Rank-Loss Support Instance Machines for MIML Instance Annotation

Forrest Briggs
Oregon State University
School of EECS
Corvallis, Oregon
briggsf@eecs.oregonstate.edu

Xiaoli Z. Fern
Oregon State University
School of EECS
Corvallis, Oregon
xfern@eecs.oregonstate.edu

Raviv Raich
Oregon State University
School of EECS
Corvallis, Oregon
raich@eecs.oregonstate.edu

ABSTRACT

Multi-instance multi-label learning (MIML) is a framework for supervised classification where the objects to be classified are bags of instances associated with multiple labels. For example, an image can be represented as a bag of segments and associated with a list of objects it contains. Prior work on MIML has focused on predicting label sets for previously unseen bags. We instead consider the problem of predicting instance labels while learning from data labeled only at the bag level. We propose Rank-Loss Support Instance Machines, which optimize a regularized rank-loss objective and can be instantiated with different aggregation models connecting instance-level predictions with bag-level predictions. The aggregation models that we consider are equivalent to defining a "support instance" for each bag, which allows efficient optimization of the rank-loss objective using primal sub-gradient descent. Experiments on artificial and real-world datasets show that the proposed methods achieve higher accuracy than other loss functions used in prior work, e.g., Hamming loss, and recent work in ambiguous label classification.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval; I.5.2 [**Design Methodology**]: Classifier Design and Evaluation

General Terms

Algorithms, Performance, Experimentation

Keywords

instance annotation, image annotation, multi-instance, multi-label, support vector machine, sub-gradient, bioacoustics

1. INTRODUCTION

Many problems in supervised classification have a certain structure, where the objects of interest (e.g., images or text documents) can naturally be decomposed into a collection of parts or formally, a bag-of-instances representation. For example, in image classification, an image is typically a bag, and the pixels or segments in it are instances. This structure motivates multiple-instance learning (MIL) [7]. The original formulation of MIL concerns problems where bags are associated with a single binary label. Zhou and Zhang [30] proposed multi-instance multi-label learning (MIML), where bags are instead associated with a set of labels. For example, an image might be associated with a list of the objects it contains.

MIML arises in situations where the cost of labeling individual instances becomes prohibitive and consequently multiple instances are grouped and associated with a set of labels. For example, labeling individual pixels in an image takes minutes, but assigning a few words to an image can be accomplished in seconds. In MIML, the training dataset consists of a collection of bags of instances, where each bag is associated with multiple labels. The goal is to learn a classifier that predicts the label set for a previously unseen bag. Numerous algorithms for MIML have been proposed and applied to image and text domains [30, 31, 27, 14, 20].

A related problem which has received little attention [29] is learning to predict instance labels from MIML training data. For example, one might train a classifier on a collection of images paired with lists of object names in each image, then make predictions about the label for each region in an image. This problem is called the *instance annotation* problem for MIML. The key issue in instance annotation is how to learn an instance-level classifier from a MIML dataset (which presents only bag-level labels).

A common strategy in designing MIML algorithms is to learn an instance-level model by minimizing a loss function defined at the bag level. For example, several MIML algorithms minimize bag-level Hamming loss, which captures the disagreement between a ground-truth label set, and a predicted label set. For instance annotation, typically the instance-level classifier outputs a score for each class, and the instance label is predicted as the highest scoring class. Therefore the predicted label depends on the ranking of class scores. Hamming loss is not appropriate in this context, despite its success at bag-level predictions, because it lacks a mechanism to calibrate the scores between different classes. This observation motivates us to introduce a rank-loss objective for instance annotation, which directly optimizes the ranking of classes.

In order to learn an instance-level classifier using a bag-level loss function, it is necessary to define an aggregation

KDD'12, August 12–16, 2012, Beijing, China.

model that connects instance-level predictions with bag-level predictions. In this paper, we examine two different aggregation models, which are equivalent to defining a "support instance" for each bag. Therefore, we name our methods Rank-Loss Support Instance Machines (SIM). In this paper, we make the following contributions:

- We propose a general rank-loss objective for instance annotation that can be instantiated with different aggregation models (Sec. 4).
- We propose an optimization procedure for the rank-loss objective which alternates between updating the support instances and solving a convex optimization problem. Using the Pegasos framework [18, 19], we show that a primal sub-gradient descent algorithm finds a solution to the convex problem within ϵ of optimal, with runtime linear in the number of bags, and $\frac{1}{\epsilon}$ (Sec. 4.2.2). The alternating optimization is also linear in the total number of instances.
- Experiments show that Rank-loss SIM achieves higher accuracy than Hamming loss, or ambiguous loss (a comparable state-of-the-art approach [5, 6]), and a novel `softmax` aggregation model outperforms the `max` model which has been used in prior work (Sec. 5).
- We introduce a real-world MIML dataset for instance annotation derived from over 90 minutes of bird song recordings collected in the field, containing multiple simultaneously vocalizing birds of different species. Rank-Loss SIM with the `softmax` model achieves over 80% accuracy in predicting the species of bird responsible for each sound in the recordings (given the list of species present in the recording).

2. PROBLEM STATEMENT

In this section we formalize the instance annotation problem and contrast it with several related problems.

We are given a training set of n bags $(X_1, Y_1), \ldots, (X_n, Y_n)$. Each X_i is a bag of n_i instances, i.e., $X_i = \{\mathbf{x}_{i,1}, \cdots, \mathbf{x}_{i,n_i}\}$, with $\mathbf{x}_{i,q} \in \mathcal{X}$, where $\mathcal{X} = \mathbb{R}^d$ is a d-dimensional feature space. Each bag X_i is associated with a label set $Y_i \subseteq \mathcal{Y}$ where $\mathcal{Y} = \{1, \cdots, c\}$ and c is the total number of classes. The goal of instance annotation in MIML is to learn an instance level classifier $f_{IA} : \mathcal{X} \to \mathcal{Y}$ that maps an element of the input space $\mathcal{X}$ to its corresponding class label.

The instance annotation problem is different from the traditional MIML learning problem studied by Zhou and Zhang [30] and many others, where the goal is to learn a bag-level classifier $F_{MIML} : 2^{\mathcal{X}} \to 2^{\mathcal{Y}}$. However, the design principles of many traditional MIML algorithms can be used to learn instance-level classification models, which is the approach we take in this paper.

