2014 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, SEPT. 21–24, 2014, REIMS, FRANCE

KERNEL-BASED INSTANCE ANNOTATION IN MULTI-INSTANCE MULTI-LABEL LEARNING

Anh T. Pham and Raviv Raich

School of EECS, Oregon State University, Corvallis, OR 97331-5501
{phaman, raich}@eecs.oregonstate.edu

ABSTRACT

Multi instance multi label learning is a framework in which objects are represented as bags of instances and labels are provided at the bag level. Instance annotation is the problem of assigning labels to the instances in a bag given only the bag label. Recently, OR-ed logistic regression (OR-LR) model and an EM based inference method have been proposed for instance annotation. Due to the linear nature of the logistic regression function, OR-LR performance on linearly inseparable data is limited. This paper addresses this problem by proposing a regularized kernel-based extension to the OR-LR framework. Experiments show that the kernel-based OR-LR algorithm achieves a significant improvement in classification accuracy over the linear OR-LR from 3% to 9% on audio bird song and image annotation datasets and two synthetic datasets.

Index Terms— Instance annotation, kernel-based learning, regularization, multi instance multi label learning

1. INTRODUCTION

Multi instance multi label learning (MIML) is a framework of classification under label ambiguity. Specifically, in MIML learning, the training data consists of multiple bags and each bag contains multiple instances. The label for each instance in a bag is unavailable. Instead, a bag label is provided for the entire bag. The MIML setting introduces several classification tasks. One task deals with the prediction of the bag label for an unlabeled bag given previously observed labeled bags. Another problem is instance label prediction given only bag labels. This problem, commonly referred to as *instance annotation*, is the focus of this paper.

Many algorithms for bag label prediction in the MIML setting have been proposed [1, 2]. However, many bag label prediction algorithms cannot be easily modified to perform instance annotation. For example, MIMLSVM [2] is a MIML algorithm designed for bag label learning. The algorithm uses the Hausdorff distance among bags to train an SVM-based classifier. Since MIMLSVM does not base the bag label prediction on instance label predictions, it cannot be easily modified for instance annotation.

Although very few algorithms address the instance annotation problem in MIML learning, some of the bag label prediction algorithms can be modified to perform instance label predictions. For example, in MIMLfast [3], for each class a bag level score is computed by taking the maximum over the instance level scores. The instance level class score functions can be used to perform instance annotation. To avoid ignoring other instances in the bag, rank-loss support instance machine (SIM) [4] considers a softmax score to include all the instances with corresponding weights.

Many of the aforementioned algorithms use approximation techniques, e.g., MIMLSVM deploys k-centroid clustering and MIMLfast uses sampling to reduce computational complexity, which may degrade the classification accuracy. Recently, an OR-ed logistic regression (OR-LR) discriminative model and an exact inference method [5] have been proposed for instance annotation. Due to the linear relation in the logistic regression model, OR-LR is best suited for problems in which linear classifiers achieve high accuracy.

To extend the results to linearly inseparable data [6], we propose a kernel extension to the OR-LR algorithm. Our contribution in this paper is three-fold. First, we develop a kernel version of the OR-LR, which can work well on linearly inseparable data. Second, to avoid overfitting, we propose an $L_{2,1}$ norm regularization. Additionally, the norm encourages dependence of the classifier on only a few instances yielding computationally efficient classification. Third, we empirically evaluate the efficiency of our algorithm by running experiments on bird song and image annotation datasets and two synthetic datasets. We demonstrate significant improvements in accuracy over MIMLfast and the linear version of the OR-LR algorithm.

The organization of the paper is as follows. Section 2 provides the background about the OR-LR framework, kernel-based learning, regularization, and a motivating example. Section 3 derives the kernel extension of the OR-LR. Numerical evaluation of the proposed framework is presented in Section 4. Finally, Section 5 concludes the paper.

2. BACKGROUND

Our goal is to develop methods for training nonlinear kernel-based instance level classifiers for instance annotation under label ambiguity. Moreover, we would like to demonstrate that by replacing linear classifiers with kernel-based classifiers for instance annotation in the MIML setting, significant performance improvements can be achieved. To that end, we begin by reviewing the OR-LR framework. We continue with a review of kernel-based learning and regularization. Finally, we provide a motivating example to illustrate the potential improvements that can be achieved using kernel-based learning on single instance single label setting (SISL) in instance annotation problems.

2.1. OR-LR for instance annotation in MIML

To perform instance annotation in the MIML setting, the challenging task of training an instance level classifier from ambiguous labels should be addressed. In [5], a discriminative model and an EM inference approach are developed to train a logistic regression instance level classifier. The model is shown in Fig. 1.

978-1-4799-3694-6/14/$31.00

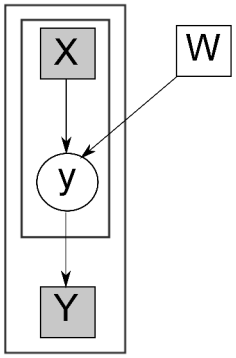

Fig. 1: OR-ed logistic regression graphical model for MIML data. Observed instances and bag labels are shaded.

2.1.1. OR-LR model

Consider training data consisting of B labeled bags denoted by $\{(\mathbf{X}_b, \mathbf{Y}_b)\}_{b=1}^{B}$. Each bag contains n_b instances, i.e., $\mathbf{X}_b = \{\mathbf{x}_{b1}, \mathbf{x}_{b2}, \ldots, \mathbf{x}_{bn_b}\}$. The bag label $\mathbf{Y}_b$ is the union of its instance labels $\{\mathbf{y}_{b1}, \mathbf{y}_{b2}, \ldots, \mathbf{y}_{bn_b}\}$. Recovering $\mathbf{y}_{bi}$ is the goal of instance annotation.

