Similarity-based Multi-label Learning

Ryan A. Rossi
Adobe Research
rrossi@adobe.com

Nesreen K. Ahmed
Intel Labs
nesreen.k.ahmed@intel.com

Hoda Eldardiry
Palo Alto Research Center
heldardiry@parc.com

Rong Zhou
Google
rongzhou@google.com

***Abstract*—Multi-label classification is an important learning problem with many applications. In this work, we propose a similarity-based approach for multi-label learning called SML. We also introduce a similarity-based approach for predicting the label set size. SML is amenable to streaming data and online learning, naturally able to handle changes in the problem domain, robust to training data with skewed class label sets, accurate with low variance, and lends itself to an efficient parallel implementation. The experimental results demonstrate the effectiveness of SML for multi-label classification where it is shown to compare favorably with a wide variety of existing algorithms across a range of evaluation criterion.**

I. Introduction

Multi-label classification is an important learning problem [1] with applications in bioinformatics [2], image & video annotation [3], [4] and query suggestions [5]. The goal of multi-label classification is to predict a label vector $\mathbf{y} \in \{0,1\}^K$ for a given unseen data point $\mathbf{x} \in \mathbb{R}^M$. Previous work has mainly focused on reducing the multi-label problem to a more standard one such as multi-class [6], [7] and binary classification [8], ranking [9] and regression [10], [11]. Standard multi-class approaches can be used by mapping a multi-label problem with K labels to a multi-class problem with 2^K labels [6], [7]. Binary classification methods can also be used by copying each feature vector K times and for each copy k an additional dimension is added with value k; and the training label is set to 1 if label k is present and 0 otherwise [8]. Rank-based approaches attempt to rank the relevant labels higher than irreverent ones [9]. Regression methods map the label space onto a vector space where standard regression methods can be applied [10], [11]. Methods that explicitly model label correlations have also been proposed [12], [13] as well as methods for large-scale problems with priors [14] and missing labels [15]. For further details, we refer the reader to a recent survey by Zhang *et al.* [16].

In this work, we introduce a similarity-based approach for multi-label learning called SML that gives rise to a new class of methods for multi-label classification. SML has the following important properties: it is accurate with low variance, amenable to streaming data and online learning, naturally able to handle changes in the problem domain, robust to training data with skewed/unbalanced class label sets, and lends itself to an efficient parallel implementation. Furthermore, we also present a similarity-based set size prediction algorithm for predicting the number of labels associated with an unknown test instance $\mathbf{x}$. Experiments on a number of data sets demonstrate the effectiveness of SML as it compares favorably to existing methods across a wide range of evaluation criterion. The experimental results indicate the practical significance of SML.

In addition, SML is a direct approach for multi-label learning. This is in contrast to existing methods that are mostly *indirect approaches* that transform the multi-label problem to a binary, multi-class, or regression problem and apply standard algorithms (*e.g.*, decision trees). Furthermore, other rank-based approaches such as RANK-SVM [9] are also indirect extensions of SVM [17], [18] to multi-label classification. Notably, SML completely avoids such mappings (required by SVM) and is based on the more general notion of similarity [19].

II. Preliminaries

Let $\mathcal{X} = \mathbb{R}^M$ denote the input space and let $\mathcal{Y} = \{1, 2, \ldots, K\}$ denote the set of possible class labels. Given a multi-label training set $\mathcal{D}$ defined as:

$$\mathcal{D} = \{(\mathbf{x}_1, Y_1), \ldots, (\mathbf{x}_N, Y_N)\} \tag{1}$$

where $\mathbf{x}_i \in \mathcal{X}$ is a M-dimensional training vector representing a single instance and Y_i is the label set associated with $\mathbf{x}_i$. Given $\mathcal{D}$ the goal of the multi-label learning problem is to learn a function $h : \mathcal{X} \rightarrow 2^K$ which predicts a set of labels for an unseen instance $\mathbf{x}_j \in \mathbb{R}^M$. A multi-label learning algorithm typically outputs a real-valued function $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ where $f_k(\mathbf{x}_i)$ is the confidence of label $k \in \mathcal{Y}$ for the unseen test instance $\mathbf{x}_i$. Given an instance $\mathbf{x}_i$ and its associated label set Y_i, a good multi-label learning algorithm will output larger values for labels in Y_i and smaller values for labels not in Y_i.

We consider a variety of evaluation criterion for comparing multi-label learning algorithms. The multi-label hamming loss is the fraction of incorrectly classified instance-label pairs:

$$\mathbb{E}_{\mathcal{D}}(f) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{K} \Big| h(\mathbf{x}_i) \,\Delta\, Y_i \Big| \tag{2}$$

where Δ is the symmetric difference between the predicted label set $\widehat{Y}_i = h(\mathbf{x}_i)$ and the actual ground truth label set Y_i. A misclassified instance-label pair corresponds to either not predicting an actual label of $\mathbf{x}_i$ or incorrectly predicting an irrelevant label for $\mathbf{x}_i$. One-error evaluates how many times the top-ranked label is not in the set of ground truth (held-out) labels:

$$\mathbb{E}_{\mathcal{D}}(f) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\Big[\big[\arg\max_{k \in \mathcal{Y}} f_k(\mathbf{x}_i)\big] \notin Y_i \Big] \tag{3}$$

978-1-5090-6014-6/18/$31.00 ©2018 IEEE

where for any predicate p the indicator function $\mathbb{I}[\,p\,] = 1$ iff p holds and 0 otherwise. Perfect performance is achieved when $\mathbb{E}_{\mathcal{D}}(f) = 0$. Given a set of labels ordered from most likely to least, coverage measures the max position in the ordered list such that all proper labels are recovered:

$$\mathbb{E}_{\mathcal{D}}(f) = \frac{1}{N} \sum_{i=1}^{N} \max_{k \in Y_i} \pi(\mathbf{x}_i, k) - 1 \tag{4}$$

where $\pi(\mathbf{x}_i, k)$ is the rank of label $k \in Y_i$ when the real-valued function values $f_1(\mathbf{x}_i), f_2(\mathbf{x}_i), \ldots, f_K(\mathbf{x})$ representing label confidence scores are sorted in descending order (largest to smallest). *Ranking loss* measures the fraction of reversely ordered label pairs:

$$\mathbb{E}_{\mathcal{D}}(f) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{|Y_i||\bar{Y}_i|} \left| \left\{ (k, k') \in Y_i \times \bar{Y}_i \mid f_k(\mathbf{x}_i) \leq f_{k'}(\mathbf{x}_i) \right\} \right| \tag{5}$$

Average precision measures the average fraction of relevant labels ranked higher than a particular label $k \in Y_i$:

$$\mathbb{E}_{\mathcal{D}}(f) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{|Y_i|} \sum_{k \in Y_i} \frac{\left| \left\{ k' \in Y_i \mid \pi(\mathbf{x}_i, k') \leq \pi(\mathbf{x}_i, k) \right\} \right|}{\pi(\mathbf{x}_i, k)} \tag{6}$$

Multi-label algorithms should have high precision (Eq. 6) with low hamming loss (Eq. 2), one-error (Eq. 3), coverage (Eq. 4), and ranking loss (Eq. 5).