Instance annotation is closely related to the classic supervised classification problem, where the goal is to learn an instance classifier, but using training data that does not have a bag structure and has each instance labeled individually. In contrast to MIML, this classic setting is often referred to as single-instance single-label (SISL) learning.

Ambiguous label classification (ALC) [13, 6] is another related framework. In ALC, there are no bags; instead instances are paired with a set of possible labels, only one of which is correct. An ALC dataset is $(\mathbf{x}_1, Y_1), \ldots, (\mathbf{x}_m, Y_m)$, where $\mathbf{x}_i \in \mathcal{X}$ and $Y_i \subseteq \mathcal{Y}$. In ALC the goal is to learn a classifier that predicts instance labels, hence an ALC classifier is a function $f_{ALC} : \mathcal{X} \to \mathcal{Y}$. A MIML instance annotation problem can be transformed into a ALC problem by creating one ALC instance for each instance in a MIML bag, paired with all of the labels from the bag. Hence ALC algorithms can be applied to MIML instance annotation problems. However, this reduction may discard useful bag-level structure in the MIML data.

3. BACKGROUND

Here we discuss some design patterns in traditional MIML algorithms (aimed at bag-level predictions) that contribute to our proposed methods (see Sec. 6 for different approaches to instance annotation, e.g., graphical models).

One common approach in MIML algorithms is to make bag-level predictions based on the outputs of instance-level models. Many algorithms leverage an assumption that the bag label set is equal to the union of the instance labels (i.e. there are no missing or spurious labels). This assumption is used in several MIML algorithms including M^{3}MIML [28], and D-MimlSvm [31]. The following formulation is frequently used to capture this assumption. Let $f_j(\mathbf{x}) : \mathcal{X} \to \mathbb{R}$ be a function which takes an instance and returns a real-valued score for class j. The output at the bag-level for class j is defined to be $F_j(X) = \max_{\mathbf{x} \in X} f_j(\mathbf{x})$. A bag-level classifier can be obtained by applying a threshold (e.g., 0) to the bag-level scores, i.e. $F(X) = \{y \in \mathcal{Y} : F_y(X) > 0\}$. Note that if an instance $\mathbf{x}^*$ within bag X is predicted to belong to class j, i.e., $f_j(\mathbf{x}^*) > 0$, the predicted label set for bag X will necessarily contain j because $F_j(X) = \max_{\mathbf{x} \in X} f_j(\mathbf{x}) \geq f_j(\mathbf{x}^*) > 0$. Hence, connecting instance and bag-level scores via the max function is equivalent to defining the bag label set as the union of the instance labels.

In contrast with SISL or instance annotation which are evaluated based on instance-level accuracy, MIML algorithms are evaluated based on their label set predictions. Two common performance measures are Hamming loss, and rank loss [31]. Hamming Loss is the number of false positives and false negatives, averaged over all classes and bags,

$$\frac{1}{nc} \sum_{i=1}^{n} \sum_{j=1}^{c} I[j \in F(X_i), j \notin Y_i] + I[j \notin F(X_i), j \in Y_i]$$

Rank loss captures the number of label pairs that are incorrectly ordered by the scores of the MIML classifier. Classes in the true label set should receive higher scores than classes that are not. Let $\bar{Y}$ denote the complement of Y. Rank loss is defined as

$$\frac{1}{n} \sum_{i=1}^{n} \frac{1}{|Y_i||\bar{Y}_i|} \sum_{j \in Y_i, k \in \bar{Y}_i} I[F_j(X_i) \leq F_k(X_i)] \tag{1}$$

These objectives are difficult to optimize directly because they are not continuous. Several algorithms for MIML can be viewed as optimizing a surrogate for Hamming loss. For example, D-MimlSvm [31] and M^{3}MIML [28] optimize variations of the following loss function (with different regularization terms)

$$\frac{1}{nc} \sum_{i=1}^{n} \sum_{j=1}^{c} \max\{0, 1 - Y_i^j F_j(X_i)\} \tag{2}$$

where $Y_i^j = +1$ if $j \in Y_i$ and -1 if $j \notin Y_i$.

The hinged Hamming-loss objective (2) can be decomposed into an independent MIL problem for each class, so it does not calibrate the scores between classes, which could make predicting an instance label based on the highest scoring class unreliable. To overcome this limitation, we consider rank loss instead. We are not aware of any MIML algorithms that learn an instance-level model by minimizing rank loss (i.e. rank loss has only been used as a performance measure, not as an objective). Rank loss has been used as an objective for (single-instance) multi-label SVMs [9].

4. PROPOSED METHODS

We consider classifiers for instance annotation that use one instance-level model for each class $f_j(\mathbf{x}) = \mathbf{w}_j \cdot \mathbf{x}$, and predict a specific label via $f(\mathbf{x}) = \arg\max_j f_j(\mathbf{x})$. The goal is to learn the weights $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_c]$ (note $\mathbf{x} \in \mathbb{R}^d, \mathbf{w}_j \in \mathbb{R}^d$, and $\mathbf{W} \in \mathbb{R}^{cd}$). The predicted instance label depends on the ranking of scores for each class, so we propose to directly optimize this ranking.

4.1 Rank-Loss Support Instance Machines

Because we are learning from a MIML dataset, we can only use a loss function that evaluates predictions at the bag level, i.e. the loss function measures the agreement between a bag label set and the bag-level scores $F_j(X_i)$. We propose a regularized surrogate for the rank loss (1):

$$g_{rank}(\mathbf{W}) = \frac{\lambda}{2}||\mathbf{W}||^2 + \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|Y_i||\bar{Y}_i|}\sum_{j\in Y_i, k\in\bar{Y}_i}\max\{0, 1 - \big(F_j(X_i) - F_k(X_i)\big)\}$$

This objective is designed to encourage a correct ranking of the bag-level scores for each class. For a bag X_i with corresponding label set Y_i, if $j \in Y_i$ and $k \in \bar{Y}_i$, then the loss is zero only if $F_j(X_i) > F_k(X_i) + 1$ (requiring a difference of at least 1 promotes a large-margin solution). The objective is also designed to facilitate primal sub-gradient descent optimization methods, which we discuss in the next section.