The model assumes that the instance labels are independent. Moreover, the label of each instance $\mathbf{y}_{bi}$ given its instance feature vector $\mathbf{x}_{bi}$ follows a logistic regression function

$$p(\mathbf{y}_{bi}|\mathbf{x}_{bi}, \mathbf{W}) = \frac{\prod_{c=1}^{C} e^{I(\mathbf{y}_{bi}=c)\mathbf{w}_c^T \mathbf{x}_{bi}}}{\sum_{c=1}^{C} e^{\mathbf{w}_c^T \mathbf{x}_{bi}}}, \tag{1}$$

where $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_C]$. Furthermore, OR-LR also assumes that the label of the bag is the union of its instance labels $\mathbf{Y}_b = \bigcup_{i=1}^{n_b} \mathbf{y}_{bi}$, which can be mathematically expressed as follows

$$p(\mathbf{Y}_b|\mathbf{y}_{b1}, \mathbf{y}_{b2}, \ldots, \mathbf{y}_{bn_b}) = \Psi(\mathbf{Y}_b, \mathbf{y}_{b1}, \mathbf{y}_{b2}, \ldots, \mathbf{y}_{bn_b}), \tag{2}$$

where $\Psi(\mathbf{Y}_b, \mathbf{y}_{b1}, \mathbf{y}_{b2}, \ldots, \mathbf{y}_{bn_b})$ is given by

$$\Psi(\mathbf{Y}_b, \mathbf{y}_{b1}, \mathbf{y}_{b2}, \ldots, \mathbf{y}_{bn_b}) = \begin{cases} 1 & \text{if } \mathbf{Y}_b = \bigcup_{i=1}^{n_b} \mathbf{y}_{bi} \\ 0 & otherwise. \end{cases}$$

From (1), (2), and the dependence assumptions of the OR-LR model, the log-likelihood can be expressed as

$$\log(p(\mathbf{Y}_D|\mathbf{X}_D, \mathbf{W})) = \sum_{b=1}^{B} \log(p(\mathbf{Y}_b|\mathbf{X}_b, \mathbf{W})) = \tag{3}$$
$$\sum_{b=1}^{B} \log[\sum_{\mathbf{y}_{b1}=1}^{C} \cdots \sum_{\mathbf{y}_{bn_b}=1}^{C} \prod_{i=1}^{n_b} p(\mathbf{y}_{bi}|\mathbf{x}_{bi}, \mathbf{W}) \Psi(\mathbf{Y}_b, \mathbf{y}_{b1}, \ldots, \mathbf{y}_{bn_b})],$$

where $(\mathbf{X}_D, \mathbf{Y}_D) \triangleq \{(\mathbf{X}_b, \mathbf{Y}_b)\}_{b=1}^{B}$. To train the LR classifier under the MIML setting, the maximum likelihood estimator of $\mathbf{W}$ is obtained using the EM algorithm.

2.1.2. OR-LR inference

The surrogate function of the log-likelihood is expressed as

$$g(\mathbf{W}, \mathbf{W}') = E_{\mathbf{y}}[\log p(\mathbf{Y}_D, \mathbf{y}|\mathbf{X}_D, \mathbf{W})|\mathbf{Y}_D, \mathbf{X}_D, \mathbf{W}'] \tag{4}$$
$$= \sum_{b=1}^{B} \sum_{i=1}^{n_b} [\sum_{c=1}^{C} p(\mathbf{y}_{bi} = c|\mathbf{Y}_b, \mathbf{X}_b, \mathbf{W}') \mathbf{w}_c^T \mathbf{x}_{bi} - \log(\sum_{c=1}^{C} e^{\mathbf{w}_c^T \mathbf{x}_{bi}})]$$
$$+ \zeta,$$

where ζ is a constant independent of $\mathbf{W}$. The expectation and maximization steps in the EM-algorithm for the OR-LR model are as

- E-step: Compute $p(\mathbf{y}_{bi} = c|\mathbf{Y}_b, \mathbf{X}_b, \mathbf{W}^{(k)})$
- M-step: Find $\mathbf{W}^{(k+1)}$ s.t. $g(\mathbf{W}^{(k+1)}, \mathbf{W}^{(k)}) \geq g(\mathbf{W}^{(k)}, \mathbf{W}^{(k)})$.

While the M-step can be implemented using a gradient method, the E-step is challenging. Inefficient calculation of the probability such as in (3) may run in $O(C^{n_b})$ per bag. To resolve this challenge, [5] proposes a dynamic programming approach.

2.1.3. Dynamic programming for the E-step in OR-LR inference

OR-LR inference algorithm uses a forward algorithm to compute $p(\mathbf{y}, \mathbf{Y}_D|\mathbf{X}_D, \mathbf{W})$ and then obtain $p(\mathbf{y}|\mathbf{Y}_D, \mathbf{X}_D, \mathbf{W})$ using Bayes rule. Assuming an arbitrary order on a bag, OR-LR denotes the label for the sub-bag containing the 1st up to the jth instances as $\mathbf{Y}_b^j$ and $\mathbf{X}_b^j$. Moreover, [5] swaps the ith instance to the last instance, then incrementally computes $p(\mathbf{Y}_b^{j+1}|\mathbf{X}_b^{j+1}, \mathbf{W})$ from $p(\mathbf{Y}_b^j|\mathbf{X}_b^j, \mathbf{W})$, for j = 1 to $n_b - 2$, as follows