III. Similarity-based Multi-label Learning

This section describes the class of similarity-based multi-label learning methods called SML.

A. Estimation

Given a multi-label training set $\mathcal{D} = \{(\mathbf{x}_1, Y_1), \ldots, (\mathbf{x}_j, Y_j), \ldots, (\mathbf{x}_N, Y_N)\}$ where $\mathbf{x}_j \in \mathbb{R}^M$ is a M-dimensional training vector representing a single instance and Y_j is the label set associated with $\mathbf{x}_j$, the goal of multi-label classification is to predict the label set Y_i of an unseen instance $\mathbf{x}_i \in \mathbb{R}^M$. In this work, we normalize $\mathbf{x}_i$ as $\|\mathbf{x}_i\| = \sqrt{\langle \mathbf{x}_i, \mathbf{x}_i \rangle}$ where $\langle \mathbf{x}_i, \mathbf{x}_i \rangle$ is the inner product and $\|\mathbf{x}_i\|$ is simply the magnitude of $\mathbf{x}_i$, thus the normalized vector is simply $\mathbf{x}_i / \|\mathbf{x}_i\|$. However, SML works well for other norms which can be selected depending on the application. Given the subset $\mathcal{D}_k \subseteq \mathcal{D}$ of training instances with label $k \in \{1, 2, \ldots, K\}$ defined as

$$\mathcal{D}_k = \left\{ (\mathbf{x}_i, Y_i) \in \mathcal{D} \mid k \in Y_i \right\} \tag{7}$$

we estimate the weight $f_k(\mathbf{x}_i)$ of label k for an unseen test instance $\mathbf{x}_i \in \mathbb{R}^M$ as:

$$f_k(\mathbf{x}_i) \; = \sum_{\mathbf{x}_j \in \mathcal{D}_k} \Phi \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle \tag{8}$$

where Φ denotes an arbitrary similarity function. Notably, the proposed family of similarity-based multi-label learning algorithms can leverage any arbitrary similarity function Φ. Furthermore, our approach does not require mappings in high-dimensional Hilbert spaces [20], [21] as required by RANK-SVM [9]. We define a few parameterized similarity functions below. Given M-dimensional vectors $\mathbf{x}_i$ and $\mathbf{x}_j$, the RBF similarity function is:

Algorithm 1 Similarity-based Multi-label Learning (SML)

```
1  procedure SML(a set of training instances $\mathcal{D} = \{(\mathbf{x}_1, Y_1), \ldots, (\mathbf{x}_N, Y_N)\}$, an unseen test instance $\mathbf{x}_i$, a similarity function $\Phi$ with hyperparameter $\gamma$)
2      Normalize the unseen test instance $\mathbf{x}_i \leftarrow g(\mathbf{x}_i)$
3      $\mathbf{p} = \begin{bmatrix} 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^K$        ▷ also denoted by $f(\mathbf{x}_i)$
4      parallel for each $(\mathbf{x}_j, Y_j) \in \mathcal{D}$ do
5          $S_{ij} = \Phi \langle \mathbf{x}_i, \mathbf{x}_j \rangle$
6          for each $k \in Y_j$ do
7              $p_k = p_k + S_{ij}$
8          end for
9      end parallel
10     Predict label set $\widehat{Y}_i$ using Eq. 13 (or by solving Eq. 16 and using $t$ to predict $\widehat{Y}_i$)
11     return label confidences $\mathbf{p} \in \mathbb{R}^K$ and label set $\widehat{Y}_i$
```

$$\Phi(\mathbf{x}_i, \mathbf{x}_j) = \exp\Big[-\gamma \, \|\mathbf{x}_i - \mathbf{x}_j\|^2 \Big] \tag{9}$$

A common class of similarity measures for vectors of uniform length are polynomial functions:

$$\Phi(\mathbf{x}_i, \mathbf{x}_j) = \Big[\langle \mathbf{x}_i, \mathbf{x}_j \rangle + c \Big]^d \tag{10}$$

where $\langle \cdot, \cdot \rangle$ is the inner product of two vectors, d is the degree of the polynomial, and c is a regularization term trading off higher-order terms for lower-order ones in the polynomial. Linear-SML and quadratic-SML are special cases of Eq. (10) where $d = 1$ and $d = 2$, respectively. Polynomial kernels are important for NLP and other applications [22]. Furthermore, all label weights denoted by $f(\mathbf{x}_i)$ for test instance $\mathbf{x}_i$ are estimated as:

$$f(\mathbf{x}_i) \; = \begin{bmatrix} f_1(\mathbf{x}_i) \\ \vdots \\ f_K(\mathbf{x}_i) \end{bmatrix} = \begin{bmatrix} \sum\limits_{\mathbf{x}_j \in \mathcal{D}_1} \Phi \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ \vdots \\ \sum\limits_{\mathbf{x}_j \in \mathcal{D}_K} \Phi \langle \mathbf{x}_i, \mathbf{x}_j \rangle \end{bmatrix} \tag{11}$$

The approach is summarized in Algorithm 1.

After estimating $f(\mathbf{x}_i) = \begin{bmatrix} f_1(\mathbf{x}_i) & \cdots & f_K(\mathbf{x}_i) \end{bmatrix}^T \in \mathbb{R}^K$ via Eq. 11, we predict the label set Y_i of $\mathbf{x}_i$; see Section III-B for further details. As an aside, binary and multi-class problems are special cases of the proposed family of similarity-based multi-label learning algorithms. Furthermore, the binary and multi-class algorithms are recovered as special cases of SML when $|Y_i| = 1$, for $1 \leq i \leq N$. Indeed, the proposed similarity-based multi-label learning approach expresses a family of algorithms as many components are interchangeable such as the similarity function Φ, normalization, and the sampling or sketching approach to reduce the training data. The expressiveness and flexibility of SML enables it to be easily adapted for application-specific tasks and domains. In addition, SML lends itself to an efficient and straightforward parallel implementation.

B. Similarity-based Label Set Prediction

We present a similarity-based approach for predicting the label set size. For each label set Y_i corresponding to a training instance $\mathbf{x}_i$ in the training set $\mathcal{D}$, we set its label to $|Y_i|$, *i.e.*, the number of labels associated with $\mathbf{x}_i$. Let $\mathbf{y} = [\, y_1 \; y_2 \; \cdots \; y_N \,] \in \mathbb{R}^N$ denote an N-dimensional label vector where each $y_i = |Y_i|$ is the new transformed cardinality label of $\mathbf{x}_i$ in $\mathcal{D}$. The new label vector $\mathbf{y} \in \mathbb{R}^N$ is used to predict the label set size. In particular, the new training data is: $\mathcal{D}' = \{(\mathbf{x}_i, y_i)\}$, for $i = 1, 2, \ldots, N$ where the label set Y_i of each instance is replaced by its transformed label y_i that encodes the label set size $|Y_i|$ of $\mathbf{x}_i$. Furthermore, let $\mathcal{Y}' = \{|Y_i|\}_{i=1}^N$ denote the label space given by the transformation and $K' = |\mathcal{Y}'|$ denote the number of unique labels (*i.e.*, label set cardinalities). It is straightforward to see that the above transforms the original multi-label classification problem into a general multi-class problem for predicting the label set size.

