From Group to Individual Labels using Deep Features

Dimitrios Kotzias[1] Misha Denil[2] Nando De Freitas[2,3] Padhraic Smyth[1]

[1]University of California, Irvine
[2]University of Oxford
[3]Canadian Institute for Advanced Research

[1]{dkotzias,smyth}@ics.uci.edu
[2]{mdenil,nando}@cs.ox.ac.uk

ABSTRACT

In many classification problems labels are relatively scarce. One context in which this occurs is where we have labels for groups of instances but not for the instances themselves, as in multi-instance learning. Past work on this problem has typically focused on learning classifiers to make predictions at the group level. In this paper we focus on the problem of learning classifiers to make predictions at the instance level. To achieve this we propose a new objective function that encourages smoothness of inferred instance-level labels based on instance-level similarity, while at the same time respecting group-level label constraints. We apply this approach to the problem of predicting labels for sentences given labels for reviews, using a convolutional neural network to infer sentence similarity. The approach is evaluated using three large review data sets from IMDB, Yelp, and Amazon, and we demonstrate the proposed approach is both accurate and scalable compared to various alternatives.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; I.2.7 [**Artificial Intelligence**]: Text analysis

Keywords

Multi-instance learning, unsupervised learning, deep learning, sentiment analysis

1. INTRODUCTION

There are a variety of classification problems where class labels are not available at the instance level but are available for groups of instances. For example, text documents such as product and movie reviews may have positive or negative labels at the document (group) level rather than at the sentence (instance) level. Similarly, images may be labeled in terms of whether an object is present or not within the image, but without any labels for localized regions within the image.

This "group-level" labeling is often expedient when manually labeling data since it is relatively easy for a human to quickly provide labels at the group level, e.g., indicating whether a review is positive or negative or whether an image contains a cat or not. In contrast generating labels at the instance-level (e.g. for sentences within reviews) is much more time consuming.

There are many applications, however, where we would like to be able to go beyond the group level and make predictions about class labels at the instance level. For example, in generating summaries of positive and negative reviews for human interpretation it would be useful identify not only which reviews (groups) are most positive and negative, but also which sentences within reviews are positive or negative. In turn this information could be used for summarization purposes indicating the most positive or negative sentences.

As an example, in cases where reviews are generally positive, detecting negative comments is a key step toward improving customer service. In contexts involving individual information and privacy, such as healthcare and census data, quantifying how much information can be predicted at the individual level given group information is an important issue (e.g., see [13, 33]).

In this paper we propose a new approach to the problem of using group-level labels to learn instance-level classification models. The resulting classifiers can be used to transfer information from the group-level to the instance level when group labels are available, in addition to making predictions about new instances and groups.

Our approach is based on an objective function that takes advantage of instance similarity in order to impose smoothness on instance labels, while at the same time respecting group-level label constraints. We demonstrate how this idea can be used to infer ratings of sentences (instances) from ratings of reviews (groups of sentences). A key step in this approach is the use of embedding techniques, to obtain vector-based representations for sentences. Here we use convolutional neural networks for the embeddings. Using these vector representations, we formulate a regularized manifold learning objective function to learn the labels of each sentence. We transfer the labels from entire reviews to individual sentences and in doing so, eliminate the high cost of gathering labels at the sentence level. Although the focus of this paper is on text documents, the ideas are applicable to a considerably broader range of problems.

KDD'15, August 10-13, 2015, Sydney, NSW, Australia.

ACM 978-1-4503-3664-2/15/08 ...$15.00.
DOI: http://dx.doi.org/10.1145/2783258.2783380 .

Prior work related to this problem has focused on more constrained scenarios. For example, in multi-instance learning (MIL), where groups of instances are referred to as "bags," a common assumption is that all negatively-labeled bags contain only negative instances and all positively-labeled bags contain at least 1 positive instance [8]. This type of assumption can be somewhat restrictive when dealing with real-world data sets such as reviews, which in general can contain both positive and negative sentences. In addition, much of the earlier work in MIL is focused on learning to predict labels at the group (bag) level rather than at the instance level. More recent approaches e.g., [34] have been proposed that relax the OR assumption and that focus on predicting instance-level labels. However, these methods often do not scale well computationally in the number of instances [30]. Relative to this earlier work, the primary contributions of this paper are:

- A novel and flexible objective function that uses instance similarity and group label constraints to learn instance-level classification models, with a scalable learning algorithm using stochastic gradient methods;
- The application and evaluation of this approach on the problem of sentence-level sentiment prediction for multiple large real-world review data sets, using neural network embeddings to obtain vector-valued representations for sentences.

In Section 2 below we review related work. Section 3 introduces our new objective function, with a description of how the instance-level similarity and group-level label constraints are combined, as well as an outline of how the objective can be minimized using stochastic gradient optimization. In Section 4 we describe the use of convolutional neural networks to learn vector representations for sentences. Section 5 presents our experimental results, on three large review data sets: movie reviews from IMDB, restaurant reviews from Yelp, and product reviews from Amazon. We show that our proposed method performs well across these data sets. We also compare our method against more classical approaches in MIL. In Section 6 we discuss scalability aspects of our approach including the relatively fast convergence of stochastic gradient approach. Section 7 contains conclusions.

2. RELATED WORK

MIL focuses on classification problems where labels are associated with *sets of instances*, often referred to as *bags* or *groups*, instead of individual instances[1]. The labels associated with the groups are assumed to be some function of the unobserved instance-level labels. MIL was first explored as a variant of a standard supervised learning problem in which training examples are ambiguous in the sense that each object has multiple feature vectors but only one of those may be responsible for its label [8]. The MIL framework has since been applied to a large variety of applications, such as content-based image retrieval and classification [21], text categorization [1], object recognition [14] and privacy [13].

The traditional definition of MIL, however, makes a strong assumption that the aggregation function over instance labels is an OR function, i.e., that positive bags contain at least 1 positive instance and that negative bags contain only negative instances. Moreover, the primary focus is to infer labels at the group level rather than at the instance level.

A number of approaches relax the assumption of an OR function and propose other forms of aggregation. Weidmann et al. [31] consider a generalization where the presence of a combination of instance types determines the label of the group. Xu and Frank [32] assume that all instances contribute equally and independently to a group's class label. Zhou et al. [35] build a model that solves MIL through semi-supervised learning techniques by considering a negative label for every instance in a negative group. These types of solutions are typically tailored to handle specific assumptions about the whole-part relationship between groups and instances. Our proposed approach is more flexible as it allows for more general parametrizations of such relationships.