$$\begin{aligned} &p(\mathbf{Y}_b^{j+1} = \mathbf{L}|\mathbf{X}_b^{j+1}, \mathbf{W}) \\ &= \sum_{c \in \mathbf{L}} [p(\mathbf{y}_{b(j+1)} = c|\mathbf{x}_{b(j+1)}, \mathbf{W}) p(\mathbf{Y}_b^j = \mathbf{L}|\mathbf{X}_b^j, \mathbf{W}) \\ &+ p(\mathbf{y}_{b(j+1)} = c|\mathbf{x}_{b(j+1)}, \mathbf{W}) p(\mathbf{Y}_b^j = \mathbf{L}_{\setminus c}|\mathbf{X}_b^j, \mathbf{W})], \end{aligned} \tag{5}$$

where $\mathbf{L}_{\setminus c}$ consists of all labels in $\mathbf{L}$ except c. Finally, the desired probability is computed by considering the last instance as follows

$$\begin{aligned} &p(\mathbf{y}_{n_b} = c, \mathbf{Y} = \mathbf{L}|\mathbf{X}, \mathbf{W}) \\ &= p(\mathbf{y}_{n_b} = c|\mathbf{x}_{n_b}, \mathbf{W}) [p(\mathbf{Y}^{n_b - 1} = \mathbf{L}|\mathbf{X}^{n_b - 1}, \mathbf{W}) \\ &+ p(\mathbf{Y}^{n_b - 1} = \mathbf{L}_{\setminus c}|\mathbf{X}^{n_b - 1}, \mathbf{W})]. \end{aligned} \tag{6}$$

The aforementioned mechanism reduces the computational complexity associated with the E-step from exponential to quadratic in the number of instances per bag. Moreover, the instance level classification accuracy obtained with this tuning free algorithm is larger than that of SIM [4] by 4%-14% on four datasets.

2.2. Kernel-based learning

Kernel-based learning can offer improvement in classification accuracy over linear classification when the data is linearly inseparable [6, 7, 8], and specifically for logistic regression classification [9]. Kernel-based learning transforms the data from the original feature space into a higher dimension. In the new space, the data is linearly separable making it possible for linear classifiers to be applied successfully. Consider a simple two-class logistic regression function as follows

$$p(\mathbf{y} = 1|\mathbf{x}, \mathbf{w}) = \frac{1}{1 + e^{\mathbf{w}^T \mathbf{x}}}. \tag{7}$$

Replacing $\mathbf{x}$ with $\phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_l(\mathbf{x})]^T$, a high dimensional mapping function and setting $\mathbf{w} = \sum_i^N \alpha_i \phi(\mathbf{x}_i)$ in (7), we obtain the kernel-based logistic function

$$p(\mathbf{y} = 1|\mathbf{x}, \alpha) = \frac{1}{1 + e^{\sum_i^N \alpha_i K(\mathbf{x}, \mathbf{x}_i)}} = \frac{1}{1 + e^{\alpha^T \mathbf{k}(\mathbf{x})}}, \quad (8)$$

where $K(\mathbf{x}, \mathbf{x}_i) = \phi(\mathbf{x})^T \phi(\mathbf{x}_i)$ and $\mathbf{k}(\mathbf{x})$ is $[K(\mathbf{x}, \mathbf{x}_1), K(\mathbf{x}, \mathbf{x}_2), \ldots, K(\mathbf{x}, \mathbf{x}_N)]^T$. The kernel-based logistic regression in (8) does not require to compute the high dimensional $\phi(\mathbf{x})$. Instead, only the dot product $K(\cdot, \cdot)$ is evaluated. In this paper, we consider linear kernel and radial basis function (RBF) kernel as follows:

- Linear kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$.
- RBF kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \cdot \delta}$.

The increase in dimension associated with the kernel trick results in a large number of model parameters. Such increase in model complexity may result in overfitting.

2.3. Regularization

Regularization is commonly applied to prevent overfitting in the high dimensional low sample size data. When minimizing an empirical risk, a regularization term is appended to the empirical risk term. For example, training a linear classifier $\mathbf{w}^T \mathbf{x}$ can be accomplished by minimizing the following objective

$$\frac{1}{n} \sum_{i=1}^{n} loss(\mathbf{w}^T \mathbf{x}_i, \mathbf{y}_i) + \lambda Reg(\mathbf{w}), \quad (9)$$

where $Reg(\mathbf{w})$ can be the L_2 norm, $\|\mathbf{w}\|_2 = (\sum_{i=1}^d \mathbf{w}_i^2)^{1/2}$, or L_1 norm, $\|\mathbf{w}\|_1 = \sum_{i=1}^d |\mathbf{w}_i|$. A multi-class classifier is parameterized by $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_C]$ in which each column is the parameter vector for the corresponding class. One can compute a group norm $L_{2,1}$ as follows

$$\|\mathbf{W}\|_{2,1} = \|\mathbf{w}^1\|_2 + \|\mathbf{w}^2\|_2 + \cdots + \|\mathbf{w}^d\|_2, \quad (10)$$

where $\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^d$ are the rows of $\mathbf{W}$. The norm in (10) can be viewed as the L_1 norm of the L_2 norm of the rows of $\mathbf{W}$. Since L_1 norm would make the output of the parameter vector sparse [10], some rows of $\mathbf{W}$ will be zero. As a result, the linear classifier is less complex and only relies on a subset of the elements in $\mathbf{x}$. Group norm regularization was applied for SISL learning problems such as multi-task learning and logistic regression [11, 12].

2.4. Motivating example

Recall that our goal is to improve the accuracy in instance annotation in the MIML setting by using regularized kernel learning. To assess the potential improvement in this approach, we first test the improvement obtained using the kernel approach in the standard SISL setting on MIML datasets in which instance labels are provided. We would like to estimate the potential performance gain using kernel versions of SVM and logistic regression (LR). Accuracy for non-regularized linear kernel, regularized linear kernel, and regularized RBF kernel for SVM and LR are shown in Table 1.