Given $\mathcal{D}' = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, the label set size of an unseen instance $\mathbf{x}_i$ is predicted as follows. First, the similarity of $\mathbf{x}_i$ with respect to each training instance $(\mathbf{x}_j, y_j) \in \mathcal{D}'$ is derived as $\Phi(\mathbf{x}_i, \mathbf{x}_j)$, $1 \leq j \leq N$ and the similarities from training instances with the same set size (label) $k \in \mathcal{Y}'$ are combined via addition. More formally, the similarity of instances in $\mathcal{D}'$ of the same set size (class label) $k \in \mathcal{Y}'$ with respect to $\mathbf{x}_i$ is:

$$f_k(\mathbf{x}_i) \; = \sum_{\mathbf{x}_j \in \mathcal{D}'_k} \Phi \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle \tag{12}$$

where $\mathcal{D}'_k \subseteq \mathcal{D}'$ is the subset of training instances with label $k \in \mathcal{Y}'$. Therefore, we predict the set size of $\mathbf{x}_i$ using the following decision function:

$$\xi(\mathbf{x}_i) = \operatorname*{arg\,max}_{k \in \mathcal{Y}'} \sum_{\mathbf{x}_j \in \mathcal{D}'_k} \Phi \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle \tag{13}$$

where $\xi(\cdot)$ is the predicted label set size for $\mathbf{x}_i$. It is straightforward to see that $\xi(\mathbf{x}_i)$ is the label set size with maximum similarity. Given the label set size $\xi(\mathbf{x}_i)$, we predict the label set $\widehat{Y}_i$ of $\mathbf{x}_i$ by ordering the labels from largest to smallest weight based on $f_1(\mathbf{x}_i), f_2(\mathbf{x}_i), \ldots, f_K(\mathbf{x}_i)$ and setting $\widehat{Y}_i$ to the top $\xi(\mathbf{x}_i)$ labels with the largest weight. We also define a stronger decision function that requires a test instance $\mathbf{x}_i$ be more similar to class k than it is to the combined weight of all other classes:

$$\xi(\mathbf{x}_i) = \operatorname*{arg\,max}_{k \in \mathcal{Y}'} \; f_k(\mathbf{x}_i) > \sum_{c \neq k} f_c(\mathbf{x}_i) \tag{14}$$

Notice that Eq. 13 and Eq. 14 essentially solve regression problems using a multi-class variant of the proposed similarity-based approach.

Alternatively, we can infer the label set of $\mathbf{x}_i$ by learning a threshold function $t : \mathcal{X} \to \mathbb{R}$ such that:

$$h(\mathbf{x}) = \Big\{ \, k \;|\; f_k(\mathbf{x}) > t(\mathbf{x}), \; k \in \mathcal{Y} \Big\} \tag{15}$$

where $f_k(\mathbf{x})$ is the confidence of label $k \in \mathcal{Y}$ for the unseen test instance $\mathbf{x}$. To learn the threshold function $t(\cdot)$, we assume a linear model $t(\mathbf{x}) = \langle \mathbf{w}, f(\mathbf{x}) \rangle + b$. More formally, we solve the following problem based on the training set $\mathcal{D}$:

$$\operatorname*{minimize}_{\mathbf{w}, b} \; \sum_{i=1}^{N} \Big[\langle \mathbf{w}, f(\mathbf{x}_i) \rangle + b - s(\mathbf{x}_i) \Big]^2 \tag{16}$$

In Eq. 16, we set $s(\mathbf{x}_i)$ as:

$$\begin{aligned} s(\mathbf{x}_i) = \operatorname*{arg\,min}_{\tau \in \mathbb{R}} \; & \big| \{ k \in Y_i \text{ s.t. } f_k(\mathbf{x}_i) \leq \tau \} \big| \\ & + \; \big| \{ q \in \bar{Y}_i \text{ s.t. } f_q(\mathbf{x}_i) \geq \tau \} \big| \end{aligned} \tag{17}$$

where $\bar{Y}_i$ is the complement of Y_i. After learning the threshold function $t(\cdot)$, we use it to predict the label set Y_i for the unseen instance $\mathbf{x}_i$. Nevertheless, any approach that predicts the label set Y_i from the learned weights $f_1(\mathbf{x}_i), \ldots, f_K(\mathbf{x}_i)$ can be used by SML; see [1], [16] for other possibilities.

C. Complexity Analysis

SML is both time and space-efficient for large data and naturally amenable to streaming data and online learning [25], [26], [27]. Given a single test instance $\mathbf{x}$, the runtime of SML is $\mathcal{O}(NM\bar{K})$ where N is the number of training instances, M is the number of attributes, and $\bar{K} = \frac{1}{N}\sum_{i=1}^{N} |Y_i|$ is the average number of labels per training instance. This is straightforward to see as SML derives the similarity between each training instance's M-dimensional attribute vector. The space complexity of SML for a single test instance $\mathbf{x}$ is $\mathcal{O}(K)$ where K is the number of labels. This obviously is not taking into account the space required by SML and other methods to store the training instances and the associated label sets. For the similarity-based set size prediction approach, the time complexity is only $\mathcal{O}(NM)$ since the label set size with maximum similarity can be maintained in $o(1)$ time. However, the approach uses $\mathcal{O}(K')$ space where $K' \leq K$.

As an aside, if the M-dimensional feature vectors $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_i, \ldots\}$ are sparse (*i.e.*, $|\Omega(\mathbf{x}_i)| \ll M$ where $\Omega(\mathbf{x}_i)$ denotes the nonzero indices of $\mathbf{x}_i$) then $\Phi(\mathbf{x}_i, \mathbf{z}_j)$, for $1 \leq i \leq N$ is solved efficiently by hashing the values of the unseen test instance $\mathbf{x}_j$ via a perfect hash function and then using this to efficiently test the similarity between only the nonzero elements of $\mathbf{x}_i$. Thus, it takes $\mathcal{O}(|\Omega(\mathbf{x}_j)|)$ time to create the hash table for the unseen test instance $\mathbf{x}_j$ which is only performed once (in the outer loop) and then for each of the nonzero values in the training instance $\mathbf{x}_i$ we obtain from $\mathbf{x}_j$ the corresponding test instance feature value in only $o(1)$ time. This gives a total time complexity of $\mathcal{O}(|\Omega(\mathbf{x}_j)| + |\Omega(\mathbf{x}_i)|)$. However, since the hash table is only computed once (in the outer loop) for all N training instances this cost becomes neglible. Therefore, evaluating $\Phi(\mathbf{x}_i, \mathbf{z}_j)$, for $1 \leq i \leq N$ takes only $\mathcal{O}(|\Omega(\mathbf{x}_j)| + N|\Omega(\mathbf{x}_i)|) = \mathcal{O}(N|\Omega(\mathbf{x}_i)|)$. In terms

TABLE I
EXPERIMENTAL RESULTS FOR EACH MULTI-LABEL LEARNING ALGORITHM ON THE YEAST DATA (MEAN$\pm$STD).