A more recent line of work addresses the problem of predicting labels for instances, but still assuming the OR function for label aggregation. For example Kandemir et al., [12] model the class distribution through a non-parametric mixture model and infer class labels that satisfy the constraints of the bags. Liu et al. [19] search for positive instances (called key instances) within positively-labeled groups using nearest-neighbor graphs among instances and voting schemes.

Support-vector machines (SVMs) have also been widely used for MIL problems. For example, Gartner et al [10] proposed kernels that work for groups of instances and then trained a traditional SVM on these. Andrews et al [1] propose an extension called mi-SVM, where they maximize a soft-margin criterion jointly over possible label assignments as well as hyperplanes. Other approaches include transductive SVMs [4] and approaches for sparse multi-instance learning [3], as well as methods for identifying key instances and connect them through labels, which is very useful for discovering regions of interest in images [17, 18]. A drawback of the SVM approach in general is the computational complexity, often necessitating approximations and limiting applicability in practice to relatively small data sets.

A different strand of work in MIL assumes that the expected proportion of instances per class is known for each group. Given this information, the goal is to predict the label of each instance within the bags [15, 27, 33]. Patrini et. al [25] developed theoretical results that show that a mean operator on the group labels can provide a minimally sufficient statistic for many proper cost functions defined on labels of instances. They demonstrate how this approach can be used to learn accurate instance-level classifiers on a variety of well-known data sets.

The method we propose in this paper can be viewed as complementary to the prior work described above. In particular we focus on instance level predictions, allow for general forms of aggregation functions, and do not require knowledge of label proportions. Our contribution is based on a new cost function that can work with general aggregation functions and a variety of instance classifiers and that can provide predictions for both group and instance level labels. Our results are, to the best of our knowledge, the first large-scale application of MIL techniques to text-based review data, and in particular to the problem of automatically identifying positive and negative sentences within reviews using an MIL approach.

[1]The literature on this topic is vast and there is often disagreement in terminology—for extensive surveys see [5, 9]

3. MULTI INSTANCE LEARNING LOSS FUNCTION

3.1 Problem Formulation

Consider a set of training instances, $\mathcal{X} = \{\mathbf{x}_i\}, i = 1 \ldots N$, where unlike the standard supervised setting we are not given labels for each training instance directly. Instead we are given labels for groups of instances:

$$\mathcal{D} = \{(\mathcal{G}_k, \ell_k)\}_{k=1,\ldots,K}$$

where $\mathcal{G}_k \subseteq \mathcal{X}$ is a multi-set of instances from $\mathcal{X}$ and ℓ_k is a label assigned to the group $\mathcal{G}_k$. We assume ℓ_k is an unknown function of the (unobserved) labels of the elements of $\mathcal{G}_k$. In this paper we focus on binary labels, $\ell_k \in \{0, 1\}$, but the approach is more broadly applicable to non-binary cases. We also assume we have a function $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) \in [0, 1]$ which measures the similarity between pairs of instances $\mathbf{x}_i$ and $\mathbf{x}_j$. This can be viewed as a kernel function—in Section 4 we discuss in detail how we use neural embeddings to construct $\mathcal{K}$ for sentences. In addition $\hat{y}_i = \hat{y}_\theta(\mathbf{x}_i)$ denotes a real-valued score, representing the likelihood that an instance $\mathbf{x}_i$ belongs to a class label, as predicted by classifier y with parameters $\boldsymbol{\theta}$.

Our goals here are twofold. Firstly, we would like to infer labels for each example by propagating information from the group labeling to the instances, essentially inverting the unknown label aggregation function on the training data. In order to achieve this we take advantage of the similarity measure $\mathcal{K}$ to compute a label assignment that is compatible with the group structure of the data, and simultaneously learn to assign the same label to similar instances.

Our second goal is more ambitious. In addition to assigning labels to the training instances we also aim to produce a classifier that predicts a label for instances not found in the training set. These labels can then be aggregated and used to infer the labels of new groups.

3.2 The General Cost Function

We can achieve both of these goals by constructing a general cost function which we minimize as a function of the classifier parameters $\boldsymbol{\theta}$:

$$J(\boldsymbol{\theta}) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) \Delta_1\Big(\hat{y}_i, \hat{y}_j\Big) + \frac{\lambda}{K} \sum_{k=1}^{K} \Delta_2\Big(\hat{\ell}_k, \ell_k\Big)$$

where:

- $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) \in [0, 1]$ represents a similarity measure between instances $\mathbf{x}_i, \mathbf{x}_j$;
- $\Delta_1\big(\hat{y}_\theta(\mathbf{x}_i), \hat{y}_\theta(\mathbf{x}_j)\big)$ is a non-negative penalty on the difference between predictions for instances i and j;
- $\Delta_2\big(\hat{\ell}_k, \ell_k\big)$ is a non-negative penalty on the difference between the prediction and the true label for group k.
- $\hat{\ell}_k = \mathcal{A}\left(\mathcal{G}_k, \boldsymbol{\theta}\right) \in [0, 1]$ is a real-valued scalar representing the output of an aggregation function for all instance-level label predictions in a group $\mathcal{G}_k$. The exact specification of this whole-part relationship will typically depend on the particular application.
- $\lambda > 0$ balances the contributions between the 2 sums, and can be selected via cross-validation on a validation set.

For example Δ_1 and Δ_2 above could be specified as square error or log-loss. N and K are the total number of instances and groups, respectively.

Both terms in the objective function above can be seen as different forms of label propagation. The first term is a standard manifold-propagation term, which spreads label information over the data manifold in feature-space. A similar term often appears in semi-supervised learning problems, where the goal is to make predictions using a partially-labeled data set. In such a setting a label propagation term alone is sufficient; however, since we have labels only for groups of instances we require additional structure in our cost function.

The second term parametrizes the whole-part relationship between the groups and the instances they contain. This has the effect of propagating information from the group labels to the instances. In addition, this term acts as a regularizer and helps avoid the trivial cases where every instance has the same label, regardless of the group it belongs to. In semi-supervised learning, we do not need such a regularizer, as correct labels of instances will avoid the trivial solution—here, however, without fine grained supervision, the regularizer term is required to deal with this issue.