This objective can be instantiated with various aggregation models that compute bag-level scores from instance-level scores. For example, the `max` model, which has been used in prior work, with a linear instance classifier is

$$F_j(X_i) = \max_{\mathbf{x}_{i,q}\in X_i} f_j(\mathbf{x}_{i,q}) = \max_{\mathbf{x}_{i,q}\in X_i} \mathbf{w}_j \cdot \mathbf{x}_{i,q}$$

It is equivalent to write $F_j(X_i) = \mathbf{w}_j \cdot \hat{\mathbf{x}}_{i,j}$, where

$$\hat{\mathbf{x}}_{i,j} = \arg\max_{\mathbf{x}_{i,q}\in X_i} \mathbf{w}_j \cdot \mathbf{x}_{i,q} \tag{3}$$

We refer to $\hat{\mathbf{x}}_{i,j}$ as the "support instance" for bag X_i, class j, because the bag-level output for X_i depends only on the support instance for each class (analogous to a support vector). Therefore we name our proposed method Rank-Loss Support Instance Machines (SIM).

The `max` model represents each bag with the most characteristic instance of each class. This approach can ignore other instances that are also useful for learning, and may not be appropriate when the assumption that the bag label set is equal to the union of instance labels does not hold. We propose an alternative `softmax` model, which can also be expressed in terms of support instances, but has the advantage of basing the support on more than one instance per class for each bag. The `softmax` model represents each bag as a weighted average of the instances, with weights specific to each class:

$$F_j(X_i) = \sum_{\mathbf{x}_{i,q}\in X_i} \alpha^j_{i,q} f_j(\mathbf{x}_{i,q}) = \sum_{\mathbf{x}_{i,q}\in X_i} \alpha^j_{i,q} \mathbf{w}_j \cdot \mathbf{x}_{i,q}$$

The weights are defined according to a softmax rule,

$$\alpha^j_{i,q} = \frac{e^{\mathbf{w}_j \cdot \mathbf{x}_{i,q}}}{\sum_{\mathbf{x}'\in X_i} e^{\mathbf{w}_j \cdot \mathbf{x}'}}$$

We can also write the `softmax` model as $F_j(X_i) = \mathbf{w}_j \cdot \hat{\mathbf{x}}_{i,j}$, with the support defined as

$$\hat{\mathbf{x}}_{i,j} = \sum_{\mathbf{x}_{i,q}\in X_i} \alpha^j_{i,q} \mathbf{x}_{i,q} \tag{4}$$

4.2 Optimization

We can rewrite the rank-loss objective in terms of support instances as

$$\hat{g}_{rank}(\mathbf{W}) = \frac{\lambda}{2}||\mathbf{W}||^2 + \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|Y_i||\bar{Y}_i|}\sum_{j\in Y_i, k\in\bar{Y}_i}\max\{0, 1 + \mathbf{w}_k \cdot \hat{\mathbf{x}}_{i,k} - \mathbf{w}_j \cdot \hat{\mathbf{x}}_{i,j}\}$$

If $\hat{\mathbf{x}}_{i,j}$ is constant, this objective is convex. However, using the `max` and `softmax` models, $\hat{\mathbf{x}}_{i,j}$ is a function of $\mathbf{w}_j$, and the objective is non-convex. We can still use convex optimization techniques by alternating between updating the support and keeping the support constant while optimizing the objective. The MI-SVM algorithm for MIL (single-label) also uses the max function to connect instance and bag labels, and alternates between computing the support and optimizing the resulting objective [1].

4.2.1 Sub-Gradient Descent

To optimize $\hat{g}_{rank}$, we use a sub-gradient descent method similar to Pegasos, an algorithm for training linear two-class SVMs [19]. The Pegasos algorithm is based on a general framework for optimizing regularized convex objectives [18]. This framework can be applied to convex optimization problems in the form:

$$\min_{\mathbf{W}\in S} g(\mathbf{W}) \text{ where } g(\mathbf{W}) = \frac{\lambda}{2}||\mathbf{W}||^2 + loss(\mathbf{W})$$

For such problems, the following algorithm can be applied:

Projected Sub-Gradient Algorithm:

Initialize $\mathbf{W}^0 \in S$
for $t = 1, 2, \ldots, T$:
  Compute a sub-gradient $\mathbf{V} \in \partial g(\mathbf{W}^{t-1})$
  $\mathbf{W}^t \leftarrow P[\mathbf{W}^{t-1} - \frac{1}{\lambda t}\mathbf{V}]$

The feasible space is S; $P[\mathbf{W}] = \arg\min_{\mathbf{W}'\in S} ||\mathbf{W} - \mathbf{W}'||^2$ is a projection back into the feasible space. Note that when $S = \{\mathbf{W} : ||\mathbf{W}|| \leq r\}$, the projection simplifies to $P[\mathbf{W}] = \min\{1, \frac{r}{||\mathbf{W}||}\}\mathbf{W}$ (this will be the case for our objective).

This algorithm can be considered an example of the projected sub-gradient method [2] with learning rate $\frac{1}{\lambda t}$. Let $\mathbf{W}^*$ be the optimal solution i.e. $\mathbf{W}^* = \arg\min_{\mathbf{W}\in S} g(\mathbf{W})$. Shalev-Shwartz and Singer [18] showed the following convergence

rate for this algorithm,

$$\min_t g(\mathbf{W}^t) \leq g(\mathbf{W}^*) + O(\frac{\log T}{T}\frac{L}{\lambda})$$

where L is a constant bounding the magnitude of the sub-gradient i.e. $\forall t, \frac{1}{2}||\mathbf{V}||^2 \leq L$. For practical purposes, the number of iterations of sub-gradient descent T is small enough to treat $\log T$ as constant, hence to obtain a solution that is within ϵ of optimal, it suffices to run $T \approx O(\frac{L}{\lambda\epsilon})$ iterations of the above algorithm [18].

Treating the support instances as constant, we can apply the Pegasos framework for sub-gradient descent to minimize the rank-loss objective $\hat{g}_{rank}$. We denote the component of the sub-gradient corresponding to $\mathbf{w}_q$ as $\mathbf{v}_q^{rank}$, where $q = 1, \ldots, c$. The full sub-gradient is $\mathbf{V}^{rank} = [\mathbf{v}_1^{rank}, \ldots, \mathbf{v}_c^{rank}]$.

$$\mathbf{v}_q^{rank} = \lambda\mathbf{w}_q + \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|Y_i||\bar{Y}_i|}\sum_{j\in Y_i, k\in\bar{Y}_i} \tag{5}$$

$$I[1 + \mathbf{w}_k \cdot \hat{\mathbf{x}}_{i,k} > \mathbf{w}_j \cdot \hat{\mathbf{x}}_{i,j}] \begin{cases} \hat{\mathbf{x}}_{i,q} & \text{if } q = k \\ -\hat{\mathbf{x}}_{i,q} & \text{if } q = j \\ 0 & \text{otherwise} \end{cases}$$

In order to prove sub-gradient descent converges, we must establish L, the bound on the sub-gradient. We begin by showing the following Lemma:

Lemma 1: Consider any objective of the form $g(\mathbf{W}) = \frac{\lambda}{2}||\mathbf{W}||^2 + loss(\mathbf{W})$, such that $loss(\mathbf{W}) \geq 0$ and $loss(\mathbf{0}) = 1$. Let the optimal solution be $\mathbf{W}^* = \arg\min_{\mathbf{W}} g(\mathbf{W})$. Then $||\mathbf{W}^*||^2 \leq \frac{2}{\lambda}$.