Evaluation criterion	SML	ML-KNN [23]	BOOSTEXTER [8]	ADTBOOST.MH [24]	RANK-SVM [9]
Hamming loss ($\downarrow$)	**0.193** $\pm$ **0.013**	0.194 $\pm$ 0.010	0.220 $\pm$ 0.011	0.207 $\pm$ 0.010	0.207 $\pm$ 0.013
One-error ($\downarrow$)	**0.220** $\pm$ **0.021**	0.230 $\pm$ 0.030	0.278 $\pm$ 0.034	0.244 $\pm$ 0.035	0.243 $\pm$ 0.039
Coverage ($\downarrow$)	**6.082** $\pm$ **0.184**	6.275 $\pm$ 0.240	6.550 $\pm$ 0.243	6.390 $\pm$ 0.203	7.090 $\pm$ 0.503
Ranking loss ($\downarrow$)	**0.155** $\pm$ **0.011**	0.167 $\pm$ 0.016	0.186 $\pm$ 0.015	N/A	0.195 $\pm$ 0.021
Average precision ($\uparrow$)	**0.783** $\pm$ **0.016**	0.765 $\pm$ 0.021	0.737 $\pm$ 0.022	0.744 $\pm$ 0.025	0.749 $\pm$ 0.026

TABLE II
RELATIVE PERFORMANCE COMPARISON OF THE MULTI-LABEL LEARNING ALGORITHMS ON THE YEAST DATA.

Evaluation criterion	$\mathcal{A}_1$=SML $\mathcal{A}_2$=ML-KNN $\mathcal{A}_3$=BOOSTEXTER $\mathcal{A}_4$=ADTBOOST.MH $\mathcal{A}_5$=RANK-SVM
Hamming loss ($\downarrow$)	$\mathcal{A}_1 \succ \mathcal{A}_3,\ \mathcal{A}_1 \succ \mathcal{A}_4,\ \mathcal{A}_1 \succ \mathcal{A}_5,\ \mathcal{A}_2 \succ \mathcal{A}_3,\ \mathcal{A}_2 \succ \mathcal{A}_4,\ \mathcal{A}_2 \succ \mathcal{A}_5,\ \mathcal{A}_4 \succ \mathcal{A}_3,\ \mathcal{A}_5 \succ \mathcal{A}_3$
One-error ($\downarrow$)	$\mathcal{A}_1 \succ \mathcal{A}_3,\ \mathcal{A}_1 \succ \mathcal{A}_4,\ \mathcal{A}_1 \succ \mathcal{A}_5,\ \mathcal{A}_2 \succ \mathcal{A}_3,\ \mathcal{A}_4 \succ \mathcal{A}_3,\ \mathcal{A}_5 \succ \mathcal{A}_3$
Coverage ($\downarrow$)	$\mathcal{A}_1 \succ \mathcal{A}_2,\ \mathcal{A}_1 \succ \mathcal{A}_3,\ \mathcal{A}_1 \succ \mathcal{A}_4,\ \mathcal{A}_1 \succ \mathcal{A}_5$ $\mathcal{A}_2 \succ \mathcal{A}_3,\ \mathcal{A}_2 \succ \mathcal{A}_4,\ \mathcal{A}_2 \succ \mathcal{A}_5,\ \mathcal{A}_3 \succ \mathcal{A}_5,\ \mathcal{A}_4 \succ \mathcal{A}_3,\ \mathcal{A}_4 \succ \mathcal{A}_5$
Ranking loss ($\downarrow$)	$\mathcal{A}_1 \succ \mathcal{A}_2,\ \mathcal{A}_1 \succ \mathcal{A}_3,\ \mathcal{A}_1 \succ \mathcal{A}_5,\ \mathcal{A}_2 \succ \mathcal{A}_3,\ \mathcal{A}_2 \succ \mathcal{A}_5$
Average precision ($\uparrow$)	$\mathcal{A}_1 \succ \mathcal{A}_2,\ \mathcal{A}_1 \succ \mathcal{A}_3,\ \mathcal{A}_1 \succ \mathcal{A}_4,\ \mathcal{A}_1 \succ \mathcal{A}_5,\ \mathcal{A}_2 \succ \mathcal{A}_3,\ \mathcal{A}_2 \succ \mathcal{A}_4$
Total order (Eq. 20)	SML(17) $>$ ML-KNN(8) $>$ ADTBOOST.MH(-3) $>$ RANK-SVM(-8) $>$ BOOSTEXTER(-14)

of space, it takes $\mathcal{O}(M)$ space to store the hash table, $\mathcal{O}(K)$ to store the estimated similarity weights for the test instance $\mathbf{x}_j$ and $\mathcal{O}(2|\Omega(\mathbf{x}_j)| + 2|\Omega(\mathbf{x}_i)|)$ to store the sparse test and train instance. As an aside, the labels of each instance are stored as sets with no additional data structures required.

D. Group-based Centroid Sketch

Now we describe a group-based centroid sketching approach for multi-label learning algorithms. The goal of the approach is to reduce the computational complexity of a multi-label classification algorithm while maintaining a similar classification performance (high accuracy, low error). As an aside, obviously the sketching algorithm must take significantly less time than solving the multi-label learning problem directly using all available training data. Therefore, in general, a sketching algorithm must be fast taking sub-linear or linear time at most.

