Scalable Optimization of Multivariate Performance Measures in Multi-instance Multi-label Learning

A. Aggarwal*, S. Ghoshal* Ankith M. S.* S. Sinha* G. Ramakrishnan* P. Kar†, P. Jain‡

Abstract

The problem of multi-instance multi-label learning (MIML) requires a bag of instances to be assigned a set of labels most relevant to the bag as a whole. The problem finds numerous applications in machine learning, computer vision, and natural language processing settings where only partial or distant supervision is available. We present a novel method for optimizing multivariate performance measures in the MIML setting. Our approach MIML^{perf} uses a novel plug-in technique and offers a seamless way to optimize a vast variety of performance measures such as macro and micro-F measure, average precision, etc which are performance measures of choice in multi-label learning domains. MIML^{perf} offers two key benefits over the state of the art. Firstly, across a diverse range of benchmark tasks, ranging from relation extraction to text categorization and scene classification, MIML^{perf} offers superior performance as compared to state of the art methods designed specifically for these tasks. Secondly, MIML^{perf} operates with significantly reduced running times as compared to other methods, often by an order of magnitude.

1 Introduction

The paucity of labeled data has fueled much interest in learning paradigms with partial or *distant* supervision. The task of Multi-instance Multi-label (MIML) learning is one such paradigm that has received much attention in areas such as relation extraction (RE), image classification etc. The problem requires a bag of instances to be assigned a set of labels most relevant to the bag as a whole. To take an example, in the RE setting, we are interested in identifying relations held by entities being discussed in a body of text. The problem is pretty straightforward in the presence of sentence level annotations *i.e.*, indications of which sentences discuss specific relations. However, this requirement is a prohibitive one, especially when the number of relations runs into the hundreds or thousands. Consequently, the problem is often posed as that of learning under distant supervision (Haffari, Nagesh, and Ramakrishnan 2015; Hoffmann et al. 2011a; Surdeanu et al. 2012a; Zeng et al. 2015) wherein one is provided only with relational facts in a database along with a non-annotated corpus that overall, supports most of the facts.

The general MIML problem has been explored through reductions to single instance or single label learning in the past (Zhou and Zhang 2006; Zhang and Zhou 2008). The RE problem has enjoyed more focused attention and several diverse approaches have been proposed for this problem such as structural risk minimization Hoffmann et. al. (2011a), neural networks (Zeng et al. 2015) and graphical models (Surdeanu et al. 2012a).

Existing algorithms for the MIML problem suffer from two main drawbacks 1) Although performance measures such as F-measure and average precision are standard for evaluation, the algorithms do not seek to optimize them directly, instead choosing to adopt heuristics that encourage good performance. An exception is the work of Haffari et al. (2015) who attempt to directly optimize the F-micro measure. 2) The algorithms are often expensive in terms of training time and cannot scale to large, web-scale datasets.

We address both these issues by developing MIML^{perf}, a *plug-in* classifier for the MIML problem. MIML^{perf} directly tries to optimize complex performance measures such as macro and micro-F measure in a scalable manner. The method excels over its competitors in its abilty to predict rare labels correctly. A notable feature of MIML^{perf} is its streaming nature, that allows it to be executed by making several passes over the data without the need for storing the entire dataset in memory. We rigorously benchmark MIML^{perf} to establish that 1) it offers far greater label extraction accuracies on RE than specialized methods for the problem, 2) it also outperforms state-of-the-art MIML approaches on text categorization and scene classification problems, and 3) it can offer orders of magnitude faster running times.

2 Related Work

Completely supervised approaches, such as those for RE (GuoDong et al. 2005; Surdeanu and Ciaramita 2007) have limited application owning to their requirement of fully annotated data which is expensive. Of the alternate paradigms, that of *distant supervision* has found much appeal in both RE and MIML literature. For the RE problem, this paradigm shift due to (Mintz et al. 2009) fueled much interest with (Riedel, Yao, and McCallum 2010; Yao, Riedel, and McCal-

*IIT Bombay, {apoorvaggarwal21, sandip.iem, suhit.iitb, msankith}@gmail.com, ganesh@cse.iitb.ac.in

†IIT Kanpur, purushot@cse.iitk.ac.in

‡Microsoft Research, prajain@microsoft.com

lum 2010) modeling the problem as that of mapping entity pairs in the database to their mentions in the corpus, in other words, a multi-instance single-label learning problem. Soon, (Hoffmann et al. 2011b; Surdeanu et al. 2012b) generalized this to allow entity pairs to participate in multiple relations, thus completing the MIML abstraction. A variety of techniques have been applied to this problem over the years. Recently (Zeng et al. 2015) applied Piecewise Convolutional Neural Networks (PCNNs) to the RE problem. However, these models were often trained by optimizing performance measures such as conditional log-likelihood (Surdeanu et al. 2012a), error rate, or bag level entropy (Zeng et al. 2015) that are not directly related to the measures actually used for evaluation such as F-measure and average precision.

The area of learning structured prediction models becomes relevant in this context since MIML requires predicting a structured array of labels. The area has seen the development of powerful large-margin methods (Taskar, Guestrin, and Koller 2003) which can incorporate hidden variables (Wang and Mori 2011; Felzenszwalb et al. 2010; Yu and Joachims 2009), which is useful since MIML admits elegant formulations as a latent variable learning problem, as well as optimize non-decomposable performance measures such as F-measure (Ranjbar et al. 2013; Tarlow and Zemel 2012; Rosenfeld et al. 2014; Keshet 2014).

However, the only direct application of these techniques to optimize non-decomposable performance measures in the MIML setting is in the work of Haffari et al. (2015), who optimize the micro averaged F-measure in the RE setting. Their optimization algorithm interleaves the Concave-Convex Procedure (CCCP) (Yuille and Rangarajan 2001) to populate latent variables using dual decomposition (Komodakis, Paragios, and Tziritas 2011; Rush and Collins 2012). This factorizes the hard optimization problem into smaller independent sub-problems over the training instances. Despite such optimization tricks and the use of heuristic local search methods replacing the exhaustive search of (Joachims 2005; Ranjbar, Vahdat, and Mori 2012)), their approach suffers from two drawbacks: (i) the approach is slow and does not scale to large datasets and (ii) it does not perform well for the *heavy tail* of rare classes with small class priors.