Proof: The optimal solution must be at least as good as $\mathbf{W} = \mathbf{0}$, therefore $g(\mathbf{W}^*) \leq 1$. Furthermore, $loss(\mathbf{W}^*) \leq 1$ (assuming the contrary implies $g(\mathbf{W}^*) > 1$, which is a contradiction). Because the loss is non-negative, $0 \leq loss(\mathbf{W}^*) \leq 1$. We will use this property to finish the proof:

$$\begin{aligned} g(\mathbf{W}^*) = \frac{\lambda}{2}||\mathbf{W}^*||^2 + loss(\mathbf{W}^*) &\leq 1 \\ \frac{\lambda}{2}||\mathbf{W}^*||^2 &\leq 1 - loss(\mathbf{W}^*) \leq 1 \\ ||\mathbf{W}^*||^2 &\leq \frac{2}{\lambda} \end{aligned}$$

The $\hat{g}_{rank}$ objective satisfies the criteria of Lemma 1, so we can replace unconstrained minimization of $\hat{g}_{rank}$ with minimization restricted to the set $S = \{\mathbf{W} : ||\mathbf{W}||^2 \leq \frac{2}{\lambda}\}$ without changing the solution. It is also necessary to bound the magnitude of an instance feature vector, $||\mathbf{x}|| \leq R$. Note that $\mathbf{W} \in S$ implies $||\lambda\mathbf{w}_q|| \leq \lambda\sqrt{\frac{2}{\lambda}} = \sqrt{2\lambda}$ (using Lemma 1). Also, $\frac{1}{|Y_i||\bar{Y}_i|} \leq \frac{1}{c-1}$ provided $Y_i \neq \mathcal{Y}$ and $Y_i \neq \{\}$ (if this is not true, the objective is undefined). Using the above, we can derive the bound L:

$$\begin{aligned} ||\mathbf{v}_q^{rank}|| &\leq \sqrt{2\lambda} + R \\ ||\mathbf{v}_q^{rank}||^2 &\leq \left(\sqrt{2\lambda} + R\right)^2 \\ ||\mathbf{V}^{rank}||^2 &= \sum_{j=1}^{c}||\mathbf{v}_j^{rank}||^2 \\ ||\mathbf{V}^{rank}||^2 &\leq c\left(\sqrt{2\lambda} + R\right)^2 \end{aligned}$$

Therefore $L = \frac{c}{2}\left(\sqrt{2\lambda} + R\right)^2$ suffices.

4.2.2 Runtime of Sub-Gradient Descent

To compute $\mathbf{V}^{rank}$, one could compute $\mathbf{v}_1^{rank}, \ldots, \mathbf{v}_c^{rank}$ from (5), but this will take $O(nc^3)$ time; where n is the number of bags and c is the number of classes. We give an $O(nc^2)$ algorithm below:

Rank-Loss Sub-Gradient Algorithm:

```
for j = 1, ..., c :
    v_j^rank ← λ w_j
for i = 1, ..., n; j ∈ Y_i, k ∈ Ȳ_i :
    if 1 + w_k · x̂_{i,k} > w_j · x̂_{i,j} :
        v_j^rank ← v_j^rank − x̂_{i,j} / (n|Y_i||Ȳ_i|)
        v_k^rank ← v_k^rank + x̂_{i,k} / (n|Y_i||Ȳ_i|)
```

Running $T = O(\frac{L}{\lambda\epsilon})$ iterations with a runtime of $O(nc^2)$ per iteration of sub-gradient descent gives an upper bound of $O(\frac{nc^3}{\epsilon\sqrt{\lambda}})$ time to find a solution within ϵ of optimal, i.e. linear in the number of bags and $\frac{1}{\epsilon}$. In practice, $T = 100$ iterations is sufficient for many datasets (Fig. 1).

4.2.3 Updating the Support Instances

For the `max` and `softmax` models, we run K phases, where each phase consists of updating the support, then running T iterations of projected sub-gradient descent. For the first phase, we start with $\mathbf{W} = \mathbf{0}$ and compute the support as the average of the instances in each bag (to compute the support for the `max` and `softmax` models, we require some prior non-zero $\mathbf{W}$). Going from one phase to the next, we use the final weights from the earlier run of sub-gradient descent as the initial weights for the next run. Note that convergence analysis only applies to the sub-gradient descent part of each phase (it does not describe changes in the support over multiple phases). We summarize the proposed algorithm below.

Rank-Loss Support Instance Machine Algorithm:

```
for phase = 1, ..., K:
    if phase = 1 :
        W ← 0
        initialize x̂_{i,j} ← (1/n_i) Σ_{x_{i,q} ∈ X_i} x_{i,q}
    else:
        compute x̂_{i,j} using max (3) or softmax (4)
    for t = 1, ..., T:
        compute a sub-gradient V^rank ∈ ∂ĝ_rank(W)
        W ← W − (1/(λt)) V^rank
        W ← min{1, sqrt(2/λ)/||W||} W
```

Computing the support takes $O(m)$ time where m is the total number of instances, and this must be done K times.

5. EXPERIMENTS

We conduct experiments on artificial and real-world MIML datasets. The goals of our experiments are to (1) compare the proposed rank-loss objective with alternative Hamming loss and ambiguous loss; and (2) compare the `max` and `softmax` aggregation models.