There are two general approaches. The first general approach to computing a sketch is based on a sampling mechanism (or distribution) $\mathbb{F}$ (*i.e.*, the distribution $\mathbb{F}$ may be a weighted or uniform distribution). These sampling-based methods compute a sketch $\mathcal{D}_s \subseteq \mathcal{D}$ of the training data where $\mathcal{D}_s$ is a small but *representative sample* of the original training set $\mathcal{D}$ such that $N \gg N_s$ where $N = |\mathcal{D}|$ and $N_s = |\mathcal{D}_s|$. For instance, in the case of a uniform distribution we have the following:

$$\mathcal{D}_s = \{(\mathbf{x}_i, Y_i) \in \mathcal{D} \mid i \sim \text{UniformDiscrete}\{1, 2, \ldots, N\}\}_{j=1}^{N_s}$$

The second type of approach is based on generating novel training instances from the initial training data. These generative-based methods compute a sketch that represents a novel training set $\mathcal{D}_s \nsubseteq \mathcal{D}$ where $(\mathbf{x}_i^*, Y_i) \notin \mathcal{D}$, for $1 \leq i \leq N_s$ such that $N_s \ll N$. The goal is to derive or learn (in an unsupervised fashion) new training instances that summarize the original training data while improving the power of generalization. As an aside, it is possible for a multi-label learning approach using the set of new training vectors to outperform the same approach using the original training data, *e.g.*, if the new training vectors generalize better. In this work, we focus primarily on the second type and propose a generative-based sketching method for multi-label problems. We describe the group-based centroid sketching approach below:

The first step is to derive $Y^* = \{Y_1^*, Y_2^*, \ldots, Y_L^*\}$ consisting of all the unique label sets from the training data $\mathcal{D}$. In addition, let $X^* = \{\mathbf{X}_1^*, \mathbf{X}_2^*, \ldots, \mathbf{X}_L^*\}$ denote the sets of training vectors associated with the $\mathcal{L}$ label sets in Y^* where $\mathbf{X}_k^* \in X^*$ is a matrix containing the training vectors associated with the label set $Y_k^* \in Y^*$. For each $\mathbf{X}_k^* \in X^*$ where $\mathbf{X}_k^*$ is an $N_k \times M$ matrix:

$$\mathbf{X}_k^* = \left[\ \cdots\ \mathbf{x}_i\ \cdots\ \right]^T \tag{18}$$

we derive a $C \times M$ matrix $\mathbf{C}_k = [\ \cdots\ \mathbf{c}_i\ \cdots\ ]$ of "iterative centroids" where $C \leq N_k$ by solving:

$$\arg\min_{\mathcal{S}} \sum_{i=1}^{C} \sum_{\mathbf{x} \in S_j} \|\mathbf{x} - \mathbf{c}_i\|^2 \tag{19}$$

where the N_k training vectors in $\mathbf{X}_k^*$ associated with the label set Y_k^* are partitioned into $C \leq N_k$ sets $\mathcal{S} = \{S_1, \ldots, S_C\}$. Notice that k-means is used to derive C "iterative centroids" (Equation 19). However, any approach that derives a compact summarization of the data can be used. Next, each iterative centroid vector $\mathbf{c}_i$ in $\mathbf{C}_k$ is assigned the label set Y_k^* (which can also be defined as a binary vector $\mathbf{y}_k^* \in \{0, 1\}^K$). Finally, we use the centroids $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_L$ along with the associated label sets $Y_1^*, Y_2^*, \ldots, Y_L^*$ as input into a multi-label learning algorithm such as SML (Algorithm 1). It is straightforward to see that if $C = N_k$ then we recover the actual training vectors $[\ \cdots\ \mathbf{x}_i\ \cdots\ ]$ as the centroids. Furthermore, if $C = 1$ then the new training vector is simply the centroid (mean vector) of the

TABLE III
RESULTS OF THE MULTI-LABEL LEARNING ALGORITHMS FOR NATURAL SCENE CLASSIFICATION (MEAN±STD).

Evaluation criterion	SML	ML-KNN [23]	BOOSTEXTER [8]	ADTBOOST.MH [24]	RANK-SVM [9]
Hamming loss (↓)	**0.140 ± 0.009**	0.169 ± 0.016	0.179 ± 0.015	0.193 ± 0.014	0.253 ± 0.055
One-error (↓)	**0.252 ± 0.026**	0.300 ± 0.046	0.311 ± 0.041	0.375 ± 0.049	0.491 ± 0.135
Coverage (↓)	0.984 ± 0.112	**0.939 ± 0.100**	**0.939 ± 0.092**	1.102 ± 0.111	1.382 ± 0.381
Ranking loss (↓)	**0.164 ± 0.020**	0.168 ± 0.024	0.168 ± 0.020	N/A	0.278 ± 0.096
Average precision (↑)	**0.852 ± 0.016**	0.803 ± 0.027	0.798 ± 0.024	0.755 ± 0.027	0.682 ± 0.093

TABLE IV
PERFORMANCE COMPARISON OF THE MULTI-LABEL LEARNING ALGORITHMS FOR SCENE CLASSIFICATION.

Evaluation criterion	$\mathcal{A}_1$=SML $\mathcal{A}_2$=ML-KNN $\mathcal{A}_3$=BOOSTEXTER $\mathcal{A}_4$=ADTBOOST.MH $\mathcal{A}_5$=RANK-SVM
Hamming loss (↓)	$\mathcal{A}_1 \succ \mathcal{A}_2,\ \mathcal{A}_1 \succ \mathcal{A}_3,\ \mathcal{A}_1 \succ \mathcal{A}_4,\ \mathcal{A}_1 \succ \mathcal{A}_5,\ \mathcal{A}_2 \succ \mathcal{A}_3,\ \mathcal{A}_2 \succ \mathcal{A}_4,\ \mathcal{A}_2 \succ \mathcal{A}_5,\ \mathcal{A}_3 \succ \mathcal{A}_4,\ \mathcal{A}_3 \succ \mathcal{A}_5,\ \mathcal{A}_4 \succ \mathcal{A}_5$
One-error (↓)	$\mathcal{A}_1 \succ \mathcal{A}_2,\ \mathcal{A}_1 \succ \mathcal{A}_3,\ \mathcal{A}_1 \succ \mathcal{A}_4,\ \mathcal{A}_1 \succ \mathcal{A}_5,\ \mathcal{A}_2 \succ \mathcal{A}_4,\ \mathcal{A}_2 \succ \mathcal{A}_5,\ \mathcal{A}_3 \succ \mathcal{A}_4,\ \mathcal{A}_3 \succ \mathcal{A}_5,\ \mathcal{A}_4 \succ \mathcal{A}_5$
Coverage (↓)	$\mathcal{A}_1 \succ \mathcal{A}_4,\ \mathcal{A}_1 \succ \mathcal{A}_5,\ \mathcal{A}_2 \succ \mathcal{A}_4,\ \mathcal{A}_2 \succ \mathcal{A}_5,\ \mathcal{A}_3 \succ \mathcal{A}_4,\ \mathcal{A}_3 \succ \mathcal{A}_5,\ \mathcal{A}_4 \succ \mathcal{A}_5$
Ranking loss (↓)	$\mathcal{A}_1 \succ \mathcal{A}_5,\ \mathcal{A}_2 \succ \mathcal{A}_5,\ \mathcal{A}_3 \succ \mathcal{A}_5$
Average precision (↑)	$\mathcal{A}_1 \succ \mathcal{A}_2,\ \mathcal{A}_1 \succ \mathcal{A}_3,\ \mathcal{A}_1 \succ \mathcal{A}_4,\ \mathcal{A}_1 \succ \mathcal{A}_5,\ \mathcal{A}_2 \succ \mathcal{A}_4,\ \mathcal{A}_2 \succ \mathcal{A}_5,\ \mathcal{A}_3 \succ \mathcal{A}_4,\ \mathcal{A}_3 \succ \mathcal{A}_5,\ \mathcal{A}_4 \succ \mathcal{A}_5$
Total order (Eq. 20)	SML(15) > ML-KNN(7) > BOOSTEXTER(5) > ADTBOOST.MH(-8) > RANK-SVM(-19)

$N_k \times M$ matrix $\mathbf{X}_k^*$.

