Collaborative multilabel classification *

Yunzhang Zhu[1], Xiaotong Shen[2], Hui Jiang[3], and Wing Hung Wong[4]

Abstract

In multilabel classification, strong label dependence is present for exploiting, particularly for word-to-word dependence defined by semantic labels. In such a situation, we develop a collaborative-learning framework to predict class labels based on label-predictor pairs and label-only data. For example, in image categorization and recognition, language expressions describe the content of an image together with a large number of words and phrases without associated images. This article proposes a new loss quantifying partial correctness for false positive and negative misclassifications due to label similarities. Given this loss, we develop the Bayes rule to capture label dependence by nonlinear classification. On this ground, we introduce a weighted random forest classifier for complete data and a stacking scheme for leveraging additional labels to enhance the performance of supervised learning based on label-predictor pairs. Importantly, we decompose multilabel classification into a sequence of independent learning tasks, based on which the computational complexity of our classifier becomes linear in the size of labels. Compared to existing classifiers without label-only data, the proposed classifier enjoys the computational benefit while enabling the detection of novel labels absent from training by exploring label dependence and leveraging label-only data for higher accuracy. Theoretically, we show that the proposed method reconstructs the Bayes performance consistently, achieving the desired learning accuracy. Numerically, we demonstrate that the proposed method compares favorably in terms of the proposed and Hamming losses against binary relevance and a regularized Ising classifier modeling conditional label dependence. Indeed, leveraging additional labels tends to improve the supervised performance, especially when the training sample is not very large, as in semisupervised learning. Finally, we demonstrate the utility of the proposed approach on the Microsoft COCO object detection challenge, PASCAL visual object classes challenge 2007, and Mediamill benchmark.

Keywords: Binary relevance, label dependence, multimodality, novel labels, nonconvex minimization, scalability.

*[1] Department of Statistics, The Ohio State University, Columbus, OH 43210; [2]School of Statistics, University of Minnesota, Minneapolis, MN 55455, [3] Department of Biostatistics, University of Michigan, MI, 48109; [4] Department of Statistics and Biomedical Data Science, Stanford University, CA 94305. Research supported in part by NSF grants DMS-1712564, DMS-1721216, DMS-1712580, DMS-1721445, DMS-1811920, DMS-1952539, DMS-2015490, and NIH grants 1R01GM126002,2R01HL105397,1R01AG069895. The authors would like to thank the action editor and two anonymous referees for helpful comments and suggestions.

1 Introduction

In multilabel classification, semantic labels such as words and phrases present strong label dependence, characterized by word-to-word dependence. Such dependence can be unconditional or conditional depending on predictors. For example, "football" and "tennis" are two semantic labels that are unconditionally dependent, usually co-occurring in a sports event, and conditionally independent given the equipment used in sports. In such a situation, classification based on inadequate training examples often yields a poor performance given the complex structure of semantic labels in the presence of a large number of additional labels without predictors. For instance, in image-captioning [12], a learner trains image-caption pairs to describe the contents of an image, together with a large number of captions from news articles in newspapers, which the learner can leverage to account for label dependence for enhancing prediction. One central issue is how to leverage additional labels while accounting for label dependence to enhance supervised learning based on label-predictor pairs. Also, do label-only data provide the relevant information for classification in a similar manner as unlabeled data in semisupervised learning [29]? In this article, we develop a classification framework, what we call collaborative multilabel classification, under which we develop classifiers to leverage label dependence and additional labels to deliver higher predictive accuracy than its supervised counterpart ignoring either additional labels or label dependence.

Given a large body of literature on multilabel classification focusing on non-semantic labels, we focus our discussion on the most relevant references. Ideally, one may perform independent binary classifications, one for each label, known as binary relevance (BR). The state-of-art binary relevance learning [10, 6, 20, 5] is an ensemble of binary classifications by treating predicted values as predictors for one additional classification. However, binary relevance largely ignores label dependence. As illustrated in [5], component-wise dependen-

cies are crucial to prediction in multi-response regression, indicating that a classification model must model label dependence To account for unconditional dependence, the frequencies of label co-occurrences are used in training examples [23]. To account for conditional label dependence, Markov/Bayesian/dependence networks are employed for estimation of the dependence structure and classification see [15] for an excellent survey. For example, [14] constructs a classifier chain utilizing the frequencies of label co-occurrences in training examples to account for unconditional dependence, yet identifying a good chain is indeed challenging. Given the mounting cost of estimation of the dependence structure, an effort of modeling label dependency has not been rewarded in the past. Recently, [8] utilizes a sparse pseudo-likelihood for an Ising model describing linear conditional dependence, which is a frequentist version of [15]. Moreover, [17] proposed a compressed sensing method by exploring output sparsity. On a related topic, [30] suggested a tree-based approach for hierarchical labels. Despite progress, issues remain, particularly for modeling semantic labels and leveraging additional labels without predictors, as in image captioning [12].

The contributions of this paper are five-fold. First, we develop a multilabel framework of collaborative classifiers for label-predictor pairs as well as additional labels, with a focus concentrating on semantic labels. One salient feature of the proposed classifier is its capability of detection of a novel class absent from training by exploring conditional label dependence. Moreover, the classifier can improve the supervised performance by leveraging additional labels, particularly when the training sample size is not large, as in semisupervised learning [29]. Second, we develop nonlinear classifiers based on random forests [4] to capture label dependence, conditionally and unconditionally, integrating additional labels through stacking [26], where the use of nonlinear classifiers is critical to learn conditional label dependence. Third, we introduce a weighted loss quantifying partial correctness for false positive and negative misclassifications due to label similarities, which is in contrast to the Hamming loss [27], the Jaccard distance [13], and the subset loss or the 0-1 loss [13]. In the literature,

a hierarchical loss imposes partial correctness through the size of offspring of a node when each label corresponds to a node in a hierarchy such as a tree [7], which dramatically differs from our situation. Moreover, existing classifiers are typically designed under the Hamming loss, or 0-1 loss, which does not target the Bayes rule under the proposed loss, and hence that they tend to perform worse when the proposed loss is used to evaluate, as demonstrated by our simulations. Fourth, we develop a computational method that decomposes the joint learning task into the independent learning of transformed labels, which dramatically reduces the computation cost. As a result, the proposed method is more scalable than its competitors in the memory requirement. In simulations, we demonstrate that the proposed classifier outperforms binary relevance and a regularized pseudo-likelihood classifier under two evaluation loss functions, namely, the proposed loss (3) and the Hamming loss; see Tables 1 and 2. Moreover, for three benchmark examples, it continues to outperform its competitors, including a ResNet50 Convolution Neural Networks (CNN) deep learner [16]. Fifth, we establish consistent recovery of the Bayes performance by the proposed classifier in terms of the evaluation loss function and give conditions for the proposed classifier to detect novel labels.

This article is organized as follows. Section 2 presents the proposed loss function, Section 3 formulates a collaborative learning framework, develops a classier to account for label dependence, and integrates additional labels through stacking. Computationally, we develop a decomposable learning strategy to allow the computational complexity of our classifier to be linear in the label size. Section 4 investigates the theoretical properties of the proposed classifier. Section 5 examines their numerical performances and compares them with some strong competitors, namely, the approaches binary relevance and an Ising model classifier through simulations, followed by an application to three benchmark examples. Finally, the appendix contains technical proofs.

2 Collaborative classification

In multilabel classification, $\Big(\boldsymbol{X} = (X_1, \cdots, X_q)^\top, \boldsymbol{Y} = (Y_1, \cdots, Y_p)^\top\Big)$ is an input-output pair, where labels $Y_1, \cdots, Y_p$ present strong label dependence given $\boldsymbol{X}$, with each label Y_j is coded as $\{-1, 1\}$; $j = 1, \cdots, p$. In addition, an independent sample of additional label observations $\{\boldsymbol{Z}_j\}_{j=n+1}^{n+m}$ following the same distribution of $\{\boldsymbol{Y}_i\}_{i=1}^n$ are available, typically a small amount of complete data with a large amount of additional labels in that the size of additional labels m may greatly exceed the sample size n. This framework is what we call collaborative multilabel classification. Our primary objective is to (1) leverage additional labels, (2) label dependence for higher accuracy of classification, as well as (3) detection of novel labels.

2.1 A novel loss

In collaborative multilabel classification, we introduce a new loss to quantify partial correctness for false positive and negative misclassifications due to label similarities.