Neither of the two individual terms in the cost function would work well by itself. This situation is not unlike what we find when we carry out kernel regression with L_1 regularization, where the likelihood term often leads to pathological problems and the regularizer simply has the effect of shrinking the parameters to a common value (typically zero). However, when we combine the two competing terms, we are able to obtain useful results. The parameter λ trades off the contributions of the two terms.

Optimizing this objective will produce a classifier $\hat{y}_{\boldsymbol{\theta}}(\mathbf{x})$ which can assign labels to previously seen or to unseen instances, despite having been trained using only group labels. This classifier simultaneously achieves both of our stated goals: we can apply the classifier to instances of $\mathcal{X}$ in order to obtain labels for the training instances, and we can also use it to make predictions for unseen test instances. We can also predict labels at the group level, by aggregating all our predictions at the instance level within a group.

This formulation relies on having a good similarity measure $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$. It would be simple to take the average score of each instance across groups, and minimize the second term of the objective. However, the presence of the first term pushes similar items across different groups to have similar labels and allows for inter-group knowledge transfer.

3.3 A Specific Cost Function

In this paper, where our ultimate goal is to apply this method to reviews and sentences within reviews, we chose relatively simple measures for each of the components in the cost function and that make sense for this application.

For the classifier we used a simple logistic regression model where:

$$\hat{y}_i = \hat{y}_{\boldsymbol{\theta}}(\mathbf{x}_i) = \sigma(\boldsymbol{\theta}^\top \mathbf{x}_i) = \frac{1}{1 + e^{-\boldsymbol{\theta}^\top \mathbf{x}_i}},$$

and we chose the two penalty functions, Δ_1, Δ_2, each to be the square loss, i.e.,

$$\Delta_1(\hat{y_i}, \hat{y_j}) = (\hat{y_i} - \hat{y_j})^2 \tag{1}$$

Other weighted loss functions could be used for the first term. The role of this term is to ensure that similar individ-

ual features $\mathbf{x}_i$ are assigned similar labels y, smoothed by the similarity function $\mathcal{K}$. For this similarity function, we used the kernel:

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2), \in [0, 1] \quad (2)$$

which is a transformation of the Euclidean norm and a special case of the radial basis function (RBF) kernel. Since the choice of a particular kernel can in general make a difference in machine learning problems, we also evaluated differently parametrized RBF kernels as well as cosine similarity for the vectors in our experiments. We found that the accuracy of the overall method was not sensitive to the particular choice and parametrization of the kernel.

To parametrize the whole-part relationship we chose a simple function that assumens that the label of a group is obtained by averaging the labels of its elements. Hence, for each group g :

$$\hat{\ell_k} = \mathcal{A}(\mathcal{G}_k, \boldsymbol{\theta}) = \frac{1}{|\mathcal{G}_k|} \sum_{i \in \mathcal{G}_k} \hat{y}_{\boldsymbol{\theta}}(\mathbf{x}_i)$$

This averaging function makes more sense for modeling reviews and sentences, compared to (say) the OR function, since we expect that the label of a review will in general tend to correlate with the average sentence-level sentiment across its constituent sentences.

Thus, in this paper, the specific cost function we optimize is:

$$J(\boldsymbol{\theta}) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} e^{(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2)} \left(\sigma(\boldsymbol{\theta}^\top \mathbf{x}_i) - \sigma(\boldsymbol{\theta}^\top \mathbf{x}_j)\right)^2 + \frac{\lambda}{K} \sum_{k=1}^{K} \left(\frac{1}{|\mathcal{G}_k|} \left(\sum_{i \in \mathcal{G}_k} \sigma(\boldsymbol{\theta}^\top \mathbf{x}_i) \right) - \ell_k \right)^2 \quad (3)$$

More complex classifiers, alternative whole-part relationships, and other penalty functions could be explored, but are beyond the scope of this paper. Instead, we have opted for simplicity in our choices above and will show later in our experimental results that these choices lead to accurate models.

3.4 Training Methodology

Given a specification of a cost function, one can train our classifier as follows:

1. Create vector representations $\mathbf{x}_i$ of the instances (discussed in the next section);
2. Optimize the cost function through stochastic gradient descent, to learn the parameters $\boldsymbol{\theta}$;
3. Predict labels for any instance $\mathbf{x}$ via the classifier $\hat{y}_{\boldsymbol{\theta}}(\mathbf{x})$, and make predictions for groups via $\hat{y}_g = \mathcal{A}_g(\hat{y}_{\boldsymbol{\theta}}(\mathbf{x}))$

To optimize our cost function we use mini-batch stochastic gradient with momentum. For each mini-batch we do k iterations of optimization. We automatically select the algorithm parameters (learning rate, mini-batch size, k, λ) via linear grid search on the training data, using the group level accuracy of our training data as our performance measure. We found in general in our experiments that the results were relatively insensitive to the exact settings of the algorithm parameters.

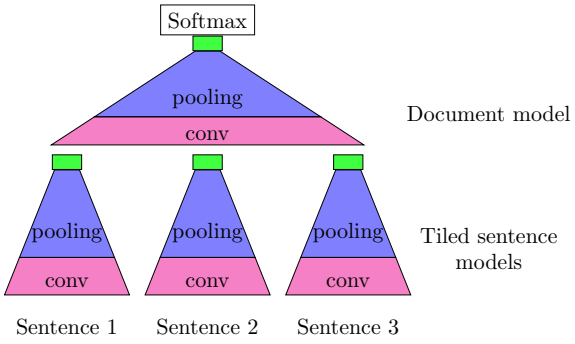


Figure 1: Model from Denil *et al.* [7]. The green squares indicate embedding vectors for sentences (atop the tiled sentence models) and for documents (atop the document model).

4. EMBEDDING NON-VECTOR INSTANCES WITH DEEP LEARNING

In practice data instances (such as sentences) are often not represented in vector form $\mathbf{x}$, and finding a good vector representation is not a a trivial problem. In this section we discuss generating feature vector representations using deep learning and text embeddings.

In particular we take advantage of recent advances in learning distributed representations for text. Early work on developing neural network representation for language dates back several decades [2, 11]. More recent work has shown that the semantic relationships of words can be effectively captured using the geometry of a continuous embedding space [6, 24, 23, 26].