Indeed, the results seem promising. We observe a significant improvement in the accuracy using SVM with RBF kernel over linear kernel. For the regularized linear kernel SVM, there is a suitable value of regularization parameter which makes the algorithm work best for each dataset, especially for the HJA bird dataset, where the improvement of regularization is significant. In addition, we also observe a significant improvement in the accuracy using LR with RBF kernel compared to that of linear kernel.

Table 1: Accuracy results (percentage) for non-regularized linear kernel, regularized linear kernel, regularized RBF kernel of SVM and LR denoted by S-L-SVM, S-R-L-SVM, S-R-RBF-SVM, S-LR, S-R-L-LR, and S-R-RBF-LR respectively, on real and synthetic datasets.

Dataset	HJA bird	MSCV2	Carroll	Frost
S-L-SVM	48.7	60.8	72.4	71.0
S-R-L-SVM	76.4	63.3	76.7	75.1
S-R-RBF-SVM	79.5	68.6	81.2	81.9
S-LR	72.0	60.5	68.0	70.6
S-R-L-LR	70.7	61.3	68.4	70.9
S-R-RBF-LR	76.7	66.6	79.3	80.0

Motivated by the aforementioned results for the SISL setting, we would like to develop a framework for training kernel-based instance level classifiers for instance annotation in the MIML setting. In this case, the number of effective instances may be small and hence regularization may play a more significant role. We proceed with the development of the kernel-based OR-ed Logistic Regression model and inference.

3. KERNEL-BASED OR-LR

In this section, we present a kernel extension to the OR-LR model and inference algorithm. In kernel learning, each feature vector $\mathbf{x}$ is represented by a kernel vector $\mathbf{k}(\mathbf{x}) = [K(\mathbf{x}, \mathbf{x}_1), K(\mathbf{x}, \mathbf{x}_2), \ldots, K(\mathbf{x}, \mathbf{x}_N)]$, where K may be for example, RBF or linear kernel mentioned in Section 2.2. Consequently, the probability of label $\mathbf{y}_{bi}$ given its instance $\mathbf{x}_{bi}$ can be rewritten as

$$p(\mathbf{y}_{bi}|\mathbf{x}_{bi}, \mathbf{W}) = \frac{\prod_{c=1}^C e^{I(\mathbf{y}_{bi}=c)\mathbf{w}_c^T \mathbf{k}(\mathbf{x}_{bi})}}{\sum_{c=1}^C e^{\mathbf{w}_c^T \mathbf{k}(\mathbf{x}_{bi})}}. \quad (11)$$

Note that each feature vector $\mathbf{x}_{bi}$ in the log-likelihood in (3) can be replaced with the corresponding kernel $\mathbf{k}(\mathbf{x}_{bi})$. As a result, the dimension of each parameter vector $\mathbf{w}_c$ in (11) is $N = \sum_{b=1}^B n_b$ which is equal to the total number of instances.

3.1. Sparsity regularized MLE

In this section, an $L_{2,1}$ norm regularization method is proposed for the log-likelihood (3). Recall that the dot product $\mathbf{w}_c^T \mathbf{k}(\mathbf{x}_{bi})$ in (11) can be expressed as follows

$$\mathbf{w}_c^T \mathbf{k}(\mathbf{x}_{bi}) = \mathbf{w}_{c1} K(\mathbf{x}_{bi}, \mathbf{x}_1) + \cdots + \mathbf{w}_{cN} K(\mathbf{x}_{bi}, \mathbf{x}_N). \quad (12)$$

From (12), if the jth row of $\mathbf{W}$ is the zero row vector, the term $\mathbf{w}_{cj} K(\mathbf{x}_{bi}, \mathbf{x}_j)$ will vanish for all c. That removes the effect of the jth instance in every class kernel-based score function. Regularization is necessary for kernel learning, since the dimension of the transformation space is equal to the number of data points in the dataset which may be high. Comparing to the hinge loss function in SVM, logistic regression does not encourage sparsity [13], therefore, regularization is much more important in the case of kernel-based logistic regression. Since we want to maximize the log-likelihood and minimize the norm, we subtract the norm $L_{2,1}$ weighted by parameter λ from the log-likelihood as follows

$$L_R(\mathbf{Y}_D|\mathbf{X}_D, \mathbf{W}) = \log p(\mathbf{Y}_D|\mathbf{X}_D, \mathbf{W}) - \lambda \|\mathbf{W}\|_{2,1}. \quad (13)$$

The log-likelihood is the same as in (3), where $p(\mathbf{y}_{bi}|\mathbf{x}_{bi}, \mathbf{W})$ is now as in (11). The optimal value $\mathbf{W}$ maximizing $L_R(\mathbf{Y}_D|\mathbf{X}_D, \mathbf{W})$ compromises between the sparsity requirement and the log-likelihood information from the data.

