) \\ &\quad + (-1).Pr(sgn(v_i.r) \neq sgn(v_j.r))] \\ &= \sum \widehat{d}_{ij}[1 - 2.Pr(sgn(v_i.r) \neq sgn(v_j.r))] \\ &= \sum \widehat{d}_{ij}\left[1 - 2\frac{arccos(v_i.v_j)}{\pi}\right] \end{aligned}$$

where $sgn(x) = 1$ if $x \geq 0$ and -1 otherwise. The last equality follows from the result proved by Goemans and Williamson [9]. Further, it can be shown that for $-1 \leq z \leq 1$, $1 - \frac{arccos(z)}{\pi} \geq \alpha.\frac{1}{2}(1+z)$, where

$$\alpha = \min_{0 \leq \theta \leq \pi} \frac{2}{\pi}.\frac{\theta}{1 - \cos\theta} \geq 0.87856$$

(the proof of the above inequality can be found in [9]). That is, $1 - 2\frac{arccos(z)}{\pi} \geq \alpha(1+z) - 1$. Since in our formulation, the $v_i s$ are all unit vectors, we have $-1 \leq z = v_i v_j \leq 1, \quad \forall i, j$. Therefore, $1 - 2\frac{arccos(v_i v_j)}{\pi} \geq \alpha(1 + v_i v_j) - 1$.

Combining all of the above, we get

$$\begin{aligned} E(W) &\geq \sum \widehat{d}_{ij}[\alpha(1 + v_i v_j) - 1] \\ &= \alpha \sum \widehat{d}_{ij} v_i v_j + (\alpha - 1)\widehat{D}_{total} \\ &\Rightarrow [E(W) + \widehat{D}_{total}] \geq \alpha \left[\sum_{i,j} \widehat{d}_{ij} v_i v_j + \widehat{D}_{total}\right] \end{aligned}$$

which proves the theorem. □

This relaxation therefore provides a better guarantee on the quality of the solution. However, this method involves solving an SDP problem, which is computationally more expensive than the previous approach. Also, the number of selected samples may not exactly equal the batch size. A thorough investigation of this solution methodology will be taken up as part of future research.