The accuracy of classification is measured by the generalization error $\mathrm{Err}(\boldsymbol{f}) = \mathbb{E}L(\boldsymbol{Y}, \boldsymbol{f})$, where $L(\cdot, \cdot)$ is a loss function measuring discrepancy between observed $\boldsymbol{Y}$ and its prediction by $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \cdots, f_p(\boldsymbol{x}))^\top$, which is a decision function vector with $f_k(\boldsymbol{x})$ predicting Y_k, and $\mathbb{E}$ is the expectation with respect to $(\boldsymbol{X}, \boldsymbol{Y})$. Now we propose a nonnegative loss to quantify false positive and negative errors:

$$\begin{aligned} L(\boldsymbol{y}, \boldsymbol{f}) \;=\; & \sum_{1 \le l,k \le p} \big(\mathbb{I}(y_l = 1) w_{-lk} + \mathbb{I}(y_l = -1) w_{+lk}\big) \mathbb{I}(y_l f_k(\boldsymbol{x}) < 0) \\ & - \sum_{k=1}^{p} \min\Big(\sum_{l: y_l = +1} w_{-lk} \,, \sum_{l: y_l = -1} w_{+lk} \Big), \end{aligned} \tag{1}$$

where $w_{+lk} \ge 0$ and $w_{-lk} \ge 0$ are the amounts of penalty for false positive and negative errors made by $f_k(\boldsymbol{x})$ for predicting label y_l, and $\mathbb{I}$ is the indicator function. For false negatives,

$w_{-lk} > 0$ when labels y_l and y_k are semantically similar, $w_{-kk} \geq w_{-kl}$; $l \neq k$ when correct classification of y_k by $f_k(\boldsymbol{x})$ is more important than that by $f_k(\boldsymbol{x})$ for $l \neq k$. For false positives, w_{+kl} is usually small and can be $w_{+kl} = 0$ without any loss; $k \neq l$, particularly when the primary objective is to identify the presence of certain labels as in image object detection. Note that the second term in (1) ensures that $L(\boldsymbol{y}, \boldsymbol{f}) = 0$ corresponds to no error because of (3) as false positive and negative errors can not occur simultaneously. For convenience, we may normalize the rows or columns of $\boldsymbol{W}_{\pm}$ so that the row or column sum equals 1.

In contrast, the Hamming (symmetric difference) loss [27], the Jaccard distance [13], and the subset loss or the 0-1 loss [13] neither discriminate the false positive and negative errors nor permit partial label correctness. Note that (1) reduces to the Hamming loss if $w_{\pm lk} = 0$ when $l \neq k$. As illustration, consider a simple case of three semantic labels "football", "basketball", and "vehicle", where $w_{+ll} = 1/3$ and $w_{+lk} = 0$ otherwise; $1 \leq l, k \leq 3$, and $w_{-11} = w_{-12} = w_{-21} = w_{-22} = 1/6$, $w_{-33} = 1/3$ and $w_{-lk} = 0$ otherwise. In this case, we are concerned about the false positive error, and hence that misclassifying "football" to "basketball" incurs a penalty of 1/3, which is smaller than misclassifying "football" to "vehicle" with a loss of 1/2, according to the degree of similarity. Hence, loss (1) appears more sensible than the aforementioned loss functions that yield neither partial correctness nor discriminate false negative and positive errors.

In practice, $\boldsymbol{W}_- = \{w_{-kl}\}_{\{1 \leq k,l \leq p\}}$ may be estimated semantic similarity measure based on an independent sample. Whereas $\boldsymbol{W}_+ = p^{-1}\boldsymbol{I}$ is sensible, particularly for our target application of object recognition, $\boldsymbol{W}_-$ is estimated by global vectors for word representation (GLOVE) [21]. Vector-space representations of this kind encode the semantic information of words as numerical vectors in the Euclidean space. They are constructed so that semantically similar words have word vectors close to each other, making it an ideal tool to measure word-to-word similarities. In particular, we propose to use the Cosine similarities between the word

vector representations of two vectors to measure their similarity. Formally, denote by $\boldsymbol{v}_j$ the word vector representation of label j. We compute a similarity measure, for any two labels j and k, as

$$\exp\left(\frac{\boldsymbol{v}_j^\top \boldsymbol{v}_k}{\|\boldsymbol{v}_j\|_2 \|\boldsymbol{v}_k\|_2}\right), \tag{2}$$

and normalize the rows of $\boldsymbol{W}_-$ so that the row sum equals to 1. For the applications considered in this paper, all labels have pre-trained GLOVE word vector representations available. For other applications involving word labels beyond GLOVE, one could fine-tune the pre-trained GLOVE models over the new corpus.

Loss (1) is non-negative and seems non-decomposable at first blush, which is unlike the Hamming loss $L_H(\boldsymbol{y}, \boldsymbol{f})$ [27] in that $L_H(\boldsymbol{y}, \boldsymbol{f}) = \frac{1}{p}\sum_{k=1}^p \mathbb{I}(y_k \neq f_k(\boldsymbol{x}))$ can be written as a sum of individual label loss functions. Surprisingly, we can decompose $L(\boldsymbol{y}, \boldsymbol{f})$ in (1) as follows

$$\begin{aligned} L(\boldsymbol{y}, \boldsymbol{f}) \;&= \textstyle\sum_{k=1}^p L_k(\boldsymbol{y}, f_k)\,, L_k(\boldsymbol{y}, f_k) = |\delta_k(\boldsymbol{y})|\mathbb{I}(\delta_k(\boldsymbol{y}) f_k(\boldsymbol{x}) < 0), \\ &\delta_k(\boldsymbol{y}) = \textstyle\sum_{l:y_l=+1} w_{-lk} - \sum_{l:y_l=-1} w_{+lk}; k = 1, \ldots, p\,, \end{aligned} \tag{3}$$

where we have used

$$\begin{aligned} L(\boldsymbol{y}, \boldsymbol{f}) \;=\; & \sum_{k=1}^p \Big(\big(\sum_{l:y_l=1} w_{-lk}\big)\mathbb{I}(f_k(\boldsymbol{x}) < 0) + \big(\sum_{l:y_l=-1} w_{+lk}\big)\mathbb{I}(f_k(\boldsymbol{x}) \geq 0) \Big) \\ & - \sum_{k=1}^p \min\big(\sum_{l:y_l=+1} w_{-lk}\,, \sum_{l:y_l=-1} w_{+lk}\big) = \sum_{k=1}^p L_k(\boldsymbol{y}, f_k) \end{aligned}$$

and the fact that $\mathbb{I}(f_k(\boldsymbol{x}) < 0) + \mathbb{I}(f_k(\boldsymbol{x}) \geq 0) = 1$.

The decomposition (3) has several consequences. First, it decomposes the overall generalization error with respect to each label classification: $\mathrm{Err}(\boldsymbol{f}) = \sum_{k=1}^p \mathrm{Err}_k(f_k)$, where

$\mathrm{Err}_k(f_k) = \mathbb{E}L_k(\boldsymbol{Y}, f_k)$ is the error for misclassifying label k. Second, $L_k(\boldsymbol{y}, f_k)$ is highly interpretable in that $\delta_k(\boldsymbol{y})$ is the aggregated misclassification error over false positives and negatives, which is determined by w_{+lk} and w_{-lk}. Moreover, it is a margin loss that is a function of functional margin $\delta_k(\boldsymbol{Y})f_k(\boldsymbol{X})$ for predicting outcome of $\mathrm{Sign}(\delta_k(\boldsymbol{Y}))$ by $\mathrm{Sign}(f_k(\boldsymbol{X}))$. Note, however, that even if $\mathrm{Sign}(\delta_k(\boldsymbol{Y})) = Y_k$ for all $1 \leq k \leq p$, minimization under the new loss is not equivalent to that under the Hamming loss. This is because weights $\delta_k(\boldsymbol{Y})$ may not necessarily equal to each other for $1 \leq k \leq p$.

3 Methods

3.1 Multilabel classification and label dependence

To predict the outcome of $\boldsymbol{Y}$ given $\boldsymbol{X}$, we derive the Bayes rule under loss L in (3), based on which our predication rule is constructed. Specifically, Lemma 1 gives the Bayes decision function $\boldsymbol{f}^\star$ that minimizes the generalization error $\mathrm{Err}(\boldsymbol{f}) = \mathrm{Err}_k(f_k)$ over $\{\boldsymbol{f} : \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \cdots, f_p(\boldsymbol{x}))^\top\}$.

Lemma 1 *(Bayes decision rule) Given loss L in* (1)*, the Bayes decision rule is expressed as*

$$\mathrm{Sign}(f_k^\star(\boldsymbol{x})) = \mathrm{Sign}\left(\sum_{l=1}^{p}\left(w_{-lk}\mathbb{P}(Y_l = 1 \mid \boldsymbol{x}) - w_{+lk}\mathbb{P}(Y_l = -1 \mid \boldsymbol{x})\right)\right), \tag{4}$$

where $f_k^\star(\boldsymbol{x}) = \mathbb{E}(\delta_k(\boldsymbol{Y}) \mid \boldsymbol{x})$ is the Bayes decision function.