3.2. Inference: expectation maximization

Computing the surrogate function for (13) requires taking the expectation over its complete regularized log-likelihood, which can be expressed as

$$L_R(\mathbf{Y}_D, \mathbf{y}|\mathbf{X}_D, \mathbf{W}) = \log p(\mathbf{Y}_D, \mathbf{y}|\mathbf{X}_D, \mathbf{W}) - \lambda\|\mathbf{W}\|_{2,1}. \quad (14)$$

The second term in (14) is a constant when taking expectation over $\mathbf{y}$. As a result, the regularized surrogate function can be computed as follows

$$\begin{aligned} g(\mathbf{W}, \mathbf{W}') &= E_{\mathbf{y}}[\log p(\mathbf{Y}_D, \mathbf{y}|\mathbf{X}_D, \mathbf{W})|\mathbf{Y}_D, \mathbf{X}_D, \mathbf{W}'] - \lambda\|\mathbf{W}\|_{2,1} \\ &= \sum_{b=1}^{B}\sum_{i=1}^{n_b}[\sum_{c=1}^{C} p(\mathbf{y}_{bi} = c|\mathbf{Y}_b, \mathbf{X}_b, \mathbf{W}')\mathbf{w}_c^T\mathbf{x}_{bi} \\ &\quad - \log(\sum_{c=1}^{C} e^{\mathbf{w}_c^T\mathbf{x}_{bi}})] - \lambda\|\mathbf{W}\|_{2,1} + \zeta, \end{aligned} \quad (15)$$

where $p(\mathbf{y}_{bi} = c|\mathbf{Y}_b, \mathbf{X}_b, \mathbf{W}')$ is as in (11). Similar to the OR-LR on the feature space, we alternate between updating $p(\mathbf{y}_{bi} = c|\mathbf{Y}_b, \mathbf{X}_b, \mathbf{W}')$ and maximizing $g(\mathbf{W}, \mathbf{W}')$ following the generalized EM framework [14].

Expectation step: We apply the dynamic programming scheme for the expectation step as in Section 2.1.3. The only difference in the computation of posterior probability in (5), the vector $\mathbf{x}_{bi}$ is replaced with $\mathbf{k}(\mathbf{x}_{bi})$.

Maximization step: The maximization step is performed using a gradient ascent approach. In calculating the gradient, both the log-likelihood and the regularization term are considered. Consider $\mathbf{e}_t \in \mathbb{R}^N$ such that $\mathbf{e}_t(i) = 1$ iff $t = i$ and $\mathbf{e}_t(i) = 0$ otherwise.

Lemma 1. *The derivative of the $L_{2,1}$ norm of matrix $\boldsymbol{W}$ is given by* $\frac{\partial\|\boldsymbol{W}\|_{2,1}}{\partial vec(\boldsymbol{W})} = vec(\sum_{t=1}^{N}\frac{\boldsymbol{e}_t\boldsymbol{e}_t^T\boldsymbol{W}}{\|\boldsymbol{e}_t^T\boldsymbol{W}\|_2})$.

The proof can be found in the Appendix. From Lemma 1 and (15) we have

$$\begin{aligned} \frac{\partial g(\mathbf{W}, \mathbf{W}^{(k)})}{\partial \mathbf{w}_c} &= \sum_{b=1}^{B}\sum_{i=1}^{n_b}[p(\mathbf{y}_{bi} = c|\mathbf{Y}_b, \mathbf{X}_b, \mathbf{W}^{(k)})\mathbf{k}(\mathbf{x}_{bi}) \\ &\quad - \frac{e^{\mathbf{w}_c^T\mathbf{k}(\mathbf{x}_{bi})}\mathbf{k}(\mathbf{x}_{bi})}{\sum_{t=1}^{C} e^{\mathbf{w}_t^T\mathbf{k}(\mathbf{x}_{bi})}}] - \lambda\sum_{t=1}^{N}\frac{\mathbf{e}_t\mathbf{e}_t^T\mathbf{w}_c}{\|\mathbf{e}_t^T\mathbf{W}\|_2} \\ = \sum_{b=1}^{B}\sum_{i=1}^{n_b}&[p(\mathbf{y}_{bi} = c|\mathbf{Y}_b, \mathbf{X}_b, \mathbf{W}^{(k)}) - p(\mathbf{y}_{bi} = c|\mathbf{X}_b, \mathbf{W})]\mathbf{k}(\mathbf{x}_{bi}) \\ &\quad - \lambda\sum_{t=1}^{N}\frac{\mathbf{e}_t\mathbf{e}_t^T\mathbf{w}_c}{\|\mathbf{e}_t^T\mathbf{W}\|_2}. \end{aligned}$$

Finally, using gradient ascent, we obtain the update rule for the cth column of $\mathbf{W}$ as follows

$$\begin{aligned} \mathbf{w}_c^{(k+1)} &= \mathbf{w}_c^{(k)} + \eta\frac{\partial g(\mathbf{W}, \mathbf{W}^{(k)})}{\partial \mathbf{w}_c}\Big|_{\mathbf{W}=\mathbf{W}^{(k)}} \\ &= \mathbf{w}_c^{(k)} + \eta\Big(\sum_{b=1}^{B}\sum_{i=1}^{n_b}[p(\mathbf{y}_{bi} = c|\mathbf{Y}_b, \mathbf{X}_b, \mathbf{W}^{(k)}) \\ &\quad - p(\mathbf{y}_{bi} = c|\mathbf{X}_b, \mathbf{W}^{(k)})]\mathbf{k}(\mathbf{x}_{bi}) - \lambda\sum_{t=1}^{N}\frac{\mathbf{e}_t\mathbf{e}_t^T\mathbf{w}_c^{(k)}}{\|\mathbf{e}_t^T\mathbf{W}^{(k)}\|_2}\Big), \end{aligned}$$

where η is selected using the backtracking line search [15].