There also has been progress in developing general purpose algorithms for the MIML problem (Zhou and Zhang 2006; Zhang and Zhou 2008) that have been applied to text and image (scene) classification tasks. The work of (Zhou and Zhang 2006) provides two methods, *viz.*, MIMLBoost and MIMLSVM for the problem. MIMLBoost solves several multi-instance single label (MISL) problems and then converts each into Single Instance Single Label (SISL) problems using multi-instance boosting. MIMLSVM, on the other hand, converts the MIML problem into several Single Instance Multi Label (SIML) problems using k-medoids clustering that uses the Hausdorff distance between all pairs of points and then solves SIML problem using SVM. Both methods work on degenerate versions of MIML and hence can be lossy. The main bottleneck in the MIMLSVM approach is the extremely expensive computation of Hausdorff distance matrix between all pairs of points. As we shall see in Section 5, this causes this approach to be extremely slow. As a concluding note we point out that the the SIML approach of reducing the problem of predicting labels for bags to that of predicting labels for individual instances is very popular in RE (Zeng et al. 2015; Haffari, Nagesh, and Ramakrishnan 2015; Hoffmann et al. 2011a) and MIML algorithms. We will revisit this approach while describing the MIML^{perf} algorithm.

3 Problem Formulation

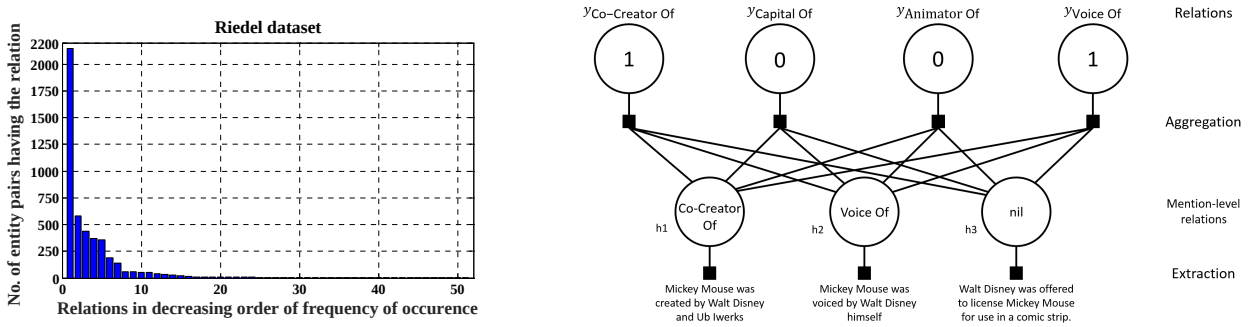
In the MIML setting, training data is presented as $\mathcal{D} := \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$ where $\mathbf{x}_i = \left\{\mathbf{x}_i^{(1)}, \ldots, \mathbf{x}_i^{(n_i)}\right\} \in \mathcal{X}$ is the i^{th} instance set (also called a *bag*) containing n_i instances and $\mathbf{y}_i = [y_{i,1}, y_{i,2}, \ldots, y_{i,L}] \in \mathcal{Y} = \{0,1\}^L$ is a vector of labels associated with $\mathbf{x}_i$. We will use $\mathbf{h}_i \in \{1, .., L, nil\}^{n_i}$ to denote the vector of *hidden* labels for $\mathbf{x}_i$. $h_{i,j}$ will encode whether the j^{th} instance in the bag $\mathbf{x}_i$ expresses one of the labels $\{1, \ldots, L\}$ or not $\{nil\}$. Denote $\mathbf{X} := \{\mathbf{x}_i\}_{i=1}^N \in \boldsymbol{\mathcal{X}}$ and $\mathbf{Y} := \{\mathbf{y}_i\}_{i=1}^N \in \boldsymbol{\mathcal{Y}}$. The above is illustrated well using an example from the RE domain. The goal in RE is to align facts to sentences in a large unlabeled corpus. The training data consists of *entity-pairs*, such as `(Walt Disney, Mickey Mouse)`. For each entity pair, we are given a set of sentences (also called *mentions*) that talk about these entities, and a set of relations known to be satisfied by the pair. For instance, `(Walt Disney, Mickey Mouse)` satisfy the relations `Co-creator-Of, Voice-Of` but not `Animator-Of` (Mickey was animated by Ub Iwerks).

The above can be cast as an MIML problem by letting $\mathbf{x}_i$ be the instance set or bag containing all mentions of the i-th entity pair and $\mathbf{y}_i$ be the relations satisfied by the pair. The hidden label vector $\mathbf{h}_i$ can be used to denote relations expressed by individual mentions. The set of L possible relations is extracted from a knowledge-base.

We wish to learn a function $\mathbf{f} : \mathcal{X} \to \mathcal{Y}$ that predicts labels for novel data points $\mathbf{f}(\mathbf{x}) = [f^1(\mathbf{x}), f^2(\mathbf{x}), \ldots, f^L(\mathbf{x})]$. Given a dataset $(\mathbf{X}, \mathbf{Y})$ containing N data points, let $\mathbf{Y}^j = [y_{1,j}, y_{2,j}, \ldots, y_{N,j}]$ encode which bags support label j. Similarly, let $\mathbf{f}^j(\mathbf{X}) = [f_j(\mathbf{x}_1), f_j(\mathbf{x}_2), \ldots, f_j(\mathbf{x})_N]$ be the vector of predictions for label j. Our learning process will be guided by a *performance measure* $\Delta : \boldsymbol{\mathcal{Y}} \times \boldsymbol{\mathcal{Y}} \to \mathbb{R}_+$.

The simplest of such performance measures is the class of *univariate* or *decomposable measures* that compute performance over a set of data points by simply averaging the performance on individual data points $\Delta(\mathbf{f}(\mathbf{X}), \mathbf{Y}) = \frac{1}{N}\sum_{i=1}^N \Delta(\mathbf{f}(\mathbf{x}_i), \mathbf{y}_i)$. Examples include the Hamming distance (Bi and Kwok 2013; Chen and Lin 2012), precision (Hsu et al. 2009; Weston, Bengio, and Usunier 2011), and recall (Steck 2010). Although convenient to work with and analyze, these performance measures are known to be ill suited in the presence of label imbalance or a heavy tailed label distribution (Koyejo et al. 2014; Narasimhan, Vaish, and Agarwal 2014) since they tend to neglect performance on rare labels. Incidentally, the heavy tail phenomenon is well documented in relation extraction settings (see Figure 1a).