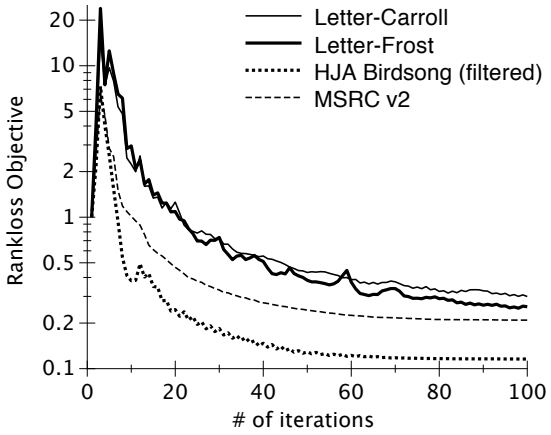


Figure 1: Convergence of sub-gradient descent for rank loss, `average` model (transductive).

Table 1: MIML datasets used in our experiments.

Dataset	Classes	Dimension	Bags	Instances
HJA Birdsong	13	38	548	10,232
MSRC v2	23	48	591	1,758
Letter-Carroll	26	16	166	717
Letter-Frost	26	16	144	565

5.1 Experimental Setup

We describe the setup of our experiments below.

5.1.1 *Transductive vs. Inductive*

We consider instance annotation in two different settings: transductive and inductive. In the transductive setting, the goal is to predict the instance labels for bags with known label sets. In this setting, the instance-level classifiers can only predict labels that appear in the bag label set. Formally, the instance classifier in the transductive setting is $f(\mathbf{x}_{i,q}) = \arg\max_{j \in Y_i} f_j(\mathbf{x}_{i,q})$. In the inductive setting, the goal is to predict instance labels in previously unseen bags (with unknown label sets). There is no restriction on which label an instance may be given in this case.

For both modes, we compute classifier accuracy as the fraction of instances correctly classified. For the inductive mode, we run 10-fold cross-validation and report average accuracy $\pm$ standard deviation in accuracy between folds.

5.1.2 *Datasets*

Table 1 summarizes the properties of each dataset used in our experiments. All of these datasets are available online.[1]

HJA Bird Song. Our collaborators have collected audio recordings of bird song at the H. J. Andrews (HJA) Experimental Forest, using unattended microphones. Our goal is to automatically identify the species of bird responsible for each utterance in these recordings, thereby generating an automatic acoustic survey of bird populations. This problem is a natural fit for the MIML instance annotation framework. We treat a 10-second audio recording as a bag with labels

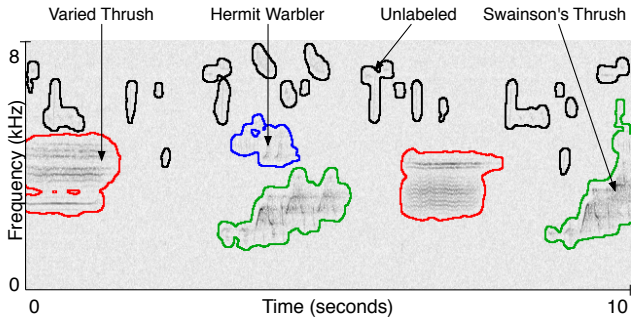


Figure 2: An example spectrogram from the HJA Birdsong dataset. This spectrogram corresponds to one bag. Each outlined region is an instance.

corresponding to the set of species present in the recording. The instances are segments in a spectrogram. A spectrogram is a graph of the spectrum of a signal as a function of time (computed by applying the Fast Fourier Transform to successive overlapping frames of the audio signal). Figure 2 shows an example spectrogram for a 10-second audio recording containing several species of birds.

Starting with a 10-second audio recording, we first convert it to a spectrogram. A series of preprocessing steps are then applied to the spectrogram to reduce noise, and to identify bird song segments in the audio [15]. Each segment is considered an instance and described by a 38-dimensional feature vector characterizing the shape of the segment, its time and frequency profile statistics, and a histogram of gradients.

This dataset contains 548 10-second recordings (bags), and a total of 10,232 segments (instances), of which 4,998 are labeled, and the rest are unlabeled. The available instance labels were provided by a human expert. Some instances were left unlabeled because they correspond to segmentation errors (i.e. noise rather than bird song), because they are too difficult to identify in the presence of other sounds, or because it is extremely time-consuming to produce these labels. The bag-level label sets are formed by taking the union of the instance labels (not including any unlabeled instances).

The presence of the unlabeled instances which are not accounted for by the bag label set presents an additional challenge for instance annotation. To evaluate how well various algorithms handle this problem, we consider two variants of this dataset: "filtered" and "unfiltered". For the filtered variant, all of the unlabeled instances are removed, and in the unfiltered variant they are left in during the training process. In both variants, the accuracy is measured only on the labeled instances.

Image Dataset: MSRC v2. A subset of the Microsoft Research Cambridge (MSRC) image dataset[2] [23] named "v2" contains 591 images and 23 classes. The MSRC v2 dataset is useful for the instance annotation problem, because pixel-level labels are included (Fig. 3). Several authors used MSRC v2 in MIML experiments [27, 24, 22].

We construct a MIML dataset from MSRC v2 as follows: We treat each image as a bag. The bag label set is the list of all classes present in the ground-truth segmentation (i.e. the

[1] `http://web.engr.oregonstate.edu/~briggsf/kdd2012datasets`

[2] `http://research.microsoft.com/en-us/projects/objectclassrecognition/default.htm`

Figure 3: An image from MSRC v2 and the corresponding pixel-level labeling. The classes in this image are 'sky', 'trees, 'grass', 'body', and 'car'. The black regions are 'void'; we discard void regions.

union of the instance labels). The instances correspond to each contiguous region in the ground-truth segmentation (to simplify the experiment, we use the ground-truth segmentation rather than automatic segmentation). Each instance is described by a 16-dimensional histogram of gradients, and a 32-dimensional histogram of colors.

Synthetic MIML Datasets. Limited availability of MIML datasets with instance labels has been a barrier to studying instance annotation (because instance labels are needed to evaluate accuracy). Using the Letter Recognition dataset [11] from the UCI Machine Learning repository, we construct two synthetic MIML datasets. The Letter Recognition dataset consists of 20,000 instances with 16-dimensional features, and 26 classes. Note that randomly forming the bags will not be realistic because real-world MIML problems often have correlations between labels. Instead, we generate datasets derived from the words in two poems, "Jabberwocky" [3], and "The Road Not Taken" [12]. We call these datasets Letter-Carroll and Letter-Frost. For each word in these poems, we create a bag, with instances corresponding to the letters in the word. For each instance, we sample (without replacement), an example from the Letter Recognition dataset with the corresponding letter. The bag-level labels are the union of the instance labels. For example, the word "diverged" is transformed into a bag with 8 instances, and the label set {d, i, v, e, r, g}.