In (4), $f_k^\star$ is a sum of weighted conditional probabilities, which reduces to the case of Hamming loss in which $f_k^\star(\boldsymbol{x}) = 2P(Y_k = 1|\boldsymbol{x}) - 1$ when $W_{\pm lj} = 0$; $l \neq j$. However, the Bayes rule in (4) typically differs from that under the Hamming loss provided that not all $W_{\pm lj} = 0$; $l \neq j$. This means that a classifier constructed under the Hamming loss is not Fisher-consistent or fail to converge to the Bayes rule, which is in contrast to the proposed classifier that is Fisher-consistent, c.f., Theorem 2.

Lemma 1 suggests that label dependence needs to be leveraged to deliver a good performance of a classifier under loss L accounting for label dependence. Based on (4), we propose our prediction rule as $Y_k = \text{Sign}(f_k^\star(\boldsymbol{X}))$, where $f_k^\star(\boldsymbol{x})$ estimates $\mathbb{E}(\delta_k(\boldsymbol{Y}) \mid \boldsymbol{x})$; $k = 1, \cdots, p$.

Given training data $(\boldsymbol{Y}^i, \boldsymbol{X}^i)_{i=1}^n$ and additional labels $(\boldsymbol{Z}^i)_{i=n+1}^{n+m}$, we construct a cost function to learn $\boldsymbol{f}$, where $\boldsymbol{Y}^i = (Y_1^i, \ldots, Y_p^i)^\top$, $\boldsymbol{Z}^i = (Z_1^i, \ldots, Z_p^i)^\top$, and $\boldsymbol{X}^i = (X_1^i, \ldots, X_p^i)^\top$. First, we propose a cost function based on complete data $(\boldsymbol{Y}^i, \boldsymbol{X}^i)_{i=1}^n$ and (3), after ignoring the constant $\sum_{k=1}^p \min\big(\sum_{l:y_l=+1} w_{-lk}\,, \sum_{l:y_l=-1} w_{+lk}\big)$:

$$S(\boldsymbol{f}) = \sum_{k=1}^{p} S_k(f_k); \quad S_k(f_k) \equiv \sum_{i=1}^{n} |\delta_k(\boldsymbol{Y}^i)| \mathbb{I}(\delta_k(\boldsymbol{Y}^i) f_k(\boldsymbol{X}^i) < 0). \tag{5}$$

Now minimizing $S(\boldsymbol{f})$ in $\boldsymbol{f}$ is equivalent to solving p independent optimizations, that is, for $k = 1, \cdots, p$,

$$\min_{f_k \in \mathcal{F}_k} S_k(f_k) = \min_{\boldsymbol{f}_k \in \mathcal{F}_k} (\sum_{i=1}^{n} |\delta_k(\boldsymbol{Y}^i)| \mathbb{I}(\delta_k(\boldsymbol{Y}^i) f_k(\boldsymbol{X}^i) < 0), \tag{6}$$

reducing the complexity of parameter estimation from $O(p^2)$ [19] to $O(p)$ in dimension p , where $\mathcal{F}_k$ is a class of candidate decision functions for f_k. In (6), we solve weighted binary classification for predicting $\text{Sign}(\delta_k(\boldsymbol{Y}))$, where the overall misclassification loss is weighted by $|\delta_k(\boldsymbol{Y}^i)|$ over p labels $Y_1^i, \cdots, Y_p^i$. Note that the solution of (6) is not unique and the indicator function there is replaced by a large margin surrogate to resolve this issue [9].

To estimate f_k in (6), we employ a weighted version of random forest [4] due to its parsimonious tree representations with the capability of variable selection, permitting treatment of nonparametric classification with many variables. Random forest uses trees and bootstrap aggregation or bagging. Bagging repeats B times to select a bootstrap sample or a random sample with replacement of the training data and fits classification trees to these samples. In particular, for $b = 1, \cdots, B$, a classification tree $\widehat{f}_{kb}$ is trained on each boot-

strap sample. Then the label of unseen $\boldsymbol{x}$ is predicted by averaging the predictions from all the individual classification trees on $\boldsymbol{x}$, that is, the predicted label $\widehat{Y}_k = \text{Sign}(\widehat{f}_k(\boldsymbol{x}))$, where $\widehat{f}_k(\boldsymbol{x}) = B^{-1}\sum_{b=1}^{B} f_{kb}(\boldsymbol{x})$. Note that the random forest suffers from a lesser degree of the curse of dimensionality with respect to the dimension of $\boldsymbol{x}$ because a classification tree involves a lower degree of piecewise constants.

3.2 Prediction based on additional labels

This section focuses on multilabel classification based on additional labels $(\boldsymbol{Z}^i)_{i=n+1}^{n+m}$, which is integrated with that based on complete data for label prediction through stacking in Section 3.3.

In the absence of input $\boldsymbol{x}$, to predict label values of $\boldsymbol{Z}$, we introduce a decision function vector $\boldsymbol{g} = (g_1, \cdots, g_p)^\top$, one for each label, which serves as baseline functions without $\boldsymbol{x}$. Then we propose a loss $L(\boldsymbol{z}, \boldsymbol{g}) = \sum_{k=1}^{p} L_k(\boldsymbol{z}, g_k) + \sum_{k=1}^{p} \min\left(\sum_{l:z_l=+1} w_{-lk}\,, \sum_{l:z_l=-1} w_{+lk}\right)$ based on (3), where $z = (z_1, \cdots, z_p)^\top$. This in turn yields our proposed cost function: for $k = 1, \cdots, p$,

$$\sum_{i=n+1}^{n+m} L_k(\boldsymbol{Z}^i, g_k) = \sum_{i=n+1}^{n+m} |\delta_k(\boldsymbol{Z}^i)| \mathbb{I}(\delta_k(\boldsymbol{Z}^i) g_k < 0). \tag{7}$$

Since we can write the RHS of (7) as

$$\begin{aligned} & \mathbb{I}(g_k < 0) \sum_{i:n+1\le i\le n+m, \delta_k(\boldsymbol{Z}^i)>0} \delta_k(\boldsymbol{Z}^i) - \mathbb{I}(g_k > 0) \sum_{i:n+1\le i\le n+m, \delta_k(\boldsymbol{Z}^i)<0} \delta_k(\boldsymbol{Z}^i) \\ = \; & \mathbb{I}(g_k < 0) \sum_{i=n+1}^{n+m} \delta_k(\boldsymbol{Z}^i) - \sum_{i:n+1\le i\le n+m, \delta_k(\boldsymbol{Z}^i)<0} \delta_k(\boldsymbol{Z}^i)\,, \end{aligned}$$

minimization of (7) in g_k yields an estimated decision function $\widehat{g}_k = \frac{\sum_{i=n+1}^{n+m} \delta_k(\boldsymbol{Z}^i)}{\sum_{i=n+1}^{n+m} |\delta_k(\boldsymbol{Z}^i)|}$. Interestingly, $\widehat{g}_k$ is the normalized value of $\delta_k(\cdot)$ aggregated over the additional observed labels. Note that the solution of (7) is not unique.

We use additional labels and feature-only data to learn the weight matrices W_+ and W_-. In particular, we consider minimizing the following objective with respect to the discriminant function and both weight matrices jointly

$$\min_{\boldsymbol{f},W_+,W_-} \sum_{k=1}^{p}\sum_{i=1}^{n} L_k(\boldsymbol{y}_i, f_k) + \lambda \left(\sum_{k=1}^{p} \ell(\mathbb{E}_1 f_k(X), \mathbb{E}_2 y_k) + \sum_{k'<k} \ell(\mathbb{E}_1 f_k(X) f_{k'}(X), \mathbb{E}_2 y_k y_{k'}) \right) \quad (8)$$

where $\mathbb{E}_1$ and $\mathbb{E}_2$ are empirical expectations over feature-only data and additional labels, and

$$\ell(p_1, p_2) = -p_1 \log(p_2) - (1 - p_1) \log(1 - p_2)\,. \quad (9)$$

Note that the joint minimization of $\boldsymbol{f}$ and weight matrices may be difficult if we use the random forest model for $\boldsymbol{f}$. This is because the random forest software can not handle the loss function in the second term (i.e., $l(\cdot,\cdot)$). We can potentially consider the following greedy approach.

We sample $(W_+^{(m)}, W_-^{(m)})$; $m = 1, \ldots, M$, and compute $\hat{f}_k^{(m)} = \arg\min_{f_k} \sum_{i=1}^{n} L_k(\boldsymbol{y}_i, f_k)$; $k = 1, \ldots, p$. Then using the sampled weight matrices $(W_+^{(m)}, W_-^{(m)})$, we compute a score

$$S_m = \ell(\mathbb{E}_1 \hat{f}_k^{(m)}(X), \mathbb{E}_2 y_k) + \sum_{k'<k} \ell(\mathbb{E}_1 \hat{f}_k^{(m)}(X) \hat{f}_{k'}^{(m)}(X), \mathbb{E}_2 y_k y_{k'})\,. \quad (10)$$

Finally, we choose $(W_+^{(m)}, W_-^{(m)})$ with the smallest S_m. The idea is that we treat the weight matrices as tuning parameters (as opposed to model parameters).