IV. EXPERIMENTS

In this section, we investigate SML for multi-label classification on a number of multi-label problems from different domains using a range of evaluation criterion. We compare the performance of SML against a variety of multi-label algorithms including:

- ML-KNN [23]: A kNN-based multi-label approach that uses Euclidean distance to find the top-k instances that are closest in the N-dimensional euclidean space. ML-KNN was shown to perform well for a variety of multi-label problems.
- BOOSTEXTER [8]: A boosting-based multi-label algorithm called BOOSTEXTER.
- ADTBOOST.MH [24]: An indirect multi-label approach that uses decision trees.
- RANK-SVM [9]: An indirect multi-label SVM approach based on ranking.

For BOOSTEXTER and ADTBOOST.MH we use 500 and 50 boosting rounds respectively since performance did not change with more rounds (which is consistent with [23]). For RANK-SVM we use polynomial kernels with degree 8 which performs best as shown in [9], [23]. Unless otherwise mentioned, our approach uses the RBF similarity function in Eq. (9); the RBF hyperparameter is learned automatically via k-fold cross-validation on 10% of the labeled data. All multi-label learning algorithms are evaluated using a wide variety of evaluation criterion including hamming loss, one error, coverage, ranking loss, and average precision. Multi-label algorithms should have high precision with low hamming loss, one-error, coverage, ranking loss.

Gene functional classification: The first multi-label learning task we evaluate is based on predicting the functions of genes from Yeast Saccharomyces cerevisiae - a widely studied organism in bioinformatics [2]. Each gene may take on multiple functional classes. Each gene consists of a concatenation of micro-array expression data and phylogenetic profile data. Following Elisseeff *et al.* [9], we preprocess the data such that only the known structure of the functional classes are used. This corresponds to using only the functional classes in the top hierarchy as done by [9], [23]. The resulting multi-label yeast data consists of $N = 2417$ genes where each gene is represented by a $(M = 103)$-dimensional feature vector. There are $K = 14$ functional classes (labels).

We use 10-fold cross-validation and show the mean and standard deviation. Experimental results for SML and other multi-label learning algorithms are reported in Table I. Notably, all multi-label algorithms are compared across a wide range of evaluation metrics. The best result for each evaluation criterion is shown in bold. In all cases, our approach outperforms all other multi-label learning algorithms across all 5 evaluation criterion. Furthermore, the variance of SML is also smaller than the variance of other multi-label learning algorithms in most cases. This holds across all multi-label learning algorithms for coverage, average precision, and ranking loss.[1]

To better understand the relative predictive performance between the multi-label classification algorithms, we define a partial order $\succ$ between the algorithms for each evaluation metric where $\mathcal{A}_1 \succ \mathcal{A}_2$ implies that algorithm $\mathcal{A}_1$ is better than $\mathcal{A}_2$ for a given evaluation criterion (*e.g.*, ranking loss). Table II summarizes the partial order between all the multi-label learning algorithms in terms of each evaluation metric.

The partial order $\succ$ measures the relative performance

[1]Note the ADTBOOST.MH [24] program does not provide ranking loss.

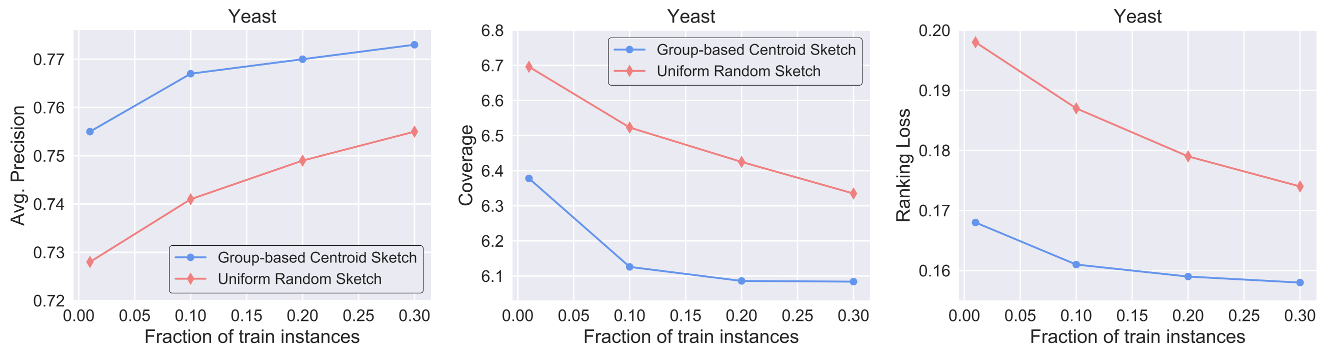

Fig. 1. Experimental results comparing the various sketch approaches as the fraction of train instances varies. Note the number of centroids learned by our approach depends on the number of training vectors associated with a given set of labels.

between two algorithms for a specific evaluation criterion, but does not measure the overall superiority of an algorithm over all algorithms and evaluation criterion (Equation 2-6). Therefore, we derive a score for each algorithm which allows us to compare the overall superiority of an algorithm over another across all evaluation criterion. For this we use the scoring scheme from [23]. Given an algorithm $\mathcal{A}_i$, we measure the overall superiority of $\mathcal{A}_i$ over all the other algorithms $\mathcal{A}_j \in \mathcal{A}$ and across all evaluation criterion $\mathcal{E} = \{\mathbb{E}_1, \ldots, \mathbb{E}_p\}$ as follows:

$$\Gamma(\mathcal{A}_i) = \sum_{\mathbb{E}_k \in \mathcal{E}} \sum_{\substack{\mathcal{A}_j \in \mathcal{A} \\ i \neq j}} \Delta\big(\mathcal{A}_i, \mathcal{A}_j \parallel \mathbb{E}_k\big) \tag{20}$$

where

$$\Delta\big(\mathcal{A}_i, \mathcal{A}_j \parallel \mathbb{E}_k\big) = \begin{cases} 1 & \text{if } \mathcal{A}_i \succ \mathcal{A}_j \text{ holds} \\ -1 & \text{if } \mathcal{A}_j \succ \mathcal{A}_i \text{ holds} \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

From Equation 20 it is straightforward to derive a total order on the set of all multi-label algorithms $\mathcal{A}$. The total order along with the score $\Gamma(\mathcal{A}_i)$ of each algorithm $\mathcal{A}_i \in \mathcal{A}$ are provided in the last row of Table II.

Overall, SML significantly outperforms all other multi-label learning algorithms across all evaluation criterion as summarized by the total order (and scores derived from Equation 20) reported in Table II. The scores shown in parentheses summarize the number of times an algorithm outperforms another or vice-versa. Strikingly, the difference in score between SML and the next best multi-label algorithm ML-KNN is large.