Preprocessing. For all datasets, we apply the following preprocessing to the instance features. First, we transform each feature to the range $[0, 1]$. Next, we apply the same feature rescaling process used in the *Convex Learning from Partial Labels Toolbox* (for ALC [6]), which centers the data and scales each feature by $\frac{1}{\sqrt{\sum_{i=1}^{m} ||\mathbf{x}_i||^2}}$.

5.1.3 Base Line Methods

We compare our proposed Rank-Loss SIM methods with a Hamming-Loss SIM, and the Ambiguous Label Classification algorithm proposed by Cour et al. [6]. As a reference, we also consider a SISL classifier, which will have the unfair advantage of learning directly from instance labels.

Hamming-Loss SIM. To compare Hamming loss to rank loss, we use the following Hamming-loss objective, with both `max` and `softmax` aggregation models:

$$g_{ham}(\mathbf{W}) = \frac{\lambda}{2}||\mathbf{W}||^2 + \frac{1}{nc}\sum_{i=1}^{n}\sum_{j=1}^{c}\max\{0, 1 - Y_i^j F_j(X_i)\}$$

or in terms of support instances,

$$\hat{g}_{ham}(\mathbf{W}) = \frac{\lambda}{2}||\mathbf{W}||^2 + \frac{1}{nc}\sum_{i=1}^{n}\sum_{j=1}^{c}\max\{0, 1 - Y_i^j \mathbf{w}_j \cdot \hat{\mathbf{x}}_{i,j}\}$$

We use a similar projected sub-gradient descent algorithm to optimize this objective (and update the support in the same way as for the rank-loss objective over multiple phases). The conditions for Lemma 1 are also met by this objective. We compute the following sub-gradient

$$\mathbf{v}_q^{ham} = \lambda \mathbf{w}_q - \frac{1}{nc}\sum_{i=1}^{n} I[Y_i^q \mathbf{w}_q \cdot \hat{\mathbf{x}}_{i,q} < 1] Y_i^q \hat{\mathbf{x}}_{i,q}$$

In this case, the bound on the magnitude of the sub-gradient is $L = \frac{c}{2}\left(\sqrt{2\lambda} + \frac{R}{c}\right)^2$.

Ambiguous Label Classification (ALC). Cour et al. [5, 6] proposed an SVM formulation for the ALC problem. We compare our proposed method to Cour's ALC algorithm because they both learn one linear model per class $f_j(\mathbf{x}) = \mathbf{w}_j \cdot \mathbf{x}$ and predict the instance label as $\arg\max_j f_j(\mathbf{x})$, and both use an L_2 regularized loss function. The primary difference in Cour's ALC method is that the loss function is designed for use with ALC data (instead of the bag-level loss functions we use for MIML data). The ALC loss function is a convex upper bound to the 0/1 loss with respect to the true (unknown) instance labels, $L(f, \mathbf{x}, Y) =$

$$\max\{0, 1 - \frac{1}{|Y|}\sum_{j \in Y} f_j(\mathbf{x})\}^2 + \sum_{j \notin Y}\max\{0, 1 + f_j(\mathbf{x})\}^2$$

Minimizing regularized ambiguous loss can be converted into an equivalent SISL SVM problem with squared hinge-loss, and solved using an off-the-shelf linear SVM [10].

Comparison to SISL. We also run a SISL SVM for the inductive setting, whose performance can be interpreted as an empirical upper bound for inductive instance annotation because it is trained using unambiguously labeled instances. For this experiment, we use LIBSVM [4] with a linear kernel. Note that LIBSVM uses one linear model for each pair of classes rather than one for each class. For the HJA Birdsong dataset, we only run the SISL SVM on the filtered variant, because we cannot use the unlabeled instances.

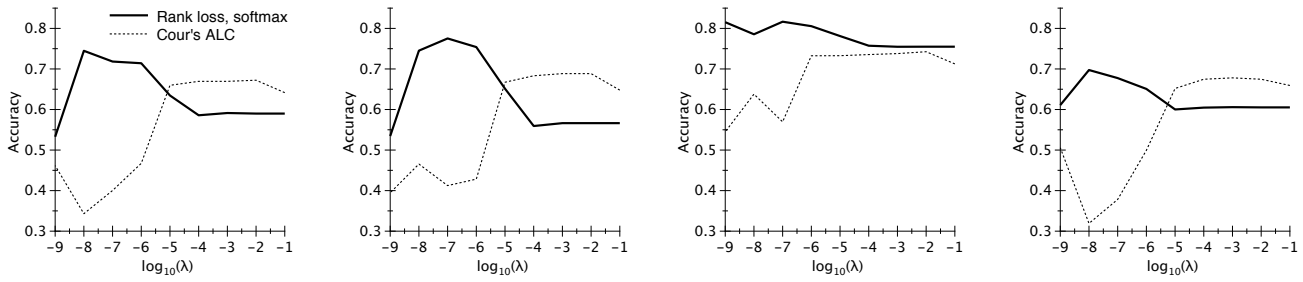
5.1.4 Parameter Selection

The Hamming and rank-loss objectives have a regularization parameter λ. Similarly, Cour's ALC method has a regularization parameter C, which achieves roughly the same effect when $C = \frac{1}{\lambda}$. To obtain a fair comparison of different methods regardless of the parameter settings, we repeat all experiments for each value of $\lambda \in \{10^{-1}, 10^{-2}, \ldots, 10^{-9}\}$, and corresponding C values, and report the maximum accuracy achieved by each method.

In practice one could use cross-validation (on the bag-level labels) to select the regularization parameter. However, our results suggest that it maybe sufficient to use a default parameter for many datasets. For example, setting λ to 10^{-7} or 10^{-8} tends to work well for all datasets that we considered (Fig. 4). This approach is consistent with prior work using Cour's ALC method where a fixed value of C is used in all experiments.[3]

For the SISL SVM, the regularization parameter C is optimized by nested 10-fold cross-validation (within each fold

[3]Cour et al. used $C = 10^3$ for all experiments in prior work. Our results support this choice.



(a) Letter-Carroll (b) Letter-Frost (c) HJA Birdsong (filtered) (d) MSRC v2

Figure 4: Accuracy vs. regularization parameter (transductive).

of 10-fold cross validation, we run 10-fold cross validation in the training set to select the parameter). We search over the range $C \in \{10^1, 10^2, \ldots, 10^7\}$.