3.3 Integration of complete and additional labels

This section derives a stacking or an ensemble method, to further enhance the accuracy of prediction based on complete data by incorporating the prediction from additional labels. Specifically, we combine two prediction functions in a form $\widehat{h}_k(\alpha, \boldsymbol{x}) = \alpha \widehat{f}_k(\boldsymbol{x}) + (1 - \alpha)\widehat{g}_k$, where $\alpha \in [0, 1]$ is a tuning parameter for model averaging. Our objective is to tune α to

ensure that predictive accuracy of h_k is no less than that based on complete data alone.

Given cross-validation data $(\tilde{\boldsymbol{Y}}^i, \tilde{\boldsymbol{X}}^i)_{i=1}^N$, we minimize the predictive loss:

$$T(\alpha) \equiv \frac{1}{N} \sum_{k=1}^{p} \sum_{i=1}^{N} |\delta_k(\tilde{\boldsymbol{Y}}^i)| \mathbb{I}(\delta_k(\tilde{\boldsymbol{Y}}^i) \widehat{h}_k(\alpha, \tilde{\boldsymbol{X}}^i) < 0); \quad k = 1, \cdots, p, \tag{11}$$

with respect to uniform grids over $[0,1]$, that is, $\alpha \in \left\{0, \frac{1}{c_N}, \frac{2}{c_N}, \ldots, \frac{c_N - 1}{c_N}, 1\right\}$. The minimizer $\widehat{\alpha}$ leads to our final prediction: $\widehat{Y}_k = \text{Sign}(\widehat{h}_k(\widehat{\alpha}_k, \boldsymbol{X}))$.

The next lemma says that the combined classifier performs no worse than that complete data alone asymptotically.

Lemma 2 *Assume that $\sum_{k=1}^p \sum_{l=1}^p (w_{+lk} + w_{-lk})$ is upper bounded by a constant, and $E[T(\alpha)]$ is L-Lipschitz continuous in α at $\alpha^\star$, where $\alpha^\star = \text{argmin}_{\alpha \in [0,1]} E[T(\alpha)]$. Moreover, suppose that*

$$v(\alpha) := Var(T(\alpha) - T(\alpha^\star)) \leq a \{E(T(\alpha) - T(\alpha^\star))\}^\gamma, \tag{12}$$

for some constants $a, \gamma \geq 0$. Suppose that $c_N = O(N^{\min(1/(2-\gamma),1)})$. Then for the stacked classifier $\widehat{\boldsymbol{h}}(\boldsymbol{x}) = (\widehat{h}_1(\boldsymbol{x}), \ldots, \widehat{h}_p(\boldsymbol{x})$ with $\widehat{h}_k(\boldsymbol{x}) = \widehat{h}_k(\widehat{\alpha}_k, \boldsymbol{x})); k = 1, \ldots, p$,

$$\text{Err}(\widehat{\boldsymbol{h}}) \leq \text{Err}(\widehat{\boldsymbol{h}}(\alpha^\star)) + O_p(N^{-\min(\frac{1}{2-\gamma}, 1)}). \tag{13}$$

As a technical remark, (12) is a commonly used smoothness assumption in the classification literature, see, for example, [24], which is implied by the low-noise condition (margin assumption) [28].

Usually, $\text{Err}(\widehat{\boldsymbol{h}}(\alpha^\star))$ does not converge faster than $O_p(n^{-1})$. Hence Lemma 2 implies that if $N \approx n$ then $\text{Err}(\widehat{\boldsymbol{h}}) \leq (1 + o(1)) \text{Err}(\widehat{\boldsymbol{h}}(\alpha^\star))$.

3.4 Nonlinear learning and label dependence

This section presents a result arguing that the pairwise label dependence is taken into account through nonlinear learning regardless of such dependence is expressed in terms of linear conditional dependence. To see this aspect, we examine pairwise conditional dependence of Y_j and Y_k given $\boldsymbol{X}$ in an Ising model.

Lemma 3 *Suppose that* $\boldsymbol{Y} = (Y_1, \cdots, Y_p)^\top$ *follows a conditional Ising model given a predictor vector* $\boldsymbol{X}$*. The probability density of* Y *given* $\boldsymbol{X} = \boldsymbol{x}$ *is*

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{\alpha}(\boldsymbol{x}))} \exp\left(\frac{1}{2}\sum_{j=1}^{p} \alpha_{jj}(\boldsymbol{x}) y_j + \frac{1}{2}\sum_{k>j} \alpha_{jk}(\boldsymbol{x}) y_j y_k\right), \tag{14}$$

where $\boldsymbol{y} = (y_1, \cdots, y_p)^\top$, $\boldsymbol{\alpha}(\boldsymbol{x}) = (\alpha_{11}(\boldsymbol{x}), \alpha_{12}(\boldsymbol{x}), \cdots, \alpha_{pp}(\boldsymbol{x}))$ *is a* $p(p+1)/2$*-dimensional vector and* $Z(\boldsymbol{\alpha}(\boldsymbol{x}))$ *is the partition function. Then, for* $k = 1, \cdots, p$, *logit*$(\mathbb{P}(Y_k = 1 \mid \boldsymbol{x}))$ *is written as*

$$\alpha_{kk}(\boldsymbol{x}) + \log \frac{\sum_{\boldsymbol{y}_{-k}} \exp\left(\frac{1}{2}\sum_{j\neq k}(\alpha_{jj}(\boldsymbol{x}) + \alpha_{jk}(\boldsymbol{x}))y_j + \frac{1}{2}\sum_{j<j'} \alpha_{jj'}(\boldsymbol{x}) y_j y_{j'}\right)}{\sum_{\boldsymbol{y}_{-k}} \exp\left(\frac{1}{2}\sum_{j\neq k}(\alpha_{jj}(\boldsymbol{x}) - \alpha_{jk}(\boldsymbol{x}))y_j + \frac{1}{2}\sum_{j<j'} \alpha_{jj'}(\boldsymbol{x}) y_j y_{j'}\right)} \equiv g_k(\boldsymbol{\alpha}(\boldsymbol{x})). \tag{15}$$

As indicated by Lemma 3, the conditional covariance of Y_j and Y_k given $\boldsymbol{X} = \boldsymbol{x}$, which is proportional to $\alpha_{jk}(\boldsymbol{x})$, enters into a classification model (15) nonlinearly even if $\alpha_{jk}(\boldsymbol{x})$ is linear in $\boldsymbol{x}$. This observation, together with the result of Lemma 1, suggests that our nonlinear representation of $f_k(\boldsymbol{x})$ in (6) is more appropriate than its linear counterpart for accounting for label dependence in any situation.

4 Learning theory

4.1 Detection of novel labels

When a label is absent from a training set, a classifier typically does not assign any instances to that label, failing to detect novel labels. In contrast, our classifier, as defined in (6), has the capability of detecting a novel class absent from training through label dependence. Theorem 1 gives such a result.

Theorem 1 *(Detection of novel labels) Suppose that the k-th label is absent from the training data, that is,* $Y_k^i = -1$*;* $i = 1, \cdots, n$*. If there exists an* i *with* $1 \leq i \leq n + m$ *such that* $\delta_k(\boldsymbol{Y}^i) > 0$ *then there exists* α^* *with* $0 \leq \alpha^* \leq 1$ *such that* $\widehat{h}_k(\alpha^*, \boldsymbol{x}^i) > 0$ *for some* $1 \leq i \leq n$*, or class* k *can be detected.*

4.2 Consistent recovery of the Bayes performance

The original random forests [4] rely on complex data-dependent mechanisms of selecting variables and cutting points, which makes it extremely difficult to analyze. As a result, its basic statistical properties remain not fully understood. To the best of our knowledge, most existing theoretical results focus on a simplified version of the original random forest algorithm so that statistical analysis is more tractable; see [3] for a comprehensive review of recent theoretical developments. As such, we expand existing results to our new loss function based on a simplified version of random forests [1].

More specifically, we analyze a simple random forest in (6), which is a voting classifier of simple decision trees considered in [2, 1]. For a single tree, a coordinate of $\boldsymbol{X} = (X_1, \ldots, X_q)$ is chosen at each node, and the j-th feature having probability $p_{nj} \in (0, 1)$ of being selected, and the selected cell is split along with the randomly chosen variable at the midpoint of the chosen side.