Instead, the performance measures of choice in these situ-



(a) (b)

Figure 1: (Left) The Riedel dataset exhibits a heavy tail in its label distribution, most relations are extremely rare. (Right) Mention level hidden labels can be used to compute the active relations for the entity-pair `(Walt Disney, Mickey Mouse)`.

ations are the *multivariate performance measures* that force the predictor to do well on rare labels as well. These measures are typically *non-decomposable* as their evaluation does not decompose over individual points. In this paper, we focus on the family of F-measures that are popular for RE (Haffari, Nagesh, and Ramakrishnan 2015; Surdeanu et al. 2012a), as well as label imbalanced learning settings in general (Koyejo et al. 2014; Ye et al. 2012).

The *label-wise* Precision and Recall of a predictor measure the performance on each individual label

$$\text{PREC}^j(\mathbf{f};\mathbf{X},\mathbf{Y}) := \frac{\sum_{i=1}^n y_{i,j}\cdot \mathbf{f}^j(\mathbf{x}_i)}{\sum_{i=1}^n \mathbf{f}^j(\mathbf{x}_i)}$$
$$\text{REC}^j(\mathbf{f};\mathbf{X},\mathbf{Y}) := \frac{\sum_{i=1}^n y_{i,j}\cdot \mathbf{f}^j(\mathbf{x}_i)}{\sum_{i=1}^n y_{i,j}},$$

whereas the global Precision and Recall calculate the overall performance of $\mathbf{f}$ across labels

$$\text{PREC}(\mathbf{f};\mathbf{X},\mathbf{Y}) := \frac{\sum_{i=1}^n \sum_{j=1}^L y_{i,j}\cdot \mathbf{f}^j(\mathbf{x}_i)}{\sum_{i=1}^n \sum_{j=1}^L \mathbf{f}^j(\mathbf{x}_i)}$$
$$\text{REC}(\mathbf{f};\mathbf{X},\mathbf{Y}) := \frac{\sum_{i=1}^n \sum_{j=1}^L y_{i,j}\cdot \mathbf{f}^j(\mathbf{x}_i)}{\sum_{i=1}^n \sum_{j=1}^L y_{i,j}}$$

For any label j, we define the label-wise F-measure as $F_\beta^j(\mathbf{f};\mathbf{X},\mathbf{Y}) := \left(\frac{\beta}{\text{PREC}^j(\mathbf{f};\mathbf{X},\mathbf{Y})} + \frac{1-\beta}{\text{REC}^j(\mathbf{f};\mathbf{X},\mathbf{Y})}\right)^{-1}$, as well as the **Macro** and **Micro F-measure** as

$$F_\beta^{\text{macro}}(\mathbf{f};\mathbf{X},\mathbf{Y}) := \frac{1}{L}\sum_{j=1}^L F_\beta^j(\mathbf{f};\mathbf{X},\mathbf{Y}), \text{ and}$$
$$F_\beta^{\text{micro}}(\mathbf{f};\mathbf{X},\mathbf{Y}) := \left(\frac{\beta}{\text{PREC}(\mathbf{f};\mathbf{X},\mathbf{Y})} + \frac{1-\beta}{\text{REC}(\mathbf{f};\mathbf{X},\mathbf{Y})}\right)^{-1},$$

where $\beta = 0.5$ gives the standard F_1 measure. These performance measures, especially the macro F-measure, penalize predictors that perform poorly on rare labels. In the next section, we develop scalable techniques for optimizing multivariate performance measures such as F-measure variants in the MIML and RE settings.

4 Proposed Approach

Despite having attractive properties as discussed above, multivariate performance measures such as F-measure present challenges to learning algorithms. Due to their non-decomposability, classical algorithmic tools such as online and stochastic optimization, as well as analytical tools such as uniform convergence bounds are not readily applicable.

The recent years have seen a growing interest in the problem of optimizing non-decomposable performance measures, especially F-measure variants (Dembczyński et al. 2013; Narasimhan, Kar, and Jain 2015). However, past work is mostly restricted to binary classification. An exception is the recent work of Haffari et al. (2015) that utilizes the structural SVM approach (Joachims 2005) to optimize the F-micro measure in the RE setting. However, the approach is woefully non-scalable, struggling to cope with even a few thousand data points. This severely restricts its applicability to real life and production-grade problems.

This section will develop MIML^{perf}, a scalable tool for optimizing multivariate performance measures in the MIML setting. MIML^{perf} is based on a *plug-in* approach to classification. Plug-in classifiers have been studied in binary classification settings with great success (Kotłowski and Dembczynski 2015; Narasimhan, Vaish, and Agarwal 2014; Koyejo et al. 2014; Ye et al. 2012). However, to the best of our knowledge, plug-in approaches have not been studied in MIML or RE settings.

4.1 Plug-In Classifiers

Consider a simple binary classification problem where the task is to assign every data point $\mathbf{x} \in \mathcal{X}$, a binary label $y \in \{\pm 1\}$. Plug-in classifiers achieve this by first learning to predict *Class Probability Estimate* (CPE) scores. A function $g : \mathcal{X} \to \mathbb{R}_+$ is learnt such that $g(\mathbf{x}) \approx \mathbb{P}[y = 1]$. Various tools such as logistic regression may be used to learn this CPE model g. The final classifier is of the form $\text{sign}(g(\mathbf{x}) - \eta)$ where η is a threshold that is tuned to maximize the performance measure being considered, e.g. classification accuracy, F-measure, G-mean etc.

Plug-in approaches offer various benefits. They can be used to optimize complex multivariate performance measures. In fact the same CPE model can be reused to target several performance measures by simply changing the threshold tuning step. Plug-in approaches have been rigorously analyzed and are known to be statistically consistent (Narasimhan, Vaish, and Agarwal 2014).