For the SIM algorithms, we use $K = 10$ phases, with $T = 100$ iterations of sub-gradient descent in each iteration. Figure 1 shows the rank-loss objective vs. number of iterations of sub-gradient descent with a fixed set of support instances on each dataset. These results show that most of the improvement in the objective occurs within 100 iterations. Note the convergence analysis does not guarantee the objective will decrease monotonically (only that an upper bound decreases monotonically).

5.2 Results and Discussions

Table 2 lists results in the transductive setting, and Table 3 lists results in the inductive setting.

Comparing Different Loss Functions. First, we focus on the transductive setting. Rank-loss methods generally outperform Hamming loss and ambiguous-loss methods. In particular, when used with the same aggregation model, rank loss consistently outperforms Hamming loss on all datasets. The performance differences of these two methods for the `softmax` aggregation model range from 4% for HJA filtered to 15% for Letter-Carroll, and are even more pronounced for the `max` model. These results support our claim that rank loss is more appropriate than Hamming loss for instance annotation. Rank loss with either aggregation model also consistently outperforms ambiguous loss (ALC) in the transductive setting.

In the inductive setting, the performance generally is lower than the transductive setting, which is expected because in the transductive setting predictions are made with extra bag-level label information. However, rank-loss methods still achieve the best overall performance (excluding SISL SVM, which learns from instance labels). The performance of the Hamming-loss based methods is degraded most severely (possibly because Hamming loss does not calibrate scores between classes, so its predictions become less reliable without the restriction on which classes may be selected imposed by the bag label sets).

Comparing Different Aggregation Models. The `softmax` model generally outperforms `max` for both rank loss and Hamming-loss objectives. In fact, rank loss with `softmax` achieves the best performance for all five datasets in both settings (except for the inductive HJA Birdsong filtered dataset, where the difference in accuracy between rank loss with `softmax` and ALC is within the margin of uncertainty). The difference between the two aggregation models is more pronounced for the Hamming-loss objective.

The Effect of Unlabeled Instances. Comparing the filtered and unfiltered variants of the HJA dataset, all methods suffer some accuracy degradation when we include the unlabeled instances. This is not surprising as these unlabeled instances introduce noise both at the instance level (because they may not correspond to any of the defined classes) and at the bag level (because the label set may be incomplete). The rank loss / `softmax` method suffers less than other algorithms in the presence of the unlabeled instances. For example, in the transductive setting the difference in accuracy is 0.4% for rank loss / `softmax` and 6.4% for Cour's ALC. Note that ALC assumes that every instance is associated with one of the labels of the bag. This assumption is violated by the unlabeled instances that are not accounted for by the bag label set, making ALC more sensitive to the presence of such instances. The `softmax` model suffers less than the `max` model, possibly due to the fact that `max` can mistakenly select the unlabeled instances as support instances.

To summarize, we observe that the rank loss / `softmax` method achieves the best overall performance. This result is verified in both transductive and inductive settings. The rank loss / `softmax` method is also the most robust in the presence of instance and label noise introduced by the unlabeled instances in the HJA Birdsong dataset.

6. RELATED WORK

MIML algorithms are developed under multiple frameworks, some of which naturally lend themselves to instance annotation. One such framework is graphical models, which have been previously used to perform bag and instance-level classification. Such models often treat instance labels as latent variables. Inference over such models allows the classification of instances. While a variety of algorithms exists, we highlight some representative examples of recent work. Dirichlet-Bernoulli Alignment [26] and the Exponential Multinomial Mixture model [25] are topic models for MIML datasets and use variational inference to perform instance labeling. Zha et al. [27] proposed the MLMIL algorithm, a conditional random field model for MIML image annotation that uses Gibbs sampling to infer instance labels. Du et al. [8] propose another application of graphical models to simultaneous image annotation and segmentation.

Table 2: Transductive setting accuracy.

Algorithm	Letter-Carroll	Letter-Frost	HJA Birdsong (filtered)	(unfiltered)	MSRC v2
Rank loss, `softmax`	0.745	0.775	0.817	0.813	0.697
Rank loss, `max`	0.728	0.773	0.809	0.779	0.697
Hamming loss, `softmax`	0.591	0.588	0.776	0.755	0.614
Hamming loss, `max`	0.459	0.48	0.709	0.59	0.55
Cour's ALC	0.672	0.688	0.742	0.678	0.678

Table 3: Inductive mode, accuracy ± standard deviation over 10-fold cross-validation.

Algorithm	Letter-Carroll	Letter-Frost	HJA Birdsong (filtered)	(unfiltered)	MSRC v2
Rank loss, `softmax`	0.540 ± 0.066	0.575 ± 0.050	0.619 ± 0.042	0.555 ± 0.053	0.467 ± 0.044
Rank loss, `max`	0.528 ± 0.049	0.561 ± 0.051	0.590 ± 0.044	0.525 ± 0.071	0.439 ± 0.043
Hamming loss, `softmax`	0.347 ± 0.065	0.338 ± 0.041	0.479 ± 0.052	0.438 ± 0.052	0.337 ± 0.034
Hamming loss, `max`	0.142 ± 0.041	0.150 ± 0.044	0.243 ± 0.052	0.119 ± 0.027	0.179 ± 0.064
Cour's ALC	0.474 ± 0.063	0.506 ± 0.063	0.621 ± 0.038	0.542 ± 0.039	0.431 ± 0.036
SISL SVM	0.772 ± 0.049	0.753 ± 0.038	0.772 ± 0.032	–	0.638 ± 0.045

While graphical models offer intuitive probabilistic interpretation, the computational complexity of inference in such models is one of the standing challenges. In this paper, we focus on another class of MIML approaches based on regularized loss minimization. Hamming-Loss SIM can be viewed as alternative approach for optimizing a similar objective to the M^{3}MIML algorithm [28] (which learns an instance-level model, but was not designed for instance annotation). Also note that Cour's ALC method follows the same framework of regularized loss minimization (but using a loss function designed for ALC).

There are a small number of other works that address MIML instance annotation [21, 22]. Vijayanarasimhan and Grauman [22] developed a MIML SVM that learns a bag-level model with a set kernel (it does not learn a model of the instance feature space). Their algorithm makes predictions at either the bag or instance level by treating an instance as a bag of one instance. Vezhnevets and Buhmann [21] proposed an algorithm for instance annotation in MIML data where images are represented as a bag of pixels. Their algorithm alternates between sampling instance labels from an estimated distribution, and training an ensemble of decision trees on the sampled labels. Similarly, Nguyen [16] proposed a MIML SVM algorithm which alternates between assigning instance labels and maximizing margin given the assigned labels (although they did not conduct experiments on instance annotation).