A voting classifier $\widehat{f}_k^{(B)}$ for the k-th label is defined as the average of B independent tree classifiers

$$\widehat{f}_k^{(B)}(\boldsymbol{x}) = \frac{1}{B}\sum_{j=1}^{B} \widehat{f}_{kb}(\boldsymbol{x}), k = 1, \ldots, p, \tag{16}$$

where $\widehat{f}_{kb}(\boldsymbol{x}); b = 1, \ldots, B$ are independent single trees with the same number of variable splits. Next we generalize consistency results of [2, 1] under the Hamming loss to the loss L in (3). Specifically, consistency of the voting classifier $\widehat{\boldsymbol{f}}^{(B)}$ means that $\mathrm{Err}(\widehat{\boldsymbol{f}}^{(B)}) = \sum_{k=1}^{p} \mathbb{E}\{|\delta_k(\boldsymbol{Y})|\mathbb{I}(\widehat{f}_k^{(B)}(\boldsymbol{X}) \neq \mathrm{Sign}(\delta_k(\boldsymbol{Y}))\}$ converges to $L^\star = \sum_{k=1}^{n} \mathbb{E}\{|\delta_k(\boldsymbol{Y})|\mathbb{I}(f_k^\star(\boldsymbol{X}) \neq \mathrm{Sign}(\delta_k(\boldsymbol{Y}))\}$, where $f_k^\star$ is a Bayes classifier.

The next theorem establishes conditions under which our method (6) is consistent under (1).

Theorem 2 *Assume that the distribution of $\boldsymbol{X}$ is supported on $[0,1]^q$. Moreover, we assume that $f_k^\star(\boldsymbol{x}) = \mathbb{E}(\delta_k(\boldsymbol{Y}) \mid X = \boldsymbol{x})$; $k = 1, \cdots, p$, are uniformly L-Lipschitz continuous:*

$$\max_{1\leq k\leq p} |f_k^\star(x_1) - f_k^\star(x_2)| \leq L\|x_1 - x_2\|_2, \tag{17}$$

where $L > 0$ is a constant independent of p, q, and n. Let S denote the number of splits for each individual tree. Then, the voting random forest classifier $\widehat{\boldsymbol{f}}^{(B)}$ is consistent if, $S \to \infty$, $\frac{p^3 S}{n} \to 0$, and $\min_{1\leq j\leq q} p_{nj} \log S - 2\log p \to \infty$ as $S \to \infty$.

It is worth mentioning that any classifier that is not Fisher-consistent is not consistent in the sense of Theorem 2. For example, a classier constructed under the Hamming loss is inconsistent because the label dependence is ignored; see the discussion after Lemma 1.

If $(p_{n1}, \ldots, p_{nq})$ is chosen to split variables at random for each tree, or $(p_{n1}, \ldots, p_{nq}) = (1/q, \ldots, 1/q)$, then the scaling condition requires that $q = o\left(\frac{\log n}{\log p}\right)$, which means that the feature dimension can only grow more slowly than $\frac{\log n}{\log p}$.

To handle a high-dimensional situation, the probabilities $(p_{n1}, \ldots, p_{nq})$ need to be chosen

adaptively and a true classification model is sparse. For example, as heuristically discussed in [1], if the target decision function $\mathbb{E}(\delta_k(\boldsymbol{Y}) \mid \boldsymbol{X})$ depends only on $q_0 = |\mathcal{S}_0|$ features, which is a subset $\mathcal{S}_0$ of the q features, then p_{nj} can be chosen adaptively using an independent validation set when so that the feature dimension q can grow much faster with n.

5 Numerical examples

This section investigates several aspects of the proposed method, namely, (1) the operating characteristics of the classifier, (2) the contribution of additional labels on the accuracy of prediction, and (3) the capability of detection of partially observed labels. Importantly, we compare the proposed classifier with four state-of-art classifiers, including two binary relevance based on SVM [9] classifications, the unweighted random forest classifier [4], and a Pseudo-Ising classifier [8] based on an Ising model. In three benchmark examples, we also include a ResNet50 Convolution Neural Networks (CNN) deep learner [16] for a comparison. The three binary relevance classifiers are denoted by linear-SVM, nonlinear-SVM, and CW, which are separate linear and Gaussian-kernel SVM classifiers, one for each label, and CW is a refitting classifier using the fitted values from binary relevance based on linear SVM [4].

To investigate the impact of modeling label dependence, we use an Ising model (14) as in [8], for linear pairwise conditional dependence by setting $\alpha_{jk}(\boldsymbol{x}) = \alpha_{kj}(\boldsymbol{x}) = \boldsymbol{x}^\top \theta_{jk} + \theta_{jk0}$ in (14), $k < j$. In particular, we generate a q-dimensional feature vector $\boldsymbol{x}$ independently from a standard Normal distribution $N(0, 1)$. Given $\boldsymbol{x}$, we recursively generate $n + m$ responses from an Ising model with the conditional density following (14), where θ_{jkl}; $l = 0, \ldots, p$ are drawn from Unif$(-\gamma_1, \gamma_1)$; $j \neq k$, and θ_{jjl}; $l = 0, \ldots, p$ are drawn from Unif$(-\gamma_2, \gamma_2)$; $j = 1, \ldots, q$, where diffident values of γ_1 and γ_2 will be examined: $\gamma_1 = \gamma_2 = 50$ in Table 1–4 and $\gamma_1 = 50$ and $\gamma_2 = .1$ in Table 5. Note that the signal strength level reduces when γ_2 decreases.

The accuracy of classification is evaluated by the test error based on a test sample of 2000 under loss L in (3) as well as the Hamming loss, approximating the generalization error $\mathrm{Err}(\boldsymbol{f})$ for $\boldsymbol{f}$, where two weight matrices defining (3) are

$$\boldsymbol{W}_+ = I_{p\times p} \text{ and } W_- = \begin{pmatrix} .6 & \frac{.4}{p-1} & \cdots & \frac{.4}{p-1} \\ \frac{.4}{p-1} & .6 & \cdots & \frac{.4}{p-1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{.4}{p-1} & \frac{.4}{p-1} & \cdots & .6 \end{pmatrix}, \tag{18}$$

where rows/columns of the weight matrices have sum equal to 1.

Numerical analysis is performed in R. For our training data, we first generate $m + n$ paired observations, with $n = 200, 500$ and $m = 2000$. Furthermore, we consider $p = 2, 10, 50, 100, 500, 1000$ and $q = 20, 50$. Finally, we generate a test set with 2000 independent complete observations. Then the test error is computed under the loss (1) over the test data. For the proposed method integrating with additional labels, five-fold cross-validation is used for selection of the tuning parameter α over a set of 1000 uniform grid points of $[0, 1]$.

As shown in Table 1, the proposed method outperforms all the competitors in terms of the test error across all the settings under our evaluation loss (3). The amounts of improvement over LSVM, NSVM, CW, and P-Ising range from 42.5% to 67.8%, from 14.9% to 61.8%, from 23.9% to 56.6%, and 13.1% to 155.4%, respectively. Roughly, large improvements occur for challenging situations, particularly when either p or q increases while n is held fixed. Interestingly, the Pseudo-Ising classifier does not perform as well as expected even although the data are generated from the Ising model. This is mainly because it uses a linear classification to account for linear label dependence. As indicated in Lemma 3, linear dependence is only captured by a nonlinear method. Moreover, a pseudo-likelihood approach is less efficient when it does not target the evacuation loss. Interestingly, the proposed method continues to fare well even under other commonly used loss functions. For instance, as

indicated in Table 2, the proposed method performs well under the Hamming loss, although the amount of improvement over the Pseudo-Ising classier shrinks as p increases. Moreover, as shown in Table 3, the aforementioned results extend to the F_1-score with higher F_1-scores than other competing methods across all situations, where $F_1 = \left(1 + \frac{FN+FP}{2TP}\right)^{-1}$. In summary, the proposed method performs well even when the classification loss differs from the evaluation loss in this case, which is attributed to the fact that label dependence has been adequately taken into account in the model.

Concerning the contribution of additional labels, Table 5 suggests that our method does improve the classification performance of its counterpart with complete data alone consistently across all the settings. However, the amount of improvement becomes large for difficult situations. The improvement is attributed primarily to the utilization of additional labels through stacking.

Concerning runtime, the proposed method runs faster than other nonlinear methods for more difficult set-ups and is slower than linear SVM, as suggested in Table 4. Importantly, the proposed method runs linearly in the number of labels, while enabling to account for label dependence. This is in contrast to binary relevance methods ignoring label dependence.

Finally, to investigate the capability of detecting novel labels for the proposed method, we consider the same simulation setup as before with $n = 200, 500$, $p = 2, 10, 50, 100, 500, 1000$, and $q = 4$. In particular, we set the first label to be -1 in all cases in the training data. As suggested by Table 6, the TPR of the proposed method is strictly positive as long as the dimension p is not that large, but it decays as p increases because the level of difficulty escalates. This occurs in 4 out of 12 cases. This indicates that the proposed method is capable of detecting the novel label if the noise level is not too large. This corroborates with the result of Theorem 1. By comparison, all competitors are not capable of detecting novel labels with TPR $= 0$ and FPR $= 0$.