3.3. Instance label prediction

After learning the parameter $\mathbf{W}$, we can predict the label of instance $\mathbf{y}_{ti}$ given $\mathbf{x}_{ti}$. We use the inductive prediction framework [4], in which the test bag label is unknown and the instance label $\mathbf{y}_{ti}$ is predicted as follows

$$\hat{\mathbf{y}}_{ti} = \operatorname*{argmax}_c p(\mathbf{y}_{ti} = c|\mathbf{x}_{ti}, \mathbf{W}). \quad (16)$$

Since $p(\mathbf{y}_{ti} = c|\mathbf{x}_{ti}, \mathbf{W})$ in (1) has the same denominator for all c, (16) is equivalent to

$$\hat{\mathbf{y}}_{ti} = \operatorname*{argmax}_c \mathbf{w}_c^T\mathbf{k}(\mathbf{x}_{ti}). \quad (17)$$

After obtaining the label for each instance, the label of the bag can be predicted as the union of its instance labels.

4. NUMERICAL EVALUATION

In this section, we study the performance improvements achieved using the kernel extension of OR-LR. Moreover, we compare the proposed algorithm with an instance-annotation version of a state-of-the-art MIML algorithm, namely, MIMLfast (Mfast for short) [3]. In the training phase, Mfast learns a parameter $\mathbf{W}$ which can be used to compute the score $f_c(\mathbf{x}_{bi})$ w.r.t. each label c on each instance $\mathbf{x}_{bi}$. However, Mfast does not predict the label $\hat{\mathbf{y}}_{bi}$ for each instance $\mathbf{x}_{bi}$. It indirectly uses $f_c(\mathbf{x}_{bi})$ for all i to compute the score of the bth bag w.r.t. each label c. In our experiment, we access $f_c(\mathbf{x}_{bi})$ and output the predicted label $\hat{\mathbf{y}}_{bi}$ as $\hat{\mathbf{y}}_{bi} = \operatorname{argmax}_c f_c(\mathbf{x}_{bi})$. We also present a performance comparison on bag label prediction between the proposed kernel OR-LR and Mfast.

4.1. Experimental setup

We use two real datasets HJA bird song and MSCV2 image annotation and two synthetic datasets Letter Carroll and Letter Frost [4]. Detailed information on these datasets is given in Table 2. For the HJA bird song, we use the filtered version, in which unlabeled instances are removed [4]. Note that in the filtered HJA [4], there are 67 bags whose labels are not equal to the union of their instance labels. We replace the label of each bag with the union of its instance labels. For the Letter Carroll, even though it is reported of having 26 classes, there is no instance in the 15th and 20th classes. We fix that by adding two synthetic instances whose feature vectors in $\mathbb{R}^{16}$ are $[10, 10, \ldots, 10]$ and $[5, 5, \ldots, 5]$ for these two classes respectively into the first bag. Similarly, we fix the Letter Frost with the 17th and 20th classes.

Table 2: Statistics of datasets in our experiments.

Name	classes *(C)*	bags *(B)*	instances *(N)*	dimension *(d)*
HJA bird	13	548	4998	38
MSCV2	23	591	1758	48
Carroll	26	166	717	16
Frost	26	144	565	16

4.2. Experimental results

Table 3: Accuracy results (percentage) for instance label prediction in inductive mode M-LR, M-RBF–LR, M-R-RBF-LR, M-Mfast, M-L-Mfast, M-RBF-Mfast, and SIM [4]. The proposed algorithms and optimal performances are highlighted.

Dataset	HJA bird	MSCV2	Carroll	Frost
M-LR	70.1±4.0	55.7±5.0	62.4±4.7	64.5±6.5
M-RBF-LR	**73.1±3.8**	**61.8±4.2**	**71.3±6.4**	**72.2±5.2**
M-R-RBF-LR	**73.4±3.8**	**61.9±4.7**	**71.6±6.3**	**72.6±4.6**
M-Mfast	58.7±2.8	45.7±6.2	52.7±6.7	52.6±7.2
M-L-Mfast	59.9±3.5	49.7±5.3	56.9±5.2	59.1±5.6
M-RBF-Mfast	63.7±4.2	55.8±4.2	66.2±4.4	67.9±5.8
SIM	62.7±4.1	45.6±4.2	53.4±6.1	57.1±5.9

Table 4: Hamming loss (percentage) for bag label prediction in inductive mode M-LR, M-RBF–LR, M-R-RBF-LR, M-Mfast, M-L-Mfast, M-RLK-Mfast, and SIM. The proposed algorithms and optimal performances are highlighted.

Dataset	HJA bird	MSCV2	Carroll	Frost
M-LR	11.4±1.2	7.6±1.0	9.0±0.7	8.1±1.2
M-RBF-LR	9.9±1.0	**6.6±0.9**	**6.9±1.8**	**6.7±1.5**
M-R-RBF-LR	9.8±1.1	**6.7±1.0**	**7.0±1.7**	**6.6±1.4**
M-Mfast	6.4±0.9	12.0±1.3	15.8±2.1	13.9±1.4
M-L-Mfast	5.8±0.7	9.2±0.9	12.9±1.6	12.1±1.8
M-RBF-Mfast	**4.8±0.5**	8.2±1.0	11.9±1.8	11.0±2.1
SIM	16.2±2.0	9.0±0.7	11.0±1.1	10.3±1.6