However, plug-in methods require total supervision, akin to requiring every instance in every bag to be labeled, whereas MIML operates under a significantly impoverished

Algorithm 1 MIML^{perf}: Training Routine

Input: Data $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, expression rate κ, perf. measure Δ

For all (i,j) such that $y_{i,j} = 1$, set random $\kappa \cdot n_i$ entries of the vector $\mathbf{z}^{(i,j)}$ to 1 *// Initialize hidden labels randomly*
while not converged **do**
Step 1: Fix hidden variables, update plug-in classifiers
  for every label $j \in [L]$ **do**
    $\mathcal{D}^j \leftarrow \{\{(\mathbf{x}_i^{(k)}, z_k^{(i,j)})\}_{k=1}^{n_i}\}_{i=1}^n$ *// Prepare datasets*
    $g^j \leftarrow$ CPE-train$(\mathcal{D}^j)$ *// Train CPE models*
  end for
  $\boldsymbol{\eta} \leftarrow$ Tune-thresholds$((\mathbf{X}, \mathbf{Y}), \mathbf{p}; \Delta)$ *// Optimize Δ*
Step 2: Fix plug-in classifiers, update hidden variables
  for $(i,j) \in [N] \times [L]$ such that $y_{i,j} = 1$ **do**
    $\mathbf{z}^{(i,j)} \leftarrow \mathbf{0}^{n_i}$ *// Reset hidden labels*
    $c^{(i,j)} \leftarrow \sum_{k=1}^{n_i} \mathbb{I}\{g^j(\mathbf{x}_i^{(k)}) \geq \eta_j\}$
    $S^{(i,j)} \leftarrow$ Sample $c^{(i,j)}$ entries of $\mathbf{z}^{(i,j)}$ according to g^j
    $z_k^{(i,j)} \leftarrow 1$ for all $k \in S^{(i,j)}$ *// Reestimate hidden labels*
  end for
end while

Algorithm 2 MIML^{perf}: Testing Routine

Input: Test point $\mathbf{x} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n_t)}\}$ with n_t instances, CPE models $g^1, \ldots, g^L$, thresholds $\boldsymbol{\eta} = [\eta_1, \ldots, \eta_L]$.

for $k = 1, 2, \ldots, n_t$ **do**
  $h_k \leftarrow \{j : g^j(\mathbf{x}^{(k)}) \geq \eta_j\}$ *// Discover hidden labels*
end for
for $j = 1, 2, \ldots, L$ **do**
  $\hat{y}_j \leftarrow \bigvee_{k=1}^{n_t} \mathbb{I}\{j \in h_k\}$ *// Aggregation step*
end for
return $\hat{\mathbf{y}} = [\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_L]$

distant supervision setting. Thus, a direct application of existing plug-in approaches to MIML problems is not possible.

4.2 MIML^{perf}: An MIML Algorithm for Optimizing Multivariate Performance Measures

MIML^{perf} reduces the problem of predicting labels for a bag to the problem of predicting the labels expressed by individual instances in that bag. This is captured by the hidden variables (see Figure 1b). For instance, the sentence *"Mickey Mouse was voiced by Walt Disney himself"* expresses the label `Voice Of` but not the label `Co-creator Of`. Afterward, a label is predicted as relevant for the bag if at least one instance in the bag expresses that label. This *aggregation* or SIML approach has been standard in MIML and RE settings (Haffari, Nagesh, and Ramakrishnan 2015; Hoffmann et al. 2011a; Zhang and Zhou 2008).

Existing approaches suffer since they use expensive techniques such as integer linear programming Haffari et al. (2015) to set these hidden variables. MIML^{perf} instead uses plug-in classifiers to set the hidden variables which results in training routines that are orders of magnitude faster. However, training these classifiers is challenging since we do not have any instance level supervision – the training data does not tell us which instances express which labels.

Overview of MIML^{perf} Training: MIML^{perf} overcomes the challenge of absence of instance level supervision in a scalable manner by combining two powerful approaches – plug-in classifiers and alternating optimization. Alternating approaches such as the EM algorithm are very widely used to train latent variable models. At every time step, MIML^{perf} makes an estimate of the hidden variables, i.e. for an instance in a training bag, which labels are likely to be expressed by that instance. These estimates are valuable since they offer a form of instance level supervision to the method. This makes it possible for MIML^{perf} to train plug-in classifiers by learning CPE models and tuning appropriate thresholds; a separate classifier is trained for each of the L labels. Using these classifiers, MIML^{perf} then improves the estimates of the hidden variables. This process is repeated for a few iterations. The details of the training and testing routines for MIML^{perf} are given in Algorithms 1 and 2.

Hidden Variables These are used to record the algorithm's beliefs about which labels are expressed by individual instances. For a set of N data points $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, the binary variable $z_k^{(i,j)} \in \{0,1\}$ indicates whether the k^{th} instance in the i^{th} bag expresses the j^{th} label or not. If the i^{th} data point has n_i instances then the vector $\mathbf{z}^{(i,j)} \in \{0,1\}^{n_i}$ encodes which of the instances in the bag express the label j. A crucial observation that makes our approach scalable is that if a bag does not have a label j, then none of its instances can express it. In other words, if $y_{i,j} = 0$ then $\mathbf{z}^{(i,j)} = \mathbf{0}$. This observation is valuable since we need only worry about $z_k^{(i,j)}$ when $y_{i,j} = 1$. Since most bags have very few labels (see Figure 1a), this introduces sparsity into our approach. We now describe the alternation steps.

Step 1: Learning Plug-In Classifiers Given an assignment to all the hidden variables $\mathbf{z}^{(i,j)}$, MIML^{perf} learns a plug-in classifier for each label. The job of the classifier corresponding to a particular label is to predict whether a given instance expresses that label or not. This is done by simply formulating a binary classification problem for each label with the hidden variables acting as the "classes". A standard procedure like logistic regression (CPE-train in Algorithm 1) is used to obtain CPE models, one for each label. Afterward, thresholds are tuned for each label such that the performance measure Δ being targeted, such as F-macro or F-micro measure, is maximized. Doing so is challenging since we are working with complex performance measures and in large scale settings, the number of CPE scores being handled runs into millions or more. We refer the reader to Section A in the supplementary material for details.

Step 2: Re-estimating Hidden Variables The CPE models learnt in the previous step can be used to obtain (noisy) CPE scores for all instances with respect to all labels. The noise is due to incorrect assignments to the hidden variables in the previous step. Nevertheless, this gives us an estimate of which instances are more likely to express a certain label. MIML^{perf} uses these CPE scores to reassign the hidden variables. In order to avoid trusting these scores completely, MIML^{perf} chooses a random set of instances with high CPE scores with respect to a label and assigns them that label. This has the effect of reinforcing good CPE models as well as smoothing out errors. For the initialization step, if a bag has a label j, the method assigns a random κ fraction of in-

stances in that bag to label j. κ is an *expression rate* parameter that is only used for initialization. We refer the reader to Section B in the supplementary material for details.