Neural network models have also been used to build representations for larger blocks of text. A notable example of this is the paragraph vector [16] which extends the earlier work of Mikolov *et al.* [24] to simultaneously build representations for words and paragraphs. Another recent development in this direction is the work of Denil *et al.* [7] which uses a convolutional neural network to build representations for words, sentences and documents simultaneously. We adopt this convolutional neural network here for our experiments because it provides a useful method for obtaining representations for sentences.

The model of Denil et al. [7] is a multi-level convolutional network, which represents a document using a two stage process. The first stage represents each sentence in the document in an embedding space and the second stage then transforms the sentence embeddings into a representation for the full document. Both stages of the model are jointly trained to predict labels at the document level. The structure of this model is shown in Figure 1.

This model is ideally suited to our setting because it produces meaningful sentence embeddings as an intermediate representation without requiring sentence-level labels. Since we have document labels (for reviews) in our training data we can train the full model in a supervised way, and afterwards extract the intermediate sentence level representations to provide vector representations $\mathbf{x}$ for the cost function described in Section 3. We obtain these vector embeddings $\mathbf{x}$ with a simple forward pass through the trained convolutional network. Details of the network architecture and training are discussed in the next section.

Query Sentence	Most Similar Sentences	Least Similar Sentences
[Product works perfectly with jawbone 2.]	[Product works perfectly with my Blackberry 8830.] [It performs exactly as described.] [It works perfect!] [It works perfectly.]	[horrible horrible phone.] [Poor fit, poor reception, then it broke.] [Poor design and would not recommend don't waste your money]
[I seriously do not recommend these to anyone.]	[The hole don't line up.] [Product didn't resemble picture shown.] [I won't buy another.] [I don't recomend wasting your money on this.]	[Excellent Excellent Excellent wireless Bluetooth.] [Perfect custom shape fits my SLVR L7 perfectly with no problems.] [It fits perfectly, works perfectly and I'm thrilled!]
[Completely Worthless.]	[terrible case.] [Very poor product.] [THE WORST.] [Dont waste your money.] [Worst Phone Ever.]	[Excellent price, excellent battery, and extremely fast delivery.] [Excellent & timely service..] [Timely service, excellent price (can you beat it?]
[They are more comfortable.]	[People always say "Cool!"] [Pros:Pairing was easy.] [Shipping was super fast.] [worked as described.]	[Horrible horrible horrible.] [Poor quality, poor construction, poor fit, almost impossible to hear out of it.] [worst bluetooth ever full of static and very poor unless your sitting in a quiet room]

Table 1: Examples of most similar and least similar sentences, obtained by computing the similarity measure of Equation 2, applied to the sentence embeddings from the convolutional neural network of Figure 1

If the distributed representations are accurate then we should expect nearby points in embedding space to correspond to semantically similar sentences, making the similarity measures based on the Euclidean norm an appropriate measure of closeness. Table 1 shows some illustrative similarities between a query sentence and its most and least similar sentences in embedding space, produced by training a convNet as in Figure 1. The sentences are examples from Amazon reviews on cell phones.

5. EXPERIMENTS

5.1 Datasets

We performed our experiments on three real world datasets, and one artificially generated dataset.

Amazon: contains reviews and scores for products sold on amazon.com in the *cell phones and accessories* category, and is part of the dataset collected by McAuley and Leskovec [22]. Scores are on an integer scale from 1 to 5. We considered reviews with a score of 4 and 5 to be positive, and scores of 1 and 2 to be negative. We randomly partitioned the data into two halves of 50%, one for training and one for testing, with 35,000 documents in each set.

IMDb: refers to the IMDb movie review sentiment dataset originally introduced by Maas *et al.* [20] as a benchmark for sentiment analysis. This dataset contains a total of 100,000 movie reviews posted on `imdb.com`. There are 50,000 unlabeled reviews and the remaining 50,000 are divided into a set of 25,000 reviews for training and 25,000 reviews for testing. Each of the labeled reviews has a binary sentiment label, either positive or negative. In our experiments, we train only on the labelled part of the training set.

Yelp: refers to the dataset from the Yelp dataset challenge[2] from which we extracted the restaurant reviews. Scores are on an integer scale from 1 to 5. We again considered reviews with scores 4 and 5 to be positive, and 1 and 2 to be negative. We randomly generated a 50-50 training and testing split, which led to approximately 300,000 documents for each set.

Sentences: for each of the datasets above, we extracted and manually labeled 1000 sentences from the *test set*, with 50% positive sentiment and 50% negative sentiment. These sentences are only used to evaluate our instance-level classifier for each dataset[3]. They are not used for model training, to maintain consistency with our overall goal of learning at a group level and predicting at the instance level.

Newsdata: refers to a multi-instance variant of the well-known 20 newsgroups data set, where 20 different multi-instance learning problems are created by artificially creating groups of documents and aggregating labels to the group level. This data has been used frequently as a testbed in prior work on multi-instance learning [1, 12, 34]. For a fair comparison we used the feature vectors and groups used in prior work by Zhou *et. al* [34] among others, and which are available online[4].

5.2 Preprocessing

For all of our datasets (except the newsgroup data set, which is already in feature form) we use Beautiful Soup[5] to preprocess each review by removing the HTML markup, breaking the review into sentences, and then breaking each sentence into words. We also map numbers to a generic `NUMBER` token and any symbol that is not in `.?!` to `SYMBOL`. We replace all words that appear less than 5 times in the training set with `UNKNOWN`.

For the **IMDb** dataset we use the same model parametrization provided by Denil *et al.* [7] for this dataset, which yields a set of sentence embeddings, $\mathbf{x}_i \in \mathbb{R}^{24}$.