First, we run instance label prediction using OR-LR and Mfast in feature learning, linear kernel, and RBF kernel denoted by M-LR, M-RBF-LR, M-R-RBF-LR, M-Mfast, M-L-Mfast, M-RBF-Mfast, respectively. Additionally, we run instance label prediction using SIM in feature learning. For the RBF kernel, we first find the mean squared-distances between every pair of instances $\overline{d^2}$. Then, we choose the RBF kernel parameter in Section 2.2, $\delta = s/\overline{d^2}$, where s is selected in $\{10^{-3}, 2\cdot 10^{-3}, 5\cdot 10^{-3}, 10^{-2}, 2\cdot 10^{-2}, 5\cdot 10^{-2}, \ldots, 10, 20, 50\}$. The value of λ is selected in $\{10^{-5}, 2\cdot 10^{-5}, 5\cdot 10^{-5}, 10^{-4}, 2\cdot 10^{-4}, 5\cdot 10^{-4}, \ldots, 10, 20, 50\}$. Since the dimension of instance vectors in the kernel-based OR-LR framework is higher than in the original OR-LR framework, we run 500 EM iterations to observe the objective function in (14) converges instead of 50 iterations in the original OR-LR. Since running MIML OR-LR for the entire grid of (λ, δ) is time consuming, we first observe the optimal values of $(\lambda_{sisl}, \delta_{sisl})$ in the SISL version. Then, we run the MIML version over that value of δ_{sisl} and two of its nearest values in the set, to find an optimal δ. Then, for that δ, we run the MIML version over the optimal value of λ_{sisl} and two of its nearest values in the set to find an optimal λ for the MIML version. We normalize the data so that the mean and standard deviation of each feature are zero and one, respectively. Since Mfast is sensitive to the norm of the data (as suggested by Theorem 1 in [3, p. 8]), we renormalize the data by replacing $\mathbf{x}_{bi}$ with $\alpha\mathbf{x}_{bi}/\sqrt{d}$ for the feature learning case and $\alpha\mathbf{x}_{bi}/\sqrt{N}$ for the kernel learning case. We then consider α a tuning parameter $\in \{2\cdot 10^{-2}, 5\cdot 10^{-2}, \ldots, 20, 50, 100\}$. For the feature and linear kernel cases, we report the performance at the optimal α. For the RBF kernel case, we report the performance at the optimal (α, δ). We report the prediction accuracy for all algorithms considered in Table 3. Since Mfast is designed for bag label prediction, when obtaining the instance label prediction accuracy for each algorithm, we also report the Hamming loss for bag label in Table 4. Specifically, let $\mathbf{Y}_t$ and $\widehat{\mathbf{Y}}_t$ denote the ground truth and predicted labels for the tth test bag, respectively, where $t \in \{1, 2, \ldots, T\}$. The Hamming loss d_H between them [16] is as follows

$$d_H = \frac{\sum_{t=1}^{T}\sum_{c=1}^{C}\left(I(c\in\mathbf{Y}_t) - I(c\in\widehat{\mathbf{Y}}_t)\right)^2}{T\cdot C},$$

where $I(\cdot)$ is the indicator function, taking the value 1 when its argument is true and 0 otherwise.

From the M-LR, M-Mfast, and SIM rows of Table 3, we observe that OR-LR [5] outperforms SIM which outperforms Mfast in *instance level prediction* in the feature learning case. As explained in [5], OR-LR uses a probability setting to accurately account for all instances while Mfast and SIM use a max or softmax principle, which may ignore some of the training instances. Comparing the performance of kernel learning versions of OR-LR and Mfast, we observe the kernel version of OR-LR outperforms the kernel version of Mfast on the four datasets.

In terms of bag level classification, even though OR-LR is designed for instance level classification, the Hamming loss of OR-LR is lower than that of Mfast on three out of the four datasets.

4.3. Accuracy vs. kernel width

To examine the effect of RBF kernel for both OR-LR and MFast, we report the accuracy of both algorithms against δ as defined in Section 4.2. The results on Letter Carroll and Letter Frost are shown in Fig. 2. Both algorithms obtain the highest accuracy at a nonzero kernel parameter value δ, suggesting the importance of considering the kernel extension to both algorithms.

5. CONCLUSION

In this paper, we address the problem of instance annotation where the data are linearly inseparable. We propose a kernel-based extension with sparsity regularization to the recent state-of-the-art OR-ed logistic regression framework for instance annotation. The proposed kernel-based OR-LR yields a significant improvement, from 3% to 6% for real bird song and image annotation datasets, and from 8% to 9% for two synthetic datasets. The experiments also show that the performance of our proposed algorithm surpasses that of the kernel version of a recent state-of-the-art algorithm Mfast, both on bag and instance label prediction. Our current focus is on reducing the runtime of the iterative implementation of the proposed method.

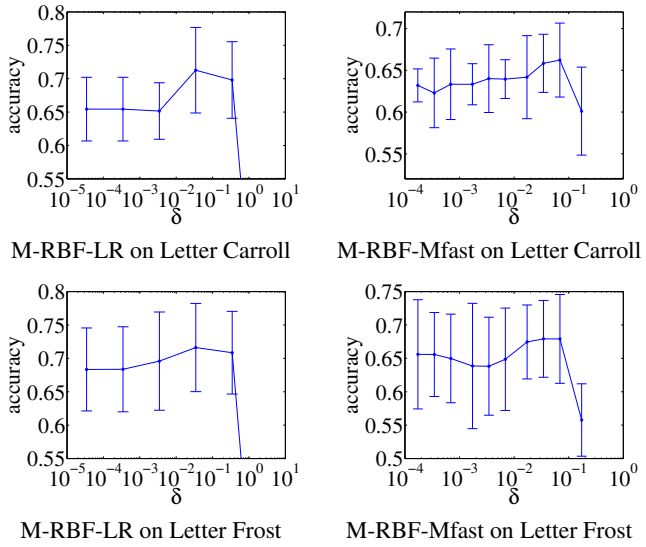

Fig. 2: Accuracy results for M-RBF-LR and M-RBF-Mfast as a function of δ on the Letter Carroll and Letter Frost datasets.

APPENDIX

Lemma 1. *The derivative of the $L_{2,1}$ norm of matrix $\mathbf{W}$ is given by* $\frac{\partial \|\boldsymbol{W}\|_{2,1}}{\partial vec(\boldsymbol{W})} = vec(\sum_{t=1}^{N} \frac{\boldsymbol{e}_t \boldsymbol{e}_t^T \boldsymbol{W}}{\|\boldsymbol{e}_t^T \boldsymbol{W}\|_2})$.