Overview of MIML^{perf} Training: The testing procedure of MIML^{perf} (see Algorithm 2) involves simply applying the plug-in classifiers and performing the aggregation step to obtain the predictions for all labels for a bag. This procedure is enormously cheaper than the integer linear programming approach followed by Haffari et al. (2015).

4.3 Theoretical Analysis

Since multi-instance learning is a non-convex learning problem with complex structure, giving strong theoretical guarantees for algorithms is a challenging albeit interesting problem. Nevertheless we are able to establish generalization guarantees for MIML^{perf} that prove that it does not overfit. Let $\hat{\mathbf{f}}$ be the classifier generated by MIML^{perf}. Let the training set $(\mathbf{X}, \mathbf{Y})$ be chosen randomly from some fixed but unknown distribution and let $(\mathbf{X}^t, \mathbf{Y}^t)$ denote a random test set drawn from the same distribution. Let π denote the minimum frequency of any label.

Theorem 1. *Let the instances be represented as d dimensional features $\mathbf{x}_i^{(k)} \in \mathbb{R}^d$. Then for any N such that $\sqrt{\frac{1}{N}\left(\log\frac{12}{\delta} + d\log\frac{2eN}{d}\right)} < \frac{\beta\pi}{2(1+\beta)}$, we have, with probability at least $1-\delta$, for some constant C*

$$\left|\mathbb{E}\left[F_\beta^{macro}(\hat{\mathbf{f}}; \mathbf{X}^t, \mathbf{Y}^t)\right] - F_\beta^{macro}(\hat{\mathbf{f}}; \mathbf{X}, \mathbf{Y})\right| \leq C\sqrt{\tfrac{1}{N}\left(d + \log\tfrac{1}{\delta}\right)}$$

The result is stated for linear models for sake of simplicity and can be extended to hypothesis spaces with finite capacity as well. A similar result holds true for F-micro measure. We leave the details of this result to the full version of the paper.

5 Experiments

We present a detailed evaluation of our approach and comparisons against the state of the art on three benchmark MIML/RE datasets.

1. **Riedel Distant Supervision Dataset:** For the distantly supervised relation extraction problem, we use the benchmark dataset created by (Riedel, Yao, and McCallum 2010). The dataset was created by aligning relations from *Freebase*[1] with the sentences in the *New York Times* corpus (Sandhaus 2008). The labels for the data points come from the Freebase database; however, Freebase is incomplete (Ritter et al. 2013). A data point is labeled *nil* when either no relation exists or the relation is absent in Freebase. Following Haffari et al. (2015), we train and evaluate the baseline and our algorithms on a subset of this dataset, termed as the *positive dataset*, which consists of only non-nil relation labeled data points.
2. **MIML Scene Classification Dataset (Scene):** The Scene data set contains 2000 scene images collected from the COREL image collection and the Internet, with five different possible class labels, *viz.*, *desert, mountains, sea, sunset* and *trees*. Each image is represented as a bag of nine instances with each instance represented as 15 dimensional feature vector that corresponds to an image patch. Following MIMLSVM, We divided the data into two parts consisting of 1600 data points for training and the remaining 400 points for testing.
3. **MIML Text Classification Dataset (Reuters):** This dataset is derived from the widely studied Reuters-21578 collection using seven most frequent classes. It consists of 2000 documents with 15% of them associated with more than one class label. Each each instance is represented as a 243-dimensional feature vector and corresponds to a text segment. Again, we follow MIMLSVM in the way we partition this dataset into training and testing splits.

[1] www.freebase.com

Table 1: Dataset Statistics

	#bags	#labels	$\frac{\text{labels}}{\text{point}}$	$\frac{\text{points}}{\text{label}}$	$\frac{\text{instances}}{\text{bag}}$
Riedel	4350	52	1.08	90.4	6.6
Scene	2000	5	1.24	494.4	9
Reuters	2000	7	1.15	329.71	3.56

Comparisons against existing approaches We first present a comparison of MIML^{perf} against state-of-the-art methods for each of the three datasets. We report F_β^{macro}, F_β^{micro} and average precision as our evaluation measures. In addition, we also report the training time and the average time required to predict labels for each test bag (instance set). We used a default value $\kappa = 1$ as the *expression rate* for training MIML^{perf}. For the sake of fairness, training splits and feature sets were kept common across all experiments.

Table 2 compares MIML^{perf} against state-of-the-art approaches MIMLSVM and M3MIML (Zhang and Zhou 2008) for the Scene and Reuters datasets. We observe that MIML^{perf} consistently outperforms both methods on all three evaluation measures. Additionally, as noted in Table 5, MIML^{perf} is significantly and consistently faster in training than MIMLSVM and even more so at testing time.

Table 3 reports a comparison for the specific distant supervision RE setting against the MM-F$_\beta$ approach Haffari et al. (2015). MIML^{perf} consistently outperforms MM-F$_\beta$ on F-macro as well as F-micro with a 143-fold speedup in training. While the gains on F-micro are marginal, we note that those on F-macro are significant which is important, especially when the class distribution is heavy tailed with lots of rare classes. In such situations, F-macro performance becomes more crucial than F-micro performance. We note that MM-F$_\beta$ is the leading algorithm for RE and itself beats other RE algorithms. For this reason we compare only to MM-F$_\beta$.

In addition to offering gains on training and test speeds, our approach does not require the entire dataset to be loaded in memory. We executed MIML^{perf} efficiently by noticing that several components of the algorithm, most importantly the training of the CPE models, are inherently parallelizable.