Scene classification: The second multi-label learning task we evaluate SML for is natural scene classification using image data. In scene classification each image may be assigned multiple labels representing different natural scenes such as an image labeled as a mountain and sunset scene. Therefore, given an unseen image the task is to predict the set of scenes (labels) present in it. The scene data consists of 2000 images where each image contains a set of manually assigned labels. There are $K = 5$ labels, namely, desert, mountains, sea, sunset, and trees. Each image is represented by a 294-dimensional feature vector derived using the approach in [7].

We use 10-fold cross-validation and show the mean and standard deviation. The experimental results of SML and the other multi-label algorithms using the natural scene classification data are reported in Table III. The best result for each evaluation criterion is in bold. From Table III, it is obvious that SML outperforms all other multi-label algorithms on all but one evaluation criterion, namely, coverage. In terms of coverage ML-KNN and BOOSTEXTER are tied and have slightly lower coverage than SML.

The relative performance between the algorithms for scene classification is shown in Table IV and is similar to the relative performance observed using the yeast data for gene functional classification. The main difference is that BOOSTEXTER outperforms ADTBOOST.MH and RANK-SVM. In particular, the total order given by Eq. 20 is SML(15) $>$ ML-KNN(7) $>$ BOOSTEXTER(5) $>$ ADTBOOST.MH(-8) $>$ RANK-SVM(-19). However, it is straightforward to derive the partial order "$\succ$" and total order "$>$" from Table III using Equation 20. Overall, it is clear from Table IV that SML is superior to all multi-label learning algorithms in terms of all evaluation criterion.

Web page categorization: We also investigate the effectiveness of SML for text categorization using a variety of web page categorization data sets collected from the Yahoo directory where each data set represents a top-level web page category from the Yahoo directory such as *Business & Economy* and the web pages under this category are categorized further into sub-categories. Following the same experimental setup in [23], we reduce the dimensionality of the feature vectors by selecting only the top 2% most frequent words used among the collection of web pages (documents). After selecting the terms, each web page (document) is represented by an M-dimensional feature vector where each feature value represents the frequency of a given word on a particular page.

Experimental results are reported in Table V-VI. The best result for each evaluation criterion is in bold. SML outperforms the other algorithms over all web category data sets in terms of one-error, coverage, ranking loss, and average precision. In terms of hamming loss, there are a few web categories

TABLE V
EXPERIMENTAL RESULTS WEB CATEGORIZATION.

	SML	ML-KNN [23]	BOOSTEXTER [8]	ADTBOOST.MH [24]	RANK-SVM [9]
Ham. loss (Eq. 2) ↓					
Arts & Humanities	0.0610	0.0612	0.0652	**0.0585**	0.0615
Business & Economy	**0.0267**	0.0269	0.0293	0.0279	0.0275
Computers & Internet	**0.0382**	0.0412	0.0408	0.0396	0.0392
Education	0.0393	**0.0387**	0.0457	0.0423	0.0398
Entertainment	**0.0572**	0.0604	0.0626	0.0578	0.0630
Health	**0.0369**	0.0458	0.0397	0.0397	0.0423
Recreation & Sports	0.0602	0.0620	0.0657	**0.0584**	0.0605
Reference	0.0294	0.0314	0.0304	**0.0293**	0.0300
Science	**0.0322**	0.0325	0.0379	0.0344	0.0340
Social & Science	0.0228	**0.0218**	0.0243	0.0234	0.0242
Society & culture	**0.0537**	**0.0537**	0.0628	0.0575	0.0555
One-error (Eq. 3) ↓					
Arts & Humanities	**0.4988**	0.6330	0.5550	0.5617	0.6653
Business & Economy	**0.1001**	0.1213	0.1307	0.1337	0.1237
Computers & Internet	**0.3694**	0.4357	0.4287	0.4613	0.4037
Education	**0.4642**	0.5207	0.5587	0.5753	0.4937
Entertainment	**0.4180**	0.5300	0.4750	0.4940	0.4933
Health	**0.3090**	0.4190	0.3210	0.3470	0.3323
Recreation & Sports	**0.4501**	0.7057	0.5557	0.5547	0.5627
Reference	**0.3957**	0.4730	0.4427	0.4840	0.4323
Science	**0.4951**	0.5810	0.6100	0.6170	0.5523
Social & Science	**0.3260**	0.3270	0.3437	0.3600	0.3550
Society & culture	**0.4040**	0.4357	0.4877	0.4845	0.4270
Coverage (Eq. 4) ↓					
Arts & Humanities	**4.5893**	5.4313	5.2973	5.1900	9.2723
Business & Economy	**1.8047**	2.1840	2.4123	2.4730	3.3637
Computers & Internet	**3.2183**	4.4117	4.4887	4.4747	8.7910
Education	**3.1180**	3.4973	4.0673	3.9663	8.9560
Entertainment	**2.5320**	3.1467	3.0883	3.0877	6.5210
Health	**2.4831**	3.3043	3.0780	3.0843	5.5400
Recreation & Sports	**3.3320**	5.1010	4.4737	4.3380	5.6680
Reference	**2.3660**	3.5420	3.2100	3.2643	6.9683
Science	**4.7420**	6.0470	6.6907	6.6027	12.401
Social & Science	**2.5242**	3.0340	3.6870	3.4820	8.2177
Society & culture	**4.6080**	5.3653	5.8463	4.9545	6.8837

TABLE VI
(CONT.) EXPERIMENTAL RESULTS FOR WEB CATEGORIZATION.

	SML	ML-KNN [23]	BOOSTEXTER [8]	ADTBOOST.MH [24]	RANK-SVM [9]
Ranking loss (Eq. 5) ↓					
Arts & Humanities	**0.1220**	0.1514	0.1458	N/A	0.2826
Business & Economy	**0.0274**	0.0373	0.0416	N/A	0.0662
Computers & Internet	**0.0640**	0.0921	0.0950	N/A	0.2091
Education	**0.0680**	0.0800	0.0938	N/A	0.2080
Entertainment	**0.0883**	0.1151	0.1132	N/A	0.2617
Health	**0.0420**	0.0605	0.0521	N/A	0.1096
Recreation & Sports	**0.1150**	0.1913	0.1599	N/A	0.2094
Reference	**0.0583**	0.0919	0.0811	N/A	0.1818
Science	**0.0882**	0.1167	0.1312	N/A	0.2570
Social & Science	**0.0470**	0.0561	0.0684	N/A	0.1661
Society & culture	**0.1087**	0.1338	0.1483	N/A	0.1716
Avg. prec. (Eq. 6) ↑					
Arts & Humanities	**0.5970**	0.5097	0.5448	0.5526	0.4170
Business & Economy	**0.9015**	0.8798	0.8697	0.8702	0.8694
Computers & Internet	**0.7040**	0.6338	0.6449	0.6235	0.6123
Education	**0.6430**	0.5993	0.5654	0.5619	0.5702
Entertainment	**0.6885**	0.6013	0.6368	0.6221	0.5637
Health	**0.7632**	0.6817	0.7408	0.7257	0.6839
Recreation & Sports	**0.6490**	0.4552	0.5572	0.5639	0.5315
Reference	**0.7021**	0.6194	0.6578	0.6264	0.6176
Science	**0.6043**	0.5324	0.5006	0.4940	0.5007
Social & Science	**0.7535**	0.7481	0.7262	0.7217	0.6788
Society & culture	**0.6502**	0.6128	0.5717	0.5881	0.5717