Tables 1-6 about here

6 Benchmarks

This section demonstrates the utility of the proposed method and also compares with some state-of-art methods for multilabel classification on two benchmark examples.

(A) Microsoft COCO object detection challenge

We first examine the 2017 COCO Object Detection Task data in the Microsoft COCO object detection challenge dataset [18], consisting of about $118,287$ images with 80 object categories and about $500,000$ disparate objects. In each image, each object is bounded by a box, which is associated with one label, with multiple bounding boxes corresponding to multiple labels. The training, validation, and testing sets are available. Moreover, all images are labeled, so we do not consider additional labels in the full analysis. As labels for the test set is unreleased, we will use the training set for learning and the validation set for testing instead.

To apply a classifier, we extract image features from these images by first applying a ResNet50 Convolution Neural Networks (CNN) deep learner [16]. Specifically, we extract the last layer of a trained ResNet50 model applying to images in our training and test sets. This gives a feature dimension of 2056 for each image from the final layer of the ResNet50 CNN model. Moreover, we also include a binary relevance CNN for comparison, that is, a ResNet50 CNN classifier [16] performing binary classification for each label, which is implemented using a Python deep learning library–Keras. To expedite training, we initialize all the weights except for the last layer using a pre-trained version of ResNet50.

For the whole data set of $118,287$ images with $p = 80$ and $q = 2056$, we run our method on a machine with 2 Twelve Core Xeon E5-2690 v3 2.6GHz CPUs and 128GB memory. For the weight matrices, we set $\boldsymbol{W}_+ = p^{-1}\boldsymbol{I}$ to be the 80×80 identity matrix and $\boldsymbol{W}_-$ is computed

using (2) based on label-label similarity matrix computed by GLOVE word vectors [21]. As suggested by Table 8, the proposed method yields an overall test errors of .002 under loss (1) and .025 under the Hamming loss over 80 categories as measured by $\mathrm{Err}(\cdot,\cdot)$ based on a test set of size 5000, supplied in the COCO data. However, the other competitors require much higher memory and are unable to run there. By comparison, the proposed method is scalable to this data without a large memory requirement.

To compare with LSVM, NSVM, Pseudo-Ising, and the ResNet50 CNN classifier, we decrease the size of the original problem by randomly subsampling 6000 images on six categories: "person", "car", "motorcycle", "bus", "truck", "train". The average numbers of positively labeled observations for the six categories are 3252, 621, 178, 200, 311, and 182. For the weight matrices, $\boldsymbol{W}_-$ becomes a 6×6 matrix, as defined in Table 7, while $\boldsymbol{W}_+$ is the 6×6 identity matrix. To be more informative, we report separate test errors for each category $\mathrm{Err}_k(\hat{f}_k)$; $k = 1, \cdots, 6$, as well as the overall error $\mathrm{Err}(\hat{\boldsymbol{f}})$, which are obtained by averaging over 100 random subsamples, where $\hat{\boldsymbol{f}} = (\hat{f}_1, \cdots, \hat{f}_6)^\top$ is a classifier.

Tables 7-11 about here

As suggested in Table 8, the proposed method outperforms the competitors substantially under loss L in (3), with the smallest test error .004 over six classes, which is in contrast to the corresponding test error .002 for the full data over 80 classes. An increased training sample size does improve the accuracy of classification. Moreover, the amounts of improvement over LSVM, NSVM, and Pseudo-Ising are 825%, 150%, and 6775%. Interestingly, the proposed method continues to perform the best in the largest two classes "Person" and "Car", and performs well for the other four small classes. However, the proposed method underperforms NSVM under the Hamming loss, although it outperforms other methods for this reduced data. This is because the classification and evaluation losses differ.

To study the impact of additional labels on classification, we take a random sample of 500 images with at least two objects within the six categories: "person", "car", "motorcycle", "bus", "truck", "train", and another sample of 100 images containing at most one in these categories. The rest of the training samples are now treated as additional labels. As shown in Table 9, the proposed method leveraging paired data and additional labels significantly outperforms its counterpart without leveraging additional labels. Note that the improvement of five competing methods without leveraging additional labels ranges from 280% to 575%. This suggests that additional labels indeed provide valuable information for prediction. This is an analogy of the situation in semisupervised learning in which a semisupervised classifier that leverages unlabeled data usually leads to an improvement over its supervised counterpart [29]. In a sense, the proposed method borrows external information from additional labels to enhance the performance of supervised learning as in transfer learning [22].

(B) PASCAL visual object classes challenge 2007

Now, we analyze a different image dataset from the PASCAL Visual Object Classes Challenge 2007 [11], consisting of training and validation sets of 5011 images and a testing set of 4952 images. Now we apply all the methods on the training and validation sets for training and use the testing set for evaluation. The weight matrix for the proposed loss is generated as in (A) using the label-label similarity matrix computed by GLOVE word vectors.

As suggested by Table 10, the proposed method continues to outperform its competitors under the proposed weighted loss. The amounts of improvement over LSVM, NSVM, Pseudo-Ising, and CNN are 200%, 16.67%, 5300%, and 66.67%. Moreover, the proposed method is the second better in both the Hamming loss and the F_1-score, that is, it is slightly worse than NSVM under the Hamming loss but better than NSVM under the F_1-score, and is slightly worse than P-Ising under the F_1-score but better than P-Ising under the Hamming loss. Overall, the proposed method fares well across three evaluation criteria.

(C) Mediamill benchmark

Lastly, we analyze a Mediamill dataset, which extracted from 85 hours of international news videos from the TRECVID 2005/2006 benchmark datasets [25] and is publicly available at `https://ivi.fnwi.uva.nl/isis/mediamill/challenge/data.php`. This dataset includes 120 low-level visual and textual features with labels consisting of a lexicon of 101 semantic concepts, like commercials, nature, and baseball. The training set is comprised of 30993 samples while the testing set has 12914 samples. Again, the weight matrix for the proposed loss is generated as in (A) using the label-label similarity matrix computed by GLOVE word vectors. There are no additional labels for this dataset.

For our analysis, we use the given training and testing sets for training and evaluation. Moreover, we examine all aforementioned classifiers except CNN because the generated features are not suited for CNN.

As seen from Table 11, the proposed method again outperforms its competitor NSVM under the proposed weighted loss as well as the Hamming loss. The amount of improvement over NSVM is about 49%, 4%, and 6% under the proposed weighted loss, the Hamming loss, and the F_1-score. Note that LSVM and Pseudo-Ising can not complete after one week.

References

[1] G. Biau. Analysis of a random forests model. *Journal of Machine Learning Research*, 13(Apr):1063–1095, 2012.

[2] G. Biau, L. Devroye, and G. Lugosi. Consistency of random forests and other averaging classifiers. *Journal of Machine Learning Research*, 9(Sep):2015–2033, 2008.

[3] G. Biau and E. Scornet. A random forest guided tour. *TEST*, 25(2):197–227, Jun 2016.

[4] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] L. Breiman and J. Friedman. Predicting multivariate responses in multiple linear regression. *Journal of the Royal Statistical Society. Series B (Methodological)*, 59:3–54, 1997.

[6] V. Carey, S. L. Zeger, and P. Diggle. Modelling multivariate binary data with alternating logistic regressions. *Biometrika*, 80(3):517–526, 1993.

[7] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Incremental algorithms for hierarchical classification. *Journal of Machine Learning Research*, 7(Jan):31–54, 2006.

[8] J. Cheng, E. Levina, P. Wang, and J. Zhu. Sparse ising model with covariates. *Biometrics*, 70(4):943–953, 2014.

[9] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[10] D. R. Cox. The analysis of multivariate binary data. *Applied statistics*, pages 113–120, 1972.

[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[12] H. Fang, S. Gupta, F. Iandola, R. K. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J. C. Platt, et al. From captions to visual concepts and back. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1473–1482, 2015.

[13] V. Gjorgjioski, D. Kocev, and S. Džeroski. Comparison of distances for multi-label classification with pcts. In *Proceedings of the Slovenian KDD Conference on Data Mining and Data Warehouses (SiKDD'11)*, 2011.

[14] S. Godbole and S. Sarawagi. Discriminative methods for multi-labeled classification. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 22–30. Springer, 2004.

[15] Y. Guo and S. Gu. Multi-label classification using conditional dependency networks. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1300, 2011.

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[17] D. J. Hsu, S. M. Kakade, J. Langford, and T. Zhang. Multi-label prediction via compressed sensing. In *Advances in neural information processing systems*, pages 772–780, 2009.

[18] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[19] Y. Liu and X. Shen. Multicategory ψ-learning. *Journal of the American Statistical Association*, 101(474):500–509, 2006.

[20] J. F. Pendergast, S. J. Gange, M. A. Newton, M. J. Lindstrom, M. Palta, and M. R. Fisher. A survey of methods for analyzing clustered binary response data. *International Statistical Review/Revue Internationale de Statistique*, pages 89–118, 1996.