Effects of changes in parameters and objective In Table 4 we compare the difference in performances of MIML^{perf} when the macro-F measure is replaced with the micro-F measure as the objective. We compare the results

Table 2: Performance of MIML^{perf} compared against MIMLSVM and M3MIML on the Scene and Reuters Datasets

Dataset	Kernel	F-Macro			F-Micro			Avg. Precision		
		MIMLSVM	M3MIML	MIML^{perf}	MIMLSVM	M3MIML	MIML^{perf}	MIMLSVM	M3MIML	MIML^{perf}
Scene	lin	0.4292	0.386	**0.5427**	0.4475	0.3868	**0.528755**	0.6635	0.5754	**0.746562**
	rbf	0.6054	0.5872	**0.6201**	0.6016	0.5796	**0.615842**	0.7704	0.74857	**0.791984**
Reuters	lin	0.8274	0.7601	**0.8492**	0.8395	0.8067	**0.862955**	0.9489	0.9463	**0.9625**
	rbf	0.88387	0.69979	**0.8901**	0.8689	0.7866	**0.89422**	0.9467	0.9403	**0.970646**

Table 3: MIML^{perf} comparison with the (MM-F$_\beta$) model by Haffari et. al. on the Riedel dataset

	F-macro	Precision	Recall	F-micro	Training time
MM-F$_\beta$	0.1366	0.6599	0.6521	0.6559	4h 20m
MIML^{perf}	**0.228354**	0.79281	0.57811	**0.66865**	1m 49s

Table 4: Effect of the training objective on test performance on the Riedel dataset

	F-macro	F-micro
Optimizing F_β^{macro}	**22.8354**	61.0615
Optimizing F_β^{micro}	18.984	**66.8648**

Table 5: Comparisons of training time and testing time per instance

Dataset	Kernel	Training time			Testing time per instance		
		MIMLSVM	MIML^{perf}	Speedup	MIMLSVM	MIML^{perf}	speedup
Scene	Linear	7m 45s	1.66s	280×	0.0920s	0.0009s	102×
	RBF	9m 24s	6m 40s	1.41×	0.0921s	0.03s	3×
Reuters	Linear	3m 55s	1.62s	145×	0.1035s	0.0005s	207×
	RBF	5m 04s	1m 02s	5×	0.1063s	0.008s	13×

Table 6: Relation-wise comparisons of F-score with MM-F$_\beta$, MIML^{perf}, $\kappa = 1$ and MIML^{perf}, tuned κ on Riedel

Relation	% of occurrence	MM-F$_\beta$	MIML^{perf} $\kappa = 1$	MIML^{perf} tuned κ
/people/person/place_lived	12.36	0.15966	**0.46556**	**0.46897**
/people/deceased_person/place_of_death	4.04	0.19231	0.31111	**0.35294**
/location/administrative_division/country	1.26	0	**0.37624**	**0.37624**
/business/company/founders	1.09	0.48148	0.68571	**0.75676**
/location/country/capital	0.77	0	0.10866	**0.11494**
/film/film/featured_film_locations	0.15	0	0.01262	**0.01325**
Overall F-Macro		0.13660	0.22835	**0.23282**

of training to optimize the F_β^{micro} measure against optimizing F_β^{macro}. In each setting, we report both F_β^{micro} and F_β^{macro}. We note the strong correspondence between the choice of performance in the training objective and performance being evaluated upon the test dataset.

Performance on Rare Labels We expect optimizing the F-macro performance measure to enhance performance on the rarer labels (that have low priors). We confirm this in Table 6, wherein we compare the performance of MIML^{perf} with MM-F$_\beta$ for some of the rarer labels. Whereas MM-F$_\beta$ reports extremely low F-scores (in some cases 0) on rare labels, MIML^{perf} performs much better. The value of κ yielding the best performance for a label changes across classes and therefore, also report results with κ tuned for each label on the training dataset. This gives us even better gains.

We also test for statistical differences between the accuracies of MIML^{perf} and MM-F$_\beta$ on all the classes using the Wilcoxon[2] signed-rank test. We notice that the sum of the signed ranks (222 with $\kappa = 1$ and 237 for κ tuned for each class) is very clearly in favour of MIML^{perf} over MM-F$_\beta$.

[2]This is a non-parametric test of the null hypothesis that there is no significant difference between the median performance of a pair of algorithms.

6 Conclusions

We presented MIML^{perf}, a scalable algorithm for MIML learning capable of optimizing several multivariate performance measures in the context of multi-instance multi-label learning (MIML). Our approach operates with significantly reduced running times and caters well even to the rarer classes, without compromising on the larger ones. Further, our *plug-in* approach appears very amenable to distributed and parallel computing and a possible future work is validating this hypothesis.

References

Bi, W., and Kwok, J. T. 2013. Efficient Multi-label Classi cation with Many Labels. In *Proceedings of ICML*.

Chen, Y.-N., and Lin, H.-T. 2012. Feature-aware Label Space Dimension Reduction for Multi-label Classification. In *Proceedings of NIPS*.

Dembczyński, K.; Jachnik, A.; Kotłowski, W.; Waegeman, W.; and Hüllermeier, E. 2013. Optimizing the F-measure in multi-

label classification: Plug-in rule approach versus structured loss minimization. In *Proceedings of ICML*.

Felzenszwalb, P. F.; Girshick, R. B.; McAllester, D. A.; and Ramanan, D. 2010. Object detection with discriminatively trained part-based models. *IEEE TPAMI* 32(9):1627–1645.

GuoDong, Z.; Jian, S.; Jie, Z.; and Min, Z. 2005. Exploring various knowledge in relation extraction. In *Proceedings of ACL*, ACL '05, 427–434. Stroudsburg, PA, USA: Association for Computational Linguistics.

Haffari, G.; Nagesh, A.; and Ramakrishnan, G. 2015. Optimizing Multivariate Performance Measures for Learning Relation Extraction Models. In *Proceedings of Human Language Technologies: the 2015 Annual Conference of the North American Chapter of the ACL*.

Hoffmann, R.; Zhang, C.; Ling, X.; Zettlemoyer, L.; and Weld, D. S. 2011a. Knowledge-based Weak Supervision for Information Extraction of Overlapping Relations. In *Proceedings of the ACL*.

Hoffmann, R.; Zhang, C.; Ling, X.; Zettlemoyer, L.; and Weld, D. S. 2011b. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of ACL*, HLT '11, 541–550. Stroudsburg, PA, USA: Association for Computational Linguistics.

Hsu, D.; Kakade, S.; Langford, J.; and Zhang, T. 2009. Multilabel Prediction via Compressed Sensing. In *Proceedings of NIPS*.

Joachims, T. 2005. A support vector method for multivariate performance measures. In *International Conference on Machine Learning (ICML)*, 377–384.

Keshet, J. 2014. Optimizing the measure of performance in structured prediction. In Nowozin, S.; Gehler, P. V.; Jancsary, J.; and Lampert, C. H., eds., *Advanced Structured Prediction*. The MIT Press.

Komodakis, N.; Paragios, N.; and Tziritas, G. 2011. Mrf energy minimization and beyond via dual decomposition. *Proceedings of IEEE TPAMI* 33(3):531–552.