Proof. The $L_{2,1}$ norm of the matrix $\mathbf{W}$ is defined as

$$\|\mathbf{W}\|_{2,1} = \|\mathbf{e}_1^T\mathbf{W}\|_2 + \|\mathbf{e}_2^T\mathbf{W}\|_2 + \cdots + \|\mathbf{e}_N^T\mathbf{W}\|_2. \tag{18}$$

Let $\mathbf{I}$ be a $C \times C$ identity matrix and define a matrix $\mathbf{A}_t = \mathbf{I} \otimes \mathbf{e}_t$, where $\otimes$ is the Kronecker product and $\mathbf{e}_t$ is as in Section 3.2. Then, (18) can be rewritten as

$$\|\mathbf{W}\|_{2,1} = \|\mathbf{A}_1^T vec(\mathbf{W})\|_2 + \|\mathbf{A}_2^T vec(\mathbf{W})\|_2 + \cdots + \|\mathbf{A}_N^T vec(\mathbf{W})\|_2. \tag{19}$$

The gradient of the tth term on the RHS of (19) can be written as

$$\begin{aligned} \frac{\partial \|\mathbf{A}_t^T vec(\mathbf{W})\|_2}{\partial vec(\mathbf{W})} &= \frac{\partial \|\mathbf{A}_t^T vec(\mathbf{W})\|_2^2}{2\sqrt{\|\mathbf{A}_t^T vec(\mathbf{W})\|_2^2}\partial vec(\mathbf{W})} \\ &= \frac{\mathbf{A}_t\mathbf{A}_t^T vec(\mathbf{W})}{\|\mathbf{A}_t^T vec(\mathbf{W})\|_2} = \frac{(\mathbf{I} \otimes \mathbf{e}_t\mathbf{e}_t^T) vec(\mathbf{W})}{\|\mathbf{e}_t^T\mathbf{W}\|_2}. \end{aligned} \tag{20}$$

Note that the last equation in (20) comes from the fact that

$$\mathbf{A}_t\mathbf{A}_t^T = (\mathbf{I} \otimes \mathbf{e}_t)(\mathbf{I} \otimes \mathbf{e}_t)^T = \mathbf{I} \otimes \mathbf{e}_t\mathbf{e}_t^T.$$

Consequently, we have

$$\frac{\partial \|\mathbf{A}_t^T vec(\mathbf{W})\|_2}{\partial vec(\mathbf{W})} = \frac{(\mathbf{I} \otimes \mathbf{e}_t\mathbf{e}_t^T) vec(\mathbf{W})}{\|\mathbf{e}_t^T\mathbf{W}\|_2} = \frac{vec(\mathbf{e}_t\mathbf{e}_t^T\mathbf{W})}{\|\mathbf{e}_t^T\mathbf{W}\|_2}. \tag{21}$$

Note that the last equation in (21) comes from the property of the Kronecker product in which $(\mathbf{A} \otimes \mathbf{B})vec(\mathbf{C}) = vec(\mathbf{BCA}^T)$. Summing (21) over all t from 1 to N, we have

$$\sum_{t=1}^{N} \frac{\partial \|\mathbf{A}_t^T vec(\mathbf{W})\|_2}{\partial vec(\mathbf{W})} = vec(\sum_{t=1}^{N} \frac{\mathbf{e}_t\mathbf{e}_t^T\mathbf{W}}{\|\mathbf{e}_t^T\mathbf{W}\|_2}). \tag{22}$$

□

A. REFERENCES

[1] Y.-F. Li, J.-H. Hu, Y. Jiang, and Z.-H. Zhou, "Towards discovering what patterns trigger what labels," in *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[2] Z.-H. Zhou and M.-L. Zhang, "Multi-instance multi-label learning with application to scene classification," *Advances in Neural Information Processing Systems*, vol. 19, pp. 1609, 2007.

[3] S.-J. Huang and Z.-H. Zhou, "Fast multi-instance multi-label learning," *arXiv preprint arXiv:1310.2049*, 2013.

[4] F. Briggs, X. Z. Fern, and R. Raich, "Rank-loss support instance machines for MIML instance annotation," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2012, KDD '12, pp. 534–542, ACM.

[5] A. T. Pham, R. Raich, and X. Z. Fern, "Efficient instance annotation in multi-instance learning," in *Proceedings of the IEEE Workshop on Statistical Signal Processing*, 2014, SSP '14, accepted.

[6] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 551–556.

[7] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[8] K. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *Neural Networks, IEEE Transactions on*, vol. 12, no. 2, pp. 181–201, 2001.

[9] A. Erkan and Y. Altun, "Semi-supervised learning via generalized maximum entropy," in *International Conference on Artificial Intelligence and Statistics*, 2010, pp. 209–216.

[10] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *The Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

[11] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," in *Advances in Neural Information Processing Systems*, 2007, pp. 41–48.

[12] L. Meier, S. Van De Geer, and P. Bühlmann, "The group lasso for logistic regression," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 70, no. 1, pp. 53–71, 2008.

[13] L. Zhang, R. Jin, C. Chen, J. Bu, and X. He, "Efficient online learning for large-scale sparse kernel logistic regression.," in *AAAI*, 2012.

[14] G. McLachlan and T. Krishnan, *The EM algorithm and extensions*, vol. 382, John Wiley & Sons, 2007.

[15] S. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge university press, 2004.

[16] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *In Advances in Neural Information Processing Systems 14*, 2001.