7. CONCLUSION AND FUTURE WORK

We introduce Rank-Loss Support Instance Machines for instance annotation of MIML datasets. The key to our approach is a loss function that is measured at the bag-level and encourages the correct ranking of classes such that classes present in the label set of a bag score higher than classes that are not present. This general objective can be instantiated with different aggregation models including the previously used `max` model or a newly proposed `softmax` model. Both models can be viewed as representing each bag with a support instance for each class. We alternate between computing this support, and optimizing a convex rank-loss objective using an efficient primal sub-gradient descent method. Experiments demonstrate that Rank-Loss SIM achieves higher accuracy than similar Hamming or ambiguous-loss based methods, and the `softmax` aggregation model achieves higher accuracy than the `max` model.

Our empirical evaluation focuses on methods based on regularized loss minimization. Comparisons to other types of methods including graphical models will be a topic of future work. We only consider linear models in this work, but our proposed methods could be extended to use kernels for non-linear classification. One possibility is to use the standard kernel trick with a dual optimization method. Shalev-Shwartz et al. [19] proposed a way of kernelizing Pegasos that could be applied to our algorithms. A third possibility proposed by Rahimi and Recht [17] is to map the original features into a feature space such that inner products in the new space are approximately equal to a kernel (e.g., a radial basis function kernel). Further work is needed to determine which of these methods works well for instance annotation, and in what circumstances.

8. ACKNOWLEDGMENTS

This work is partially funded by the Ecosystems Informatics IGERT program via NSF grant DGE 0333257, NSF grant 1055113 to X. Z. F., and the College of Engineering, Oregon State University. We would also like to thank Matthew Betts, Sarah Frey, Adam Hadley, and Jay Sexsmith for their help in collecting HJA data, Iris Koski for labeling the data, Katie Wolf for her work on noise reduction, and Lawrence Neal for his work on segmentation.

9. REFERENCES

[1] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. *Advances in Neural Information Processing Systems*, 15:561–568, 2002.

[2] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge Univ Pr, 2004.

[3] L. Carroll. *Through the looking-glass : and what Alice found there*. 1896.

[4] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001.

[5] T. Cour, B. Sapp, C. Jordan, and B. Taskar. Learning from ambiguously labeled images. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 919–926. IEEE, 2009.

[6] T. Cour, B. Sapp, and B. Taskar. Learning from partial labels. *Journal of Machine Learning Research*, 12:1225–1261, 2011.

[7] T. Dietterich, R. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997.

[8] L. Du, L. Ren, D. Dunson, and L. Carin. A bayesian model for simultaneous image clustering, annotation and object segmentation. *Advances in Neural Information Processing Systems*, 22:486–494, 2009.

[9] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. *Advances in Neural Information Processing Systems*, 14:681–687, 2001.

[10] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[11] P. W. Frey and D. J. Slate. Letter recognition using holland-style adaptive classifiers. *Machine Learning*, 6:161, 1991.

[12] R. Frost. *Mountain Interval*. 1916.

[13] E. Hüllermeier and J. Beringer. Learning from ambiguously labeled examples. *Intelligent Data Analysis*, 10(5):419–439, 2006.

[14] Y. Li, S. Ji, S. Kumar, J. Ye, and Z. Zhou. Drosophila gene expression pattern annotation through multi-instance multi-label learning. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, CA*, 2009.

[15] L. Neal, F. Briggs, R. Raich, and X. Fern. Time-frequency segmentation of bird song in noisy acoustic environments. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing*, 2011.

[16] N. Nguyen. A new svm approach to multi-instance multi-label learning. In *Tenth IEEE International Conference on Data Mining. ICDM'10*, pages 384–392, 2010.

[17] A. Rahimi and B. Recht. Random features for large-scale kernel machines. *jAdvances in Neural Information Processing Systems*, 20:1177–1184, 2007.

[18] S. Shalev-Shwartz and Y. Singer. Logarithmic regret algorithms for strongly convex repeated games. *The Hebrew University, Technical Report*, 2007.

[19] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *Proceedings of the 24th international conference on Machine learning*, pages 807–814. ACM, 2007.

[20] C. Shen, J. Jiao, B. Wang, and Y. Yang. Multi-Instance Multi-Label Learning For Automatic Tag Recommendation. In *Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics (SMC 2009)*, 2009.

[21] A. Vezhnevets, J. Buhmann, and E. Zurich. Towards Weakly Supervised Semantic Segmentation by Means of Multiple Instance and Multitask Learning. In *Conference on Computer Vision and Pattern Recognition*, 2010.

[22] S. Vijayanarasimhan and K. Grauman. What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations. 2009.

[23] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, pages 1800–1807, 2005.

[24] O. Yakhnenko. *Learning from Text and Images: Generative and Discriminative Models for Partially Labeled Data*. PhD thesis, Iowa State University, 2009.

[25] S. Yang, J. Bian, and H. Zha. Hybrid Generative/Discriminative Learning for Automatic Image Annotation. In *Conference on Uncertainty in Artificial Intelligence*, 2010.

[26] S. Yang, H. Zha, and B. Hu. Dirichlet-Bernoulli Alignment: A Generative Model for Multi-Class Multi-Label Multi-Instance Corpora. 2010.

[27] Z. Zha, X. Hua, T. Mei, J. Wang, G. Qi, and Z. Wang. Joint multi-label multi-instance learning for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–8, 2008.

[28] M. Zhang and Z. Zhou. M3MIML: A maximum margin method for multi-instance multi-label learning. In *Eighth IEEE International Conference on Data Mining. ICDM'08*, pages 688–697, 2008.

[29] Z. Zhou. Multi-instance learning: A survey. Technical report, National Laboratory for Novel Software Technology, Nanjing University, 2010.

[30] Z. Zhou and M. Zhang. Multi-instance multi-label learning with application to scene classification. *Advances in Neural Information Processing Systems*, 19:1609, 2007.

[31] Z. Zhou, M. Zhang, S. Huang, and Y. Li. Multi-instance multi-label learning. *Artificial Intelligence*, 176(1):2291–2320, 2012.