Kotłowski, W., and Dembczynski, K. 2015. Surrogate Regret Bounds for Generalized Classification Performance Metrics. In *Proceedings of ACML*.

Koyejo, O.; Natarajan, N.; Ravikumar, P.; and Dhillon, I. 2014. Consistent Binary Classification with Generalized Performance Metrics. In *Proceedings of NIPS*.

Mintz, M.; Bills, S.; Snow, R.; and Jurafsky, D. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of ACL*, ACL '09, 1003–1011. Stroudsburg, PA, USA: Association for Computational Linguistics.

Narasimhan, H.; Kar, P.; and Jain, P. 2015. Optimizing Non-decomposable Performance Measures: A Tale of Two Classes. In *Proceedings of ICML*.

Narasimhan, H.; Vaish, R.; and Agarwal, S. 2014. On the Statistical Consistency of Plug-in Classifiers for Non-decomposable Performance Measures. In *Proceedings of NIPS*.

Ranjbar, M.; Lan, T.; Wang, Y.; Robinovitch, S. N.; Li, Z.-N.; and Mori, G. 2013. Optimizing nondecomposable loss functions in structured prediction. *Proceedings of IEEE TPAMI* 35(4):911–924.

Ranjbar, M.; Vahdat, A.; and Mori, G. 2012. Complex loss optimization via dual decomposition. In *Proceedings of CVPR*, 2304–2311.

Riedel, S.; Yao, L.; and McCallum, A. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of ECML-PKDD*, ECML PKDD'10, 148–163. Berlin, Heidelberg: Springer-Verlag.

Ritter, A.; Zettlemoyer, L.; Mausam; and Etzioni, O. 2013. Modeling missing data in distant supervision for information extraction. *TACL* 1:367–378.

Rosenfeld, N.; Meshi, O.; Globerson, A.; and Tarlow, D. 2014. Learning structured models with the auc loss and its generalizations. In *Proceedings of AISTATS*.

Rush, A. M., and Collins, M. 2012. A tutorial on dual decomposition and lagrangian relaxation for inference in natural language processing. *Journal of Artificial Intelligence Research* 45:305–362.

Sandhaus, E. 2008. The New York Times annotated corpus.

Steck, H. 2010. Training and Testing of Recommender Systems on Data Missing Not at Random. In *Proceedings of KDD*.

Surdeanu, M., and Ciaramita, M. 2007. Robust information extraction with perceptrons. In *Proceedings of ACE07 Workshop*.

Surdeanu, M.; Tibshirani, J.; Nallapati, R.; and Manning, C. D. 2012a. Multi-instance Multi-label Learning for Relation Extraction. In *Proceedings of EMNLP-CoNLL*.

Surdeanu, M.; Tibshirani, J.; Nallapati, R.; and Manning, C. D. 2012b. Multi-instance multi-label learning for relation extraction. In *Proceedings of EMNLP-CoNLL*, EMNLP-CoNLL '12, 455–465. Stroudsburg, PA, USA: Association for Computational Linguistics.

Tarlow, D., and Zemel, R. S. 2012. Structured output learning with high order loss functions. In *Proceedings of Conference on Artificial Intelligence and Statistics*.

Taskar, B.; Guestrin, C.; and Koller, D. 2003. Max-margin markov networks. In *Proceedings of NIPS*.

Wang, Y., and Mori, G. 2011. Hidden part models for human action recognition: Probabilistic versus max margin. *IEEE TPAMI* 33(7):1310–1323.

Weston, J.; Bengio, S.; and Usunier, N. 2011. WSABIE: Scaling Up To Large Vocabulary Image Annotation. In *Proceedings of IJCAI*.

Yao, L.; Riedel, S.; and McCallum, A. 2010. Collective cross-document relation extraction without labelled data. In *Proceedings of EMNLP*, EMNLP '10, 1013–1023. Stroudsburg, PA, USA: Association for Computational Linguistics.

Ye, N.; Chai, K. M. A.; Lee, W. S.; and Chieu, H. L. 2012. Optimizing F-Measures: A Tale of Two Approaches. In *Proceedings of ICML*.

Yu, C. J., and Joachims, T. 2009. Learning structural svms with latent variables. In *Proceedings of ECML*, 147.

Yuille, A. L., and Rangarajan, A. 2001. The concave-convex procedure (cccp). In *Proceedings of NIPS*, 1033–1040.

Zeng, D.; Liu, K.; Chen, Y.; and Zha, J. 2015. Distant Supervision for Relation Extraction via Piecewise Convolutional Neural Networks. In *Proceedings of EMNLP*.

Zhang, M.-L., and Zhou, Z.-H. 2008. M^3MIML: A maximum margin method for multi-instance multi-label learning. In *Proceedings of ICDM*.

Zhou, Z.-H., and Zhang, M.-L. 2006. Multi-Instance Multi-Label Learning with Application to Scene Classification. In *Proceedings of NIPS*.

A Tuning the Thresholds

As we mentioned in Section 4.1, plug-in classifiers learn a CPE model to predict $g(\mathbf{x}) \approx \mathbb{P}[y = +1]$ and then tune a threshold η to obtain a classifier $f(\mathbf{x})\text{sign}(g(\mathbf{x}) - \eta)$. The threshold η is tuned in order to maximize the performance measure of interest, be it classification accuracy or F-measure or G-mean etc. A popular technique to perform this tuning is the Empirical Utility Maximization (EUM) approach (Ye et al. 2012). The EUM approach suggests that a threshold be chosen that maximizes the performance measure on the training dataset.

For simplicity, consider a binary classification problem where we have n labeled data points $(\mathbf{x}_i, y_i)$ where $y_i \in \pm 1$. The CPE model g can be used to obtain CPE scores $s_i = g(\mathbf{x}_i)$. The EUM approach seeks to find a threshold η_{opt} such that the classifier $\text{sign}(g(\mathbf{x}) - \eta_{\text{opt}})$ has, say very high F-measure, on the dataset. However, this search problem is daunting since thresholds are real valued in general. However, other results such as those of (Kotłowski and Dembczynski 2015; Narasimhan, Vaish, and Agarwal 2014) assure us that the optimal threshold will definitely be one of the CPE scores itself i.e. $\eta_{\text{opt}} = s_i$ for some i.

MIML^{perf} adopts the same EUM approach in the MIML setting as well. However, there are two key differences here

1. A separate threshold has to be tuned for the plug-in classifier corresponding to all the L labels.
2. The labels for a bag cannot be obtained from the instance CPE scores directly since there is an aggregation step involved.