[21] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[22] L. Y. Pratt. Discriminability-based transfer between neural networks. In *Advances in Neural Information Processing Systems*, pages 204–211, 1993.

[23] J. Read, P. Reutemann, B. Pfahringer, and G. Holmes. Meka: a multi-label/multi-target extension to weka. *The Journal of Machine Learning Research*, 17(1):667–671, 2016.

[24] X. Shen, L. Wang, et al. Generalization error for multi-class margin classification. *Electronic Journal of Statistics*, 1:307–330, 2007.

[25] C. G. Snoek, M. Worring, J. C. Van Gemert, J.-M. Geusebroek, and A. W. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of the 14th ACM international conference on Multimedia*, pages 421–430, 2006.

[26] K. M. Ting and I. H. Witten. Stacking bagged and dagged models. *esearchcommons.waikato.ac.nz*, 1997.

[27] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on machine learning*, page 104. ACM, 2004.

[28] A. B. Tsybakov et al. Optimal aggregation of classifiers in statistical learning. *The Annals of Statistics*, 32(1):135–166, 2004.

[29] J. Wang and X. Shen. Large margin semi-supervised learning. *Journal of Machine Learning Research*, 8(Aug):1867–1891, 2007.

[30] Q. Wu, Y. Ye, H. Zhang, T. W. Chow, and S.-S. Ho. Ml-tree: A tree-structure-based approach to multilabel learning. *IEEE transactions on neural networks and learning systems*, 26(3):430–443, 2014.

Table 1: Test errors (standard errors in parentheses) of various methods under the loss L in (3) based on 100 simulations for complete data, where LSVM, NSVM, P-Ising, CW, and Our denote linear SVM, nonlinear SVM, Pseudo-Ising classifier [8], Breiman's classifier by combining binary relevance linear-SVMs, and the proposed method.

(n, p, q)	LSVM	NSVM	P-Ising	CW	Our
$(500, 2, 20)$	.101(.002)	.077(.002)	.146(.007)	.083(.003)	.067(.002)
$(500, 10, 20)$	.106(.002)	.11(.002)	.162(.005)	.093(.003)	.082(.002)
$(500, 50, 20)$	.134(.003)	.143(.004)	.179(.006)	.124(.004)	.092(.002)
$(500, 100, 20)$	.169(.004)	.168(.004)	.184(.007)	.161(.005)	.101(.003)
$(500, 500, 20)$	.255(.005)	.246(.005)	.195(.007)	.255(.005)	.152(.007)
$(500, 1000, 20)$	.274(.004)	.266(.004)	.198(.008)	.274(.004)	.175(.007)
$(200, 2, 20)$	.104(.003)	.083(.003)	.178(.009)	.090(.004)	.078(.002)
$(200, 10, 20)$	.117(.004)	.129(.005)	.204(.009)	.111(.005)	.092(.003)
$(200, 50, 20)$	.182(.005)	.179(.006)	.225(.010)	.177(.006)	.110(.003)
$(200, 100, 20)$	.228(.006)	.214(.006)	.231(.011)	.228(.007)	.127(.005)
$(200, 500, 20)$	.279(.005)	.27(.005)	.244(.009)	.279(.005)	.180(.008)
$(200, 1000, 20)$	.288(.005)	.277(.005)	.248(.009)	.288(.005)	.201(.008)
$(500, 2, 50)$	.101(.002)	.066(.002)	.143(.007)	.065(.002)	.056(.001)
$(500, 10, 50)$	.106(.002)	.106(.002)	.162(.007)	.077(.002)	.070(.001)
$(500, 50, 50)$	.132(.002)	.14(.003)	.177(.007)	.108(.002)	.088(.002)
$(500, 100, 50)$	.166(.003)	.163(.003)	.182(.007)	.150(.004)	.102(.002)
$(500, 500, 50)$	.249(.003)	.231(.003)	.193(.007)	.249(.003)	.146(.003)
$(500, 1000, 50)$	.267(.003)	.252(.003)	.198(.009)	.267(.003)	.170(.004)
$(200, 2, 50)$	.103(.002)	.073(.002)	.186(.012)	.076(.002)	.065(.002)
$(200, 10, 50)$	.117(.003)	.125(.004)	.213(.010)	.100(.004)	.083(.002)
$(200, 50, 50)$	.178(.004)	.173(.004)	.231(.011)	.169(.005)	.111(.002)
$(200, 100, 50)$	.224(.005)	.204(.005)	.237(.010)	.222(.005)	.127(.003)
$(200, 500, 50)$	.271(.004)	.258(.006)	.250(.011)	.271(.004)	.175(.004)
$(200, 1000, 50)$	.278(.003)	.267(.006)	.255(.011)	.278(.003)	.195(.004)

Table 2: Test errors (standard errors in parentheses) of various methods under the Hamming loss based on 100 simulations for complete data, where LSVM, NSVM, P-Ising, CW, and Our denote linear SVM, nonlinear SVM, Pseudo-Ising classifier [8], Breiman's classifier by combining binary relevance linear-SVMs, and the proposed method.

(n, p, q)	LSVM	NSVM	P-Ising	CW	Our
$(500, 2, 20)$	.129(.003)	.091(.003)	.189(.009)	.099(.004)	.091(.003)
$(500, 10, 20)$	.136(.003)	.141(.003)	.209(.006)	.112(.004)	.119(.003)
$(500, 50, 20)$	.168(.004)	.189(.005)	.230(.007)	.152(.004)	.148(.004)
$(500, 100, 20)$	.209(.005)	.224(.005)	.235(.008)	.199(.006)	.168(.004)
$(500, 500, 20)$	.317(.005)	.332(.004)	.250(.008)	.317(.005)	.249(.008)
$(500, 1000, 20)$	.344(.004)	.358(.004)	.253(.009)	.344(.004)	.279(.009)
$(200, 2, 20)$	.133(.004)	.102(.004)	.227(.011)	.109(.005)	.107(.004)
$(200, 10, 20)$	.149(.005)	.167(.006)	.261(.011)	.135(.005)	.138(.004)
$(200, 50, 20)$	.226(.006)	.240(.007)	.286(.012)	.219(.008)	.183(.006)
$(200, 100, 20)$	.283(.008)	.288(.007)	.294(.013)	.282(.008)	.212(.007)
$(200, 500, 20)$	.352(.006)	.363(.004)	.310(.011)	.352(.006)	.286(.010)
$(200, 1000, 20)$	.368(.005)	.372(.003)	.315(.012)	.368(.005)	.314(.010)
$(500, 2, 50)$	.131(.002)	.080(.002)	.185(.009)	.081(.002)	.079(.002)
$(500, 10, 50)$	.138(.002)	.138(.003)	.208(.008)	.096(.003)	.105(.002)
$(500, 50, 50)$	.169(.002)	.190(.003)	.228(.008)	.136(.003)	.141(.003)
$(500, 100, 50)$	.211(.003)	.223(.004)	.234(.009)	.191(.005)	.169(.003)
$(500, 500, 50)$	.315(.004)	.319(.003)	.248(.008)	.315(.004)	.251(.004)
$(500, 1000, 50)$	.340(.004)	.345(.002)	.255(.010)	.340(.004)	.287(.005)
$(200, 2, 50)$	.135(.003)	.091(.003)	.238(.014)	.095(.003)	.094(.003)
$(200, 10, 50)$	.151(.003)	.166(.005)	.274(.012)	.126(.004)	.128(.003)
$(200, 50, 50)$	.226(.005)	.238(.005)	.297(.013)	.214(.007)	.185(.004)
$(200, 100, 50)$	.282(.006)	.280(.006)	.305(.012)	.280(.006)	.216(.005)
$(200, 500, 50)$	.346(.005)	.353(.004)	.322(.013)	.346(.005)	.293(.006)
$(200, 1000, 50)$	.360(.004)	.364(.003)	.328(.013)	.360(.004)	.322(.006)

Table 3: F_1-scores (standard errors in parentheses) of various methods based on 100 simulations for complete data, where LSVM, NSVM, P-Ising, CW, and Our denote linear SVM, nonlinear SVM, Pseudo-Ising classifier [8], Breiman's classifier by combining binary relevance linear-SVMs, and the proposed method.