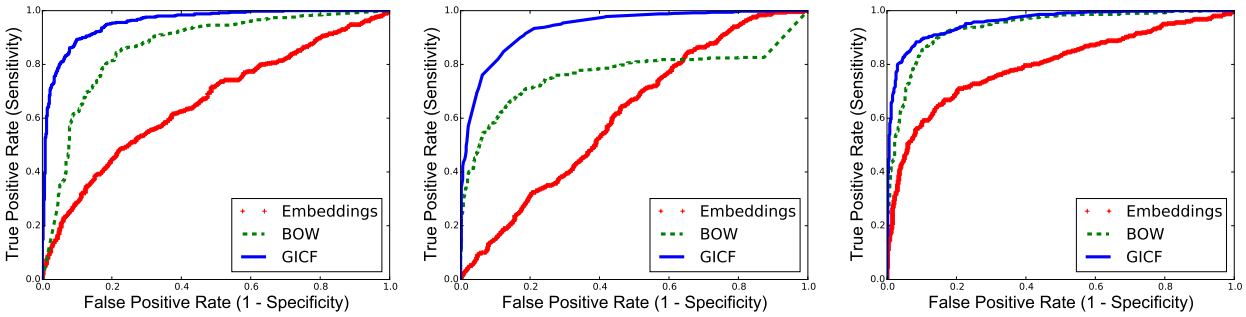
For the **Amazon** and **Yelp** datasets, because we had more data (compared to the **IMDb** data) we were able to increase the size of the convolutional network with a similar general neural architecture to that in [7]. We started with 20-dimensional word embeddings which are convolved with 10 feature maps of width 15, followed by a 7-max pooling layer and a *tanh* nonlinearity. The weights of this model are tied across sentences in a document. The document-level model convolves its input with a bank of 30 feature maps of width 9, followed by 5-max pooling and a *tanh* nonlinearity, which results in sentence embeddings of $\mathbf{x}_i \in \mathbb{R}^{150}$. We used

[2] `http://www.yelp.com/dataset_challenge`

[3] `http://archive.ics.uci.edu/ml/datasets/Sentiment+Labelled+Sentences`

[4] `http://lamda.nju.edu.cn/Data.ashx`.

[5] `http://www.crummy.com/software/BeautifulSoup/`



(a) Amazon sentence ranking (b) Imdb sentence ranking (c) Yelp Sentence ranking

Figure 2: ROC plots for instance level classification, for each of the baselines and our method for the three datasets

dropout values of 0.2 after the word level and 0.5 after the sentence level, and optimized the network using AdaGrad.

Training these networks also results in word and document embeddings which we did not use in this work.

5.3 From Review Sentiment to Sentence Sentiment

We use our manually labeled datasets of sentences, from the **IMDb**, **Amazon**, and **Yelp** data sets, in order to evaluate our performance on sentence-level sentiment classification. We refer to our method as Group-Instance Cost Function (GICF). For reference we compare the accuracy of our approach with a simple baseline method. For the baseline we train an L2-regularized logistic regression classifier on a bag of words (BOW) representation at the document level, and then apply this document-level classifier to BOW representations of BOW sentences. Although we would not necessarily expect this approach to perform well, this is the simplest and most direct way to go from document-level supervision to sentence-level testing. Each document and sentence is represented as a vector of word counts. We optimize the logistic regression model using batch gradient descent and use a validation set of 10% of the training data to learn the relative weight for the L2 regularization term in the cost function.

As a second baseline we use the same logistic regression setup, but use as features the embedding vectors which are provided by the ConvNet, where we average all the sentence vectors to produce the vector of a document. In this way we can evaluate the relative contribution of the embedding method on its own.

Figure 2 and Table 2 shows ROC plots and test set accuracies (respectively) for each dataset, for our method (GICF) and the two baselines. The results overall show that our proposed approach attains relatively high levels of accuracy results across all 3 datasets. Using the embeddings with logistic regression yields the poorest results. One reason could be that averaging the embeddings of sentences (to create the representation of a document) although intuitive is suboptimal. Logistic regression with a bag-of-words representation is more accurate, but not as accurate as our approach. This is likely to due to the fact that there is less signal, as the bag-of-word vectors are very sparse at a sentence level, which is not an issue for our approach.

To further evaluate the quality of our sentence predictions, we compared the performance of our approach with the Sentiment Analysis tool described in Socher *et al.* [29], on the **IMDb** movie data. This tool is pre-trained on movie data and available online through a web interface[6]. We use this interface to obtain predicted labels for our **IMDb** sentence test data. It is worth noting that this method is trained with supervision at the phrase-level, whereas we only require supervision at the review level. It is expensive to obtain labels at the phrase-level, but there exist millions, perhaps billions, of labeled reviews online.

	Amazon	IMDb	Yelp
Logistic w/ BOW	79.0%	76.2%	75.1%
Logistic w/ Embeddings	54.3%	57.9%	66.5%
GICF w/Embeddings	**88.2%**	**86.0%**	**86.3%**

Table 2: Accuracy Results on the instance level classification i.e., sentences between the baselines and our method (GICF)

Their method generates the probability of a sentence belonging to the following five classes: `[Very Negative, Negative, Neutral, Positive, Very Positive]` and chooses the class with the highest probability as the predicted class. To convert this output to a binary decision, we count both `Positive` and `Very Positive` labels as positive, and do the same for negative labels. When we run our 1000 test sentences from **IMDb** through the Socher et al. sentiment analysis tool, it predicts `Neutral` for 238 of these sentences.

To compare our approach with the Socher et al. output, we evaluate two different ways of treating `Neutral` predictions. In the first approach, we follow the strategy in the Socher et al. paper [29] which rejects these sentences and calculates the accuracy of the method on the remaining 762 sentences, using an effective rejection rate of 23.8%. To compare our method in the same manner, our algorithm rejects the 238 sentences for which its estimated probabilities are closest to 0.5 (its best estimate of which sentences are neutral), within the range $[0.5 - b, 0.5 + b]$, where $b = 0.048$. In the second approach, neither algorithm rejects any sentences (0% reject rate), and accuracy is calculated on all 1000 sentences. To generate a prediction for the Socher et al. method for the 238 sentences where it predicts `Neutral` with the highest probability, we select the class that it predicts with the 2nd highest probability.