More specifically, our classifier turns on the label j for a bag i only if that label is turned on for some instance k in that bag. Recall from Algorithm 2 that we set

$$\hat{y}_j = \bigvee_{k=1}^{n_t} \mathbb{I}\left\{ g^j(\mathbf{x}^{(k)}) \geq \eta_j \right\}$$

This presents a challenge since the number of CPE scores to be tuned over can run into millions or more for large datasets due to the large number of bags and large number of instances in each bag. Checking each CPE score as a candidate threshold is thus, not feasible.

We overcome this problem using a simple trick. Notice that the plug-in classifier has a monotonicity property. If $\text{sign}(g(\mathbf{x}) - \eta) = -1$ then $\text{sign}(g(\mathbf{x}') - \eta) = -1$ for all $g(\mathbf{x}') \leq g(\mathbf{x})$. Also, if $\text{sign}(g(\mathbf{x}) - \eta) = +1$ then $\text{sign}(g(\mathbf{x}') - \eta) = +1$ for all $g(\mathbf{x}') \geq g(\mathbf{x})$. Using this, it is easy to see that a label j will be turned on for a bag i if and only if the instance with the largest CPE score for that label in the bag has been assigned that label. More specifically, let $s_{\max}^{(i,j)} := \max_{k \in [n_i]} g^j(\mathbf{x}_i^{(k)})$. Then the following claim holds true

Lemma 2. *For any data point i, label j, CPE model, g^j and threshold η_j, we have $\hat{y}_j = 1$ iff $s_{\max}^{(i,j)} \geq \eta_j$.*

This shows us that while tuning thresholds for the plug-in classifier corresponding to a certain label, it is sufficient to only consider the maximum CPE score for that label in every bag as a candidate threshold. This gives a huge speedup since it decreases the number of thresholds being tuned over from $\sum_{i=1}^{N} n_i$ to simply N. In practice we observe an order of magnitude from this step alone.

However, this is still not sufficient since testing a candidate threshold itself takes $\mathcal{O}(N)$ time since that is the amount of time taken to calculate the performance measure, say F-measure, corresponding to that threshold. This brings the total time taken to tune the L thresholds as $\mathcal{O}\left(N^2 \cdot L\right)$ which is simply infeasible in large scale settings.

Fortunately, the nice structure of the performance measures such as F-micro and F-macro measure come in handy at this point. It turns out that all of these performance measures can be written as some function of the true positive (TP) and true negative (TN) numbers of the classifier being considered. For example, if we look at the label-wise F-measure which is used to define the macro F-measure, we find that if we define

$$\text{TP}^j(\mathbf{f}, \mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{n} \mathbb{I}\left\{ \mathbf{f}^j(\mathbf{x}_i) > 0 \wedge y_{i,j} > 0 \right\}$$

$$\text{FP}^j(\mathbf{f}, \mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{n} \mathbb{I}\left\{ \mathbf{f}^j(\mathbf{x}_i) > 0 \wedge y_{i,j} \leq 0 \right\}$$

$$\text{FN}^j(\mathbf{f}, \mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{n} \mathbb{I}\left\{ \mathbf{f}^j(\mathbf{x}_i) \leq 0 \wedge y_{i,j} > 0 \right\},$$

then we can write $F_\beta^j(\mathbf{f}; \mathbf{X}, \mathbf{Y})$ as

$$\frac{\text{TP}^j(\mathbf{f}, \mathbf{X}, \mathbf{Y})}{\text{TP}^j(\mathbf{f}, \mathbf{X}, \mathbf{Y}) + \beta \cdot \text{FP}^j(\mathbf{f}, \mathbf{X}, \mathbf{Y}) + (1 - \beta) \cdot \text{FN}^j(\mathbf{f}, \mathbf{X}, \mathbf{Y})}.$$

Similar expressions can be obtained for a large family of performance measures (Koyejo et al. 2014; Narasimhan, Vaish, and Agarwal 2014; Narasimhan, Kar, and Jain 2015). These expressions are very useful since the terms TP^j, FP^j etc can be calculated for *all* candidate thresholds in time $\mathcal{O}(N \log N)$. We simply need to sort all the candidate CPE scores in increasing order and simply do a linear scan to find out what value of TP^j, FP^j etc does every candidate threshold offer. This brings down the total time taken to tune the L thresholds from $\mathcal{O}\left(N^2 \cdot L\right)$ to $\mathcal{O}(N \log N \cdot L)$.

We conclude this section by noting that while optimizing macro F-measure, we tune a different threshold per label, whereas we tune a single threshold while optimizing micro F-measure.

B Estimating the Hidden Variables

As mentioned in Section 4, MIML^{perf} uses the trained CPE models to obtain CPE scores for every instance with respect to every label. Thus, the method now has an estimate $g^j(\mathbf{x}_i^{(k)})$ of the probability that the k^{th} instance in bag i expresses label j. These are now used to reassign the hidden variables. As discussed, we do not wish to trust these scores completely as they are noisy. Thus, we do not wish to set $z_k^{(i,j)} = 1$ only for the instances k with the highest values of $g^j(\mathbf{x}_i^{(k)})$. Instead we *sample* instances the instances according to their CPE scores to reassign them labels.

More specifically, if a bag i has a label j (recall that if the bag does not have label j then we can simply set $\mathbf{z}^{(i,j)} = \mathbf{0}$ since no instance in that bag could be expressing label j) then we sample the k^{th} instance with probability $g^j(\mathbf{x}_i^{(k)}) / \sum_{k'=1}^{n_i} g^j(\mathbf{x}_i^{(k')})$ where n_i is the number of instances in bag i. Using this procedure we create a set $S^{(i,j)}$ of $c^{(i,j)}$ instances. We now simply set $z_k^{(i,j)} = 1$ if $k \in S$ and $z_k^{(i,j)} = 0$ otherwise.

The choice of $c^{(i,j)}$ can be varied. In our experiments, we used $c^{(i,j)} = \kappa \cdot n_i$ in the initialization step where κ is the expression rate parameter. In subsequent steps, we used $c^{(i,j)} = \sum_{k=1}^{n_i} \mathbb{I}\{g^j(\mathbf{x}_i^{(k)}) \geq \eta_j\}$, i.e. we ask the plug-in classifier trained in the previous iteration, how many instances in that bag were found to express a certain label.