where other methods perform better than SML. The total order (indicating the superiority of a multi-label learning algorithm over another) is SML $>$ {ML-KNN, BOOSTEXTER} $>$ ADTBOOST.MH $>$ RANK-SVM. Overall, SML is clearly superior to the other multi-label learning algorithms in terms of all evaluation criterion. This data differs fundamentally from the others in two main ways. First, the features are extremely sparse. Second, there are thousands of features as opposed to tens or hundreds of features. Therefore, we implemented a variant of SML that leverages specialized sparse data structures. This provided a significant improvement in performance (between 6-14 times faster) for such sparse data with the above characteristics.

Group-based Centroid Sketch: Now we evaluate the group-based centroid sketch approach. For comparison, we evaluate the approach against a uniform random sketch that samples training instances uniformly at random. Results are provided in Figure 1. Overall, the group-based centroid approach outperforms the other approach across all fractions of training instances and evaluation criterion. Furthermore, the group-based centroid sketch has a speedup of 11x compared to SML (using the full data). For the scene classification data, we find a runtime improvement of 90x when using only $C = 1$ centroid per unique label set. The improvement in runtime is largely determined by the number of unique label sets L relative to the total number of training instances. Notice that each unique label set is represented by at least one training example (centroid). Therefore, data sets with relatively few unique label sets (relative to the total number of training instances) will lead to a better runtime improvement than a data set with a relatively large number of unique label sets.

V. CONCLUSION

This work described a general class of similarity-based multi-label learning methods called SML. Furthermore, we also presented a similarity-based approach for predicting the label set size. Experiments on a number of data sets demonstrate the effectiveness of SML as it compares favorably to a variety of existing methods across a wide range of evaluation criterion and multi-label problems. We also described a group-based centroid sketch for speeding up SML and other multi-label methods. Overall, the predictive performance of the group-based sketch approach was shown to be similar to that of SML using the full training data across a range of evaluation criterion, while improving the runtime performance by an order of magnitude.

ACKNOWLEDGMENTS

We thank all the reviewers for many helpful suggestions and feedback.

REFERENCES

[1] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *IJDWM*, vol. 3, no. 3, 2006.

[2] P. Pavlidis and W. N. Grundy, "Combining microarray expression data and phylogenetic profiles to learn gene functional categories using support vector machines," in *ICCBB*, 2000, yeast data.

[3] G. Carneiro, A. B. Chan, P. J. Moreno, and N. Vasconcelos, "Supervised learning of semantic classes for image annotation and retrieval," *TPAMI*, vol. 29, no. 3, pp. 394–410, 2007.

[4] C. Wang, S. Yan, L. Zhang, and H.-J. Zhang, "Multi-label sparse coding for automatic image annotation," in *CVPR*, 2009, pp. 1643–1650.

[5] R. Agrawal, A. Gupta, Y. Prabhu, and M. Varma, "Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages," in *WWW*, 2013, pp. 13–24.

[6] A. McCallum, "Multi-label text classification with a mixture model trained by em," in *AAAI workshop on Text Learning*, 1999, pp. 1–7.

[7] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[8] R. E. Schapire and Y. Singer, "Boostexter: A boosting-based system for text categorization," *Machine learning*, vol. 39, no. 2-3, pp. 135–168, 2000.

[9] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *NIPS*, 2002, pp. 681–687.

[10] S. Ji, L. Sun, R. Jin, and J. Ye, "Multi-label multiple kernel learning," in *NIPS*, 2009, pp. 777–784.

[11] D. J. Hsu, S. M. Kakade, J. Langford, and T. Zhang, "Multi-label prediction via compressed sensing," in *NIPS*, 2009, pp. 772–780.

[12] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni, "Incremental algorithms for hierarchical classification," *JMLR*, vol. 7, no. Jan, pp. 31–54, 2006.

[13] L. Cai and T. Hofmann, "Exploiting known taxonomies in learning overlapping concepts." in *IJCAI*, vol. 7, 2007, pp. 708–713.

[14] B. Hariharan, L. Zelnik-Manor, M. Varma, and S. Vishwanathan, "Large scale max-margin multi-label classification with priors," in *ICML*, 2010, pp. 423–430.

[15] H.-F. Yu, P. Jain, P. Kar, and I. Dhillon, "Large-scale multi-label learning with missing labels," in *ICML*, 2014, pp. 593–601.

[16] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *TKDE*, vol. 26, no. 8, pp. 1819–1837, 2014.

[17] V. N. Vladimir, "The nature of statistical learning theory," 1995.

[18] P. Wolfe, "A duality theorem for non-linear programming," *Quarterly of applied mathematics*, pp. 239–244, 1961.

[19] R. A. Rossi, R. Zhou, and N. K. Ahmed, "Relational similarity machines," in *Proceedings of the 12th International Workshop on Mining and Learning with Graphs (MLG)*, 2016, pp. 1–8.

[20] J. Weston, C. Watkins *et al.*, "Support vector machines for multi-class pattern recognition." *ESANN*, vol. 99, pp. 219–224, 1999.

[21] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[22] Y.-W. Chang, C.-J. Hsieh, K.-W. Chang, M. Ringgaard, and C.-J. Lin, "Training and testing low-degree polynomial data mappings via linear svm," *JMLR*, vol. 11, no. Apr, pp. 1471–1490, 2010.

[23] M.-L. Zhang and Z.-H. Zhou, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern recognition*, vol. 40, no. 7, pp. 2038–2048, 2007.

[24] F. De Comité, R. Gilleron, and M. Tommasi, "Learning multi-label alternating decision trees from texts and data," in *MLDM*, vol. 2734. Springer, 2003, p. 35.

[25] I. Zliobaite, A. Bifet, B. Pfahringer, and G. Holmes, "Active learning with drifting streaming data," *TNNLS*, vol. 25, no. 1, pp. 27–39, 2014.

[26] N. K. Ahmed, N. Duffield, T. L. Willke, and R. A. Rossi, "On sampling from massive graph streams," in *VLDB*, 2017, pp. 1430–1441.

[27] L. O'callaghan, N. Mishra, A. Meyerson, S. Guha, and R. Motwani, "Streaming-data algorithms for high-quality clustering," in *ICDE*. IEEE, 2002, pp. 685–694.