At a rejection rate of 23.8% our approach is significantly more accurate than the Socher et al. method, achieving an

[6] `http://nlp.stanford.edu/sentiment/`, accessed on June 20th, 2014

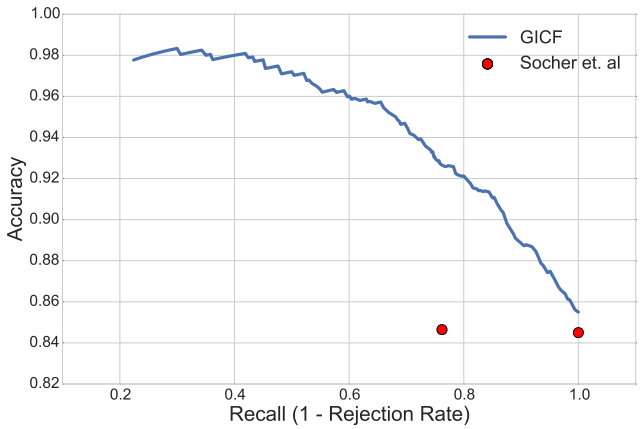


Figure 3: Accuracy of sentence classification, for various levels of rejection rate (neutral sentences) in **IMDb** dataset.

accuracy of 92.6% compared to 84.7%. At a rejection rate of 0% our approach is slightly more accurate, achieving an accuracy of 85.7% compared to 85.5%. Figure 3 further illustrates the accuracy-rejection tradeoff of our approach resulting from increasing the boundary parameter b. The two red dots indicate the precision from the two reject options for the Socher et al. method, of 0% and 23.8%, as described above. As we increase our boundary b, we see the rejection rate increasing (as the algorithm ignores more boundary cases) accompanied by a general increase in accuracy. These results suggest that while our method requires much less supervision, it is able to obtain comparable or better sentiment predictions for sentences than a state-of-the-art supervised learning approach.

5.4 Group Level Prediction

As we have seen above, the proposed method works well in transferring information from the group level to the instance level for predicting review sentiment. As a sanity check, we also evaluate the performance of our model as a group (review) classifier. To accomplish this, we utilize the aggregation function on the predicted sentence scores $\hat{y}$ in each review to classify the test set reviews. Since labels exist at the group level for all of our datasets we can directly measure the accuracies of our classifiers.

We compare the results of our method with the same baseline classifiers described earlier, namely the BOW or embedding representations with an L2-regularized logistic regression classifier.

For the BOW baseline, we generate predictions for test documents in two ways. In the first we consider the same BOW and embedding representation of each test document and classify it directly. In the second approach we aggregate the sentence scores of each document, in the same manner as we use in our method, i.e., we consider each sentence to be a document, score it, and then average all the scores for sentences in each document before thresholding at 0.5.

Table 3 reports the classification accuracy and the area under the curve (AUC) for all methods. As previously, the logistic regression on the embeddings underperforms all other methods. The logistic regression that attempts to go through the sentences to reach the group-level prediction performs significantly worse than the simple logistic regression. This is expected as the signal in each sentence may not be enough to correctly classify it, when using logistic regression. Our approach provides better accuracy and AUC results in two out the three datasets, despite going through the sentences to achieve group classification. This provides evidence that the method has been successful in transferring information from the group to the instance level, in that it has been able to transfer the review labels to the sentence level in the training data, and then back again from sentences to reviews in the test data.

5.5 Comparison with other MIL Algorithms

In our final experiment we use the **Newsdata** to evaluate the efficacy of our cost function, without the use of the information from the embeddings. We use the same feature vectors and groupings as those used in [34], which are available online. These vectors represent the top $tf-idf$ terms in the corpus and are very sparse. Groups were generated under a typical MIL assumption, namely that negative groups contain only negative instances, and positive groups may contain relatively few positive instances. Specifically, the positive groups were created in a way that $p = 3\%$ of the instances are positive and the other 97% are negative [34]. To account for this imbalance we changed our cost function to incorporate this information, by penalizing false positives by $1/p$ more than false negatives. Hence for a group $\mathcal{G}_k$, the second part of the equation becomes

$$\Delta_2(\hat{\ell_k}, \ell_k) = \big(\mathcal{A}(\mathcal{G}_k, \boldsymbol{\theta}) - \frac{1}{p}\ell_g\big)^2$$

(and the rest of the cost function remains the same as before). Table 4 compares the results for our algorithm with a variety of other well-known MIL algorithms, using the AUC-PR score and results reported in [12]. The first 5 algorithms in the table perform relatively poorly since they are designed for predicting group-level labels rather than instance level labels, reinforcing the point that group-level prediction algorithms are not being optimized for making instance-level predictions. The next 3 algorithms, VF, VFr, and DPMIL, are all designed for instance-level predictions in an MIL context, and perform better than the other 5 more traditional MIL algorithms. Despite the fact that we are using a relatively simple feature space (compared to the embedding features), our approach nonetheless performs significantly better than almost all of these approaches (and just slightly better than the best alternative method DPMIL [12], which is the only method which considers the imbalance of positive instances in groups when solving the problem).

5.6 Illustrative Applications

This section discusses some illustrative applications that are possible given the ability to attribute the group score to its instances. In our first example, Figure 4 illustrates the predicted sentiment for the sentences in a review[7] from the **IMDb** test set. Sentences highligthed in green correspond to sentences labeled as positive by our model while sentences in red font correspond to those labeled negative. This is a particularly tricky example, as it contains both positive and negative sentences. Our model is able to identify correctly both types. Moreover, most of the words in the review are negative, hence the naive strategy of a simple count of words

[7] `http://www.imdb.com/title/tt0974014/reviews`

	Accuracy			AUC		
	Amazon	IMDb	Yelp	Amazon	IMDb	Yelp
Logistic w/ BOW on Documents	85.8%	86.20%	**91.25%**	88.08%	88.32	**94.41**
Logistic w/ BOW on Sentences	88.3%	81.81%	78.16%	87.19%	82.67	67.87
Logistic w/ Embeddings on Documents	67.82%	58.23%	81.00%	61.24%	60.77	82.59
GICF w/ Embeddings on Sentences	**92.8%**	**88.56%**	88.73 %	**91.73%**	**88.36%**	92.36%

Table 3: Accuracy and Area-Under-the-Curve (AUC) scores for predicting labels at the group (document) level for the baselines and our proposed method (GICF). Training is always done at the group level. Testing on sentences corresponds to scoring each sentence separately and aggregating the results. BOW or embeddings corresponds to the features used.

MIL Algorithm	Average AUC-PR
MISVM	0.26
miSVM [1]	0.45
GPMIL	0.40
I-KISVM	0.43
B-KISVM	0.47
VF	0.59
VFr [19]	0.67
DPMIL [12]	0.70
GICF	**0.71**

Table 4: Scores for Area Under the Precision-Recall Curve, for a variety of MIL algorithms, averaging over all the datasets in the **Newsdata** corpus. Scores for each algorithm were obtained from [12], apart from the score for our algorithm.

or sentences would misclassify this, since the reviewer gave a rating of 8/10. Our approach on the other hand enables us to extract sentences that best reflect the sentiment of the entire review, and score them at the same time. Averaging the predicted sentence score, correctly classifies this as a positive review.

Figure 5 shows a review from the **Yelp** dataset. Despite the short length of the review, our classifier is able to appropriately classify each sentence. Scoring multiple sentences separately, in reviews of a specific restaurant for example, would allow users to perform feature queries. In this way, they would get a positive score when the query was about *espresso*, but a negative score for *service* or *employees*. Such detailed information cannot be obtained at the review level.

Similarly, Figure 6 is a review that describes a mixed experience of a customer. Once again apart from classifying the whole review, we get an appropriate sentiment for each sentence, despite the fact that the sentiment of the review is very mixed.

Finally, in Figure 7 we illustrate how this is also successful in larger, more intertwined, and complicated reviews.

6. SCALABILITY

The proposed method is capable of scaling well to large data sets. For example for the **Yelp** dataset with 300k training reviews, optimizing the cost function finishes in the order of a few minutes using a consumer-grade desktop computer. In contrast, many of the methods described in our discussion of related work in Section 2 do not scale well (see [30] for a detailed discussion).

In addition, in terms of optimizing our cost function, we can gain a very significant speed up over standard batch gradient descent by using mini-batch stochastic gradient meth-

Paul Bettany did a great role as the tortured father whose favorite little girl dies tragically of disease.
For that, he deserves all the credit.
However, the movie was mostly about exactly that, keeping the adventures of Darwin as he gathered data for his theories as incomplete stories told to children and skipping completely the disputes regarding his ideas.
Two things bothered me terribly: the soundtrack, with its whiny sound, practically shoving sadness down the throat of the viewer, and the movie trailer, showing some beautiful sceneries, the theological musings of him and his wife and the enthusiasm of his best friends as they prepare for a battle against blind faith, thus misrepresenting the movie completely.
To put it bluntly, if one were to remove the scenes of the movie trailer from the movie, the result would be a non descript family drama about a little child dying and the hardships of her parents as a result.
Clearly, not what I expected from a movie about Darwin, albeit the movie was beautifully interpreted.

Figure 4: For this review, which is labeled as positive, our approach assigns positive sentiment to the first two and last sentences of the review. The remaining three sentences are assigned negative sentiment.

don't bother.
the employees are the worst.
it's quite sad actually because the espresso drinks are out of this world, amazing.
still, i won't be back.
ever.

Figure 5: A short review from the **Yelp** dataset

ods. For the gradient descent method, for each iteration, the dominant complexity term in the derivative of the cost function is $\mathcal{O}(n^2)$, due to the computation of the similarity matrix between all pairs of n instances. In a stochastic gradient framework using a mini-batch of size b this can be reduced to $\mathcal{O}(b^2)$. For an epoch (visiting all n data points) the complexity is $\mathcal{O}(\frac{n}{b}b^2) = \mathcal{O}(nb)$. In contrast, the direct SVM-based approach is $\mathcal{O}(n^3)$ or $\mathcal{O}(b^3)$ More significantly, from a practical viewpoint, we have found in our experiments that the results are not particularly sensitive to the size b of the mini-batch. Figure 8 illustrates the training accuracy compared to actual time, for various batch sizes, for **Yelp**, our biggest dataset.

The cost of pairwise comparisons for minibatches, $\mathcal{O}(b^2)$, can likely be further reduced (e.g., if there are problems that require relatively large values of b). Given that the similarity kernel matrix is essentially a search for k nearest neighbors, where high similarity values are the ones that matter the most in the cost function, one could in principle

It's not awful, but nothing really wowed me about this place.
The chips and salsa are good, and their white jalapeno sauce on my chimi was great.
The drinks, although HUGE, were pretty watered down and not alot of kick to them.
Probably not somewhere I would recommend to anyone for Mexican food.

Figure 6: A short review, labeled as negative, where all of the positive sentences are in the middle

There is only 1 good thing about this place.
The decoration of the place was very artsy.
It made me felt like I was in a fish bowl or something, I felt I was trapped in some sort of container.
It was weird, but very interesting at the same time.
The food was not as good as I thought it was going to be.
As you enter the restaurant, you would notice a few chefs working at the bar making fresh noodles or some dim sum.
However, after ordering a bowl of braised beef noodle soup, I wasn't able to tell the difference between package noodles and the fresh hand made ones.
The soup base was a very plain beef soup, nothing special about it.
The braised beef was juicy, (probably from sitting in the soup for so long) but it was flavorless and there were no sauce that accompany my order to improve the taste.
The price was extremely over price, especially for the quality of our food.
My noodle soup was a small size and it was about $16.
If the food was a lot better and if the waiters pay more attention to us, it would of POSSIBLY been worth the money.
I also ordered a cocktail, (Shang Hai Wave) it was very delicious, but it was about $12.
It had a strong lychee flavor which was delicious and it has a sweet finish.
DON'T EAT HERE.
Reasons why:
- Food was way over priced
- Service was not extraordinary
- Quality of food was deceiving

Figure 7: A complicated and large review, labeled overall as negative, with positive and negative sentences intertwined

approximate this matrix and retrieve it very quickly with methods such as asymmetric locality sensitive hashing [28].

The other cost for our procedure, for problems where instances need to be represented as vectors using deep learning, is the cost of the convolutional network embedding. Fortunately, training deep neural networks is a problem where significant effort has been expended to develop fast parallel algorithms for training such models (e.g., in image analysis). In our case the convolutions are small operations that can be massively parallelized using the small computational units of a GPU. We were able to train the ConvNet from [7] on our biggest dataset in a matter of a few hours in an Amazon Web Services GPU-optimized server.

7. CONCLUSION

We presented a general framework for transferring label information from groups of instances to individual instances. Our approach relies on a flexible cost function that can learn instance-level classifiers by leveraging both instance-level similarity and group-level label information. In particular, we showed that embeddings obtained from deep learning can be a useful tool for providing effective vector representations for instances in this context. Experimental results on sentiment prediction for reviews showed that the approach is accurate across a range of different real-world data sets and more accurate than a large range of competing approaches. Directions for further investigation include exploring different choices of classifiers, using different embedding approaches, understanding the limitations they impose on our model, as well as the development of new applications using this general framework.

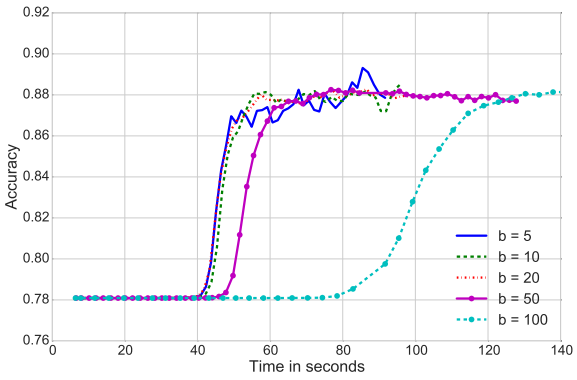


Figure 8: Accuracy vs Time in Seconds, for different batch sizes, for the **Yelp** Dataset

8. ACKNOWLEDGEMENTS

The authors would like to thank Eric Nalisnick, Jihyun Park, Kevin Bache and Moshe Lichman for their help in the manual tagging of sentences and fruitful discussions over the course of this work. This material is based upon work partially supported by the US Office of Naval Research under MURI award number N00014-08-1-1015 and by the US National Science Foundation under award number IIS-1320527

9. REFERENCES

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

[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003.

[3] R. C. Bunescu and R. J. Mooney. Multiple instance learning for sparse positive bags. In *International Conference on Machine Learning*, International Conference on Machine Learning, pages 105–112, New York, NY, USA, 2007. ACM.

[4] O. Chapelle and A. Zien. Semi–supervised classification by low density separation. In *International Workshop on Artificial Intelligence and Statistics*, pages 57–64, 2005.

[5] V. Cheplygina, D. M. Tax, and M. Loog. On classification with bags, groups and sets. *arXiv preprint arXiv:1406.0281*, 2014.

[6] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch.

[7] M. Denil, A. Demiraj, and N. de Freitas. Extraction of salient sentences from labelled documents. Technical report, University of Oxford, 2014.

[8] T. G. Dietterich, R. H. Lathrop, T. Lozano-Perez, and A. Pharmaceutical. Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89:31–71, 1997.

[9] J. Foulds and E. Frank. A review of multi-instance learning assumptions. *The Knowledge Engineering Review*, 25(01):1–25, 2010.

[10] T. Gartner, P. A. Flach, A. Kowalczyk, and A. J. Smola. Multi-instance kernels. In *In Proc. 19th International Conf. on Machine Learning*, pages 179–186. Morgan Kaufmann, 2002.

[11] G. E. Hinton. Learning distributed representations of concepts. In *Annual Conference of the Cognitive Science Society*, pages 1–12, 1986.

[12] M. Kandemir and F. A. Hamprecht. Instance label prediction by Dirichlet process multiple instance learning. In *Uncertainty in Artificial Intelligence*, 2014.

[13] D. Kifer. Attacks on privacy and de Finetti's theorem. In *International Conference on Management of Data*, pages 127–138, 2009.

[14] H. Kueck, P. Carbonetto, and N. Freitas. A constrained semi-supervised learning approach to data association. In *European Conference on Computer Vision*, pages 1–12, 2004.

[15] H. Kueck and N. de Freitas. Learning about individuals from group statistics. In *Uncertainty in Artificial Intelligence*, pages 332–339, 2005.

[16] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, volume 32, pages 1188–1196, 2014.

[17] Y. Li, J. Hu, Y. Jiang, and Z. Zhou. Towards discovering what patterns trigger what labels. In *Conference on Artificial Intelligence*, 2012.

[18] Y.-F. Li, J. T. Kwok, I. W. Tsang, and Z.-H. Zhou. A convex method for locating regions of interest with multi-instance learning. In *European Conference on Machine Learning and Knowledge Discovery in Databases: Part II*, European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, pages 15–30, 2009.

[19] G. Liu, J. Wu, and Z. Zhou. Key instance detection in multi-instance learning. In *Asian Conference on Machine Learning*, pages 253–268, 2012.

[20] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, 2011.

[21] O. Maron and A. L. Ratan. Multiple-instance learning for natural scene classification. In *International Conference on Machine Learning*, pages 341–349, 1998.

[22] J. McAuley and J. Leskovec. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Conference on Recommender Systems*, RecSys '13, pages 165–172, New York, NY, USA, 2013. ACM.

[23] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Neural Information Processing Systems*, pages 3111–3119, 2013.

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *International Conference on Learning Representations*, 2013.

[25] G. Patrini, R. Nock, T. Caetano, and P. Rivera. (almost) no label no cry. In *Advances in Neural Information Processing Systems 27*, pages 190–198. Curran Associates, Inc., 2014.

[26] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543. Association for Computational Linguistics, October 2014.

[27] N. Quadrianto, A. J. Smola, T. S. Caetano, and Q. V. Le. Estimating labels from label proportions. *Journal of Machine Learning Research*, 10:2349–2374, 2009.

[28] A. Shrivastava and P. Li. Asymmetric LSH (ALSH) for sublinear time maximum inner product search (mips). In *Advances in Neural Information Processing Systems 27*, pages 2321–2329. Curran Associates, Inc., 2014.

[29] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, 2013.

[30] X.-S. Wei, J. Wu, and Z.-H. Zhou. Scalable multi-instance learning. In *International Conference on Data Mining*, pages 1037–1042, 2014.

[31] N. Weidmann, E. Frank, and B. Pfahringer. A two-level learning method for generalized multi-instance problems. In *European Conference on Machine Learning*, volume 2837, pages 468–479, 2003.

[32] X. Xu and E. Frank. Logistic regression and boosting for labeled bags of instances. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 272–281, 2004.

[33] F. X. Yu, D. Liu, S. Kumar, T. Jebara, and S.-F. Chang. ∝svm for learning with label proportions. In *International Conference on Machine Learning*, volume 28, pages 504–512, 2013.

[34] Z.-H. Zhou, Y.-Y. Sun, and Y.-F. Li. Multi-instance learning by treating instances as non-iid samples. In *International Conference on Machine Learning*, pages 1249–1256. ACM, 2009.

[35] Z.-H. Zhou and J.-M. Xu. On the relation between multi-instance learning and semi-supervised learning. In *International Conference on Machine Learning*, pages 1167–1174, 2007.