(n, p, q)	LSVM	NSVM	Pseudo-Ising	CW	Our
$(500, 2, 20)$	.888(.002)	.919(.002)	.84(.007)	.912(.003)	.922(.002)
$(500, 10, 20)$	.882(.003)	.878(.003)	.823(.005)	.901(.003)	.899(.003)
$(500, 50, 20)$	.854(.003)	.841(.004)	.806(.006)	.866(.004)	.879(.003)
$(500, 100, 20)$	.817(.004)	.814(.004)	.801(.007)	.826(.005)	.866(.003)
$(500, 500, 20)$	.723(.005)	.737(.003)	.789(.007)	.723(.005)	.813(.006)
$(500, 1000, 20)$	.703(.004)	.72(.003)	.786(.008)	.703(.004)	.793(.006)
$(200, 2, 20)$	.885(.003)	.911(.003)	.807(.009)	.904(.004)	.908(.003)
$(200, 10, 20)$	.871(.004)	.857(.005)	.780(.010)	.881(.005)	.885(.003)
$(200, 50, 20)$	.803(.006)	.802(.006)	.758(.011)	.808(.007)	.855(.004)
$(200, 100, 20)$	.752(.007)	.767(.006)	.751(.011)	.752(.007)	.836(.005)
$(200, 500, 20)$	.697(.006)	.716(.006)	.737(.010)	.697(.006)	.788(.007)
$(200, 1000, 20)$	.688(.006)	.710(.008)	.733(.010)	.688(.006)	.770(.007)
$(500, 2, 50)$	.882(.002)	.927(.002)	.836(.008)	.927(.002)	.930(.002)
$(500, 10, 50)$	.876(.002)	.876(.002)	.816(.007)	.914(.002)	.908(.002)
$(500, 50, 50)$	.848(.002)	.834(.003)	.799(.007)	.878(.003)	.881(.002)
$(500, 100, 50)$	.811(.003)	.807(.003)	.793(.008)	.830(.004)	.860(.002)
$(500, 500, 50)$	.718(.004)	.735(.003)	.781(.007)	.718(.004)	.805(.003)
$(500, 1000, 50)$	.699(.003)	.716(.003)	.776(.009)	.699(.003)	.782(.004)
$(200, 2, 50)$	.879(.002)	.918(.002)	.789(.012)	.915(.003)	.917(.002)
$(200, 10, 50)$	.865(.003)	.853(.004)	.759(.010)	.887(.004)	.890(.003)
$(200, 50, 50)$	.797(.005)	.796(.004)	.740(.012)	.809(.006)	.849(.003)
$(200, 100, 50)$	.747(.006)	.763(.005)	.733(.011)	.750(.006)	.827(.003)
$(200, 500, 50)$	.694(.004)	.709(.006)	.719(.011)	.695(.004)	.778(.004)
$(200, 1000, 50)$	.687(.004)	.701(.006)	.714(.011)	.687(.004)	.759(.004)

Table 4: Run-times as well as standard errors (in parentheses) in minutes of methods in Table 1 based on 100 simulations. Here LSVM, NSVM, P-Ising, CW, and Our denote linear SVM, nonlinear SVM, Pseudo-Ising classifier [8], and the proposed method.

(n, p, q)	LSVM	NSVM	Pseudo-Ising	CW	Our
$(500, 2, 20)$	.256(.025)	1.05(.106)	241(9.12)	.955(.023)	7.75(.262)
$(500, 10, 20)$	.448(.038)	2.15(.131)	208(6.38)	1.31(.130)	11.6(.444)
$(500, 50, 20)$	2.29(.199)	8.85(.471)	183(3.85)	3.80(.200)	25.8(4.53)
$(500, 100, 20)$	4.94(.602)	15.7(.903)	195(4.58)	6.79(.359)	37.1(5.68)
$(500, 500, 20)$	16.4(.800)	68.2(3.40)	311(3.68)	27.0(.455)	75.1(6.54)
$(500, 1000, 20)$	42.6(4.05)	176(7.06)	447(3.71)	67.0(4.81)	104(6.67)
$(200, 2, 20)$	.145(.006)	.539(.018)	274(4.40)	.569(.014)	4.25(.129)
$(200, 10, 20)$	.234(.007)	1.08(.024)	258(2.91)	.732(.019)	4.97(.196)
$(200, 50, 20)$	.84(.041)	3.64(.078)	262(2.40)	1.54(.054)	6.74(.202)
$(200, 100, 20)$	1.35(.034)	6.20(.077)	265(1.74)	2.42(.059)	8.13(.062)
$(200, 500, 20)$	9.17(.233)	34.9(.722)	319(2.03)	15.7(.203)	16.1(.339)
$(200, 1000, 20)$	19.0(.460)	69.7(.932)	385(2.12)	32.8(1.08)	23.6(.512)
$(500, 2, 50)$	.671(.021)	2.69(.072)	816(11.7)	4.09(.092)	14.2(.229)
$(500, 10, 50)$	1.38(.034)	5.96(.111)	693(5.37)	5.37(.104)	20.0(.289)
$(500, 50, 50)$	6.38(.203)	22.6(.434)	773(4.83)	12.1(.25)	35.1(.529)
$(500, 100, 50)$	14.4(.739)	42.5(1.17)	874(5.17)	21.3(.766)	47.6(1.01)
$(500, 500, 50)$	46.8(.87)	195(3.33)	1627(5.57)	81.7(.478)	109(3.26)
$(500, 1000, 50)$	164(21.1)	641(14.5)	2570(6.99)	271(20.5)	164(5.18)
$(200, 2, 50)$	.282(.021)	1.12(.079)	454(10.7)	1.54(.035)	12.3(.389)
$(200, 10, 50)$	.463(.022)	2.34(.083)	441(11.2)	1.83(.037)	14.9(.422)
$(200, 50, 50)$	1.70(.083)	7.54(.275)	462(10.0)	3.5(.077)	24.6(2.65)
$(200, 100, 50)$	2.71(.124)	13.1(.462)	495(9.67)	5.28(.104)	36(5.47)
$(200, 500, 50)$	17.4(.600)	68.0(2.00)	780(8.22)	31.2(.216)	70.8(5.34)
$(200, 1000, 50)$	36.4(1.17)	140(3.55)	1130(6.94)	63(2.6)	97.1(4.84)

Table 5: Test errors (standard errors in parentheses) of the proposed method without and with additional labels based on 100 simulations.

(n, p, q)	w/o additional labels	w additional labels
$(500, 2, 20)$	.387(.005)	.345(.004)
$(500, 10, 20)$	.377(.005)	.346(.004)
$(500, 50, 20)$	.367(.005)	.346(.003)
$(500, 100, 20)$	.361(.005)	.345(.004)
$(500, 500, 20)$	.352(.005)	.344(.004)
$(500, 1000, 20)$	.350(.004)	.343(.003)
$(200, 2, 20)$	.388(.006)	.349(.006)
$(200, 10, 20)$	.383(.007)	.349(.006)
$(200, 50, 20)$	.373(.008)	.347(.005)
$(200, 100, 20)$	.367(.008)	.346(.005)
$(200, 500, 20)$	.356(.007)	.344(.004)
$(200, 1000, 20)$	.354(.008)	.344(.004)
$(100, 2, 20)$	.387(.008)	.351(.007)
$(100, 10, 20)$	.386(.008)	.351(.006)
$(100, 50, 20)$	.377(.009)	.350(.006)
$(100, 100, 20)$	.373(.010)	.348(.006)
$(100, 500, 20)$	.365(.014)	.347(.007)
$(100, 1000, 20)$	.363(.013)	.347(.007)
$(50, 2, 20)$	.388(.008)	.353(.008)
$(50, 10, 20)$	.387(.010)	.353(.007)
$(50, 50, 20)$	.381(.013)	.351(.008)
$(50, 100, 20)$	.377(.016)	.350(.008)
$(50, 500, 20)$	.370(.013)	.349(.007)
$(50, 1000, 20)$	.371(.016)	.350(.008)

Table 6: True positive and false positive error rates (TPR and FNR) of a novel label detection of the proposed method based on 100 simulations, where TPR>0 indicates its capability and all other competing methods are impossible with TPR$=0$.

(n,p)	TPR	FPR
(200,2)	.036(.022)	.354(.050)
(200,10)	.023(.018)	.255(.073)
(200,50)	.006(.009)	.068(.065)
(200,100)	.002(.005)	.017(.035)
(200,500)	.000(.000)	.000(.000)
(200,1000)	.000(.000)	.000(.000)
(500,2)	.044(.017)	.359(.032)
(500,10)	.021(.014)	.320(.044)
(500,50)	.011(.012)	.158(.073)
(500,100)	.005(.006)	.072(.060)
(500,500)	.000(.001)	.001(.003)
(500,1000)	.000(.000)	.000(.000)

Table 7: Label-label weight matrix $\boldsymbol{W}_{-}$ for the reduced COCO object image detection task involving six objects.

	person	car	motorcycle	bus	truck	train
person	1	.0	.0	.0	.0	.0
car	0	.4	.1	.2	.2	.1
motorcycle	0	.1	.6	.1	.1	.1
bus	0	.2	.1	.4	.2	.1
truck	0	.2	.1	.2	.4	.1
train	0	.1	.1	.1	.1	.6

Table 8: Averaged test errors for the full COCO object detection image data as well as corresponding standard errors in parentheses based on 100 random subsamples involving six categories with the average numbers of positively labeled observations 3252, 621, 178, 200, 311, 182. Here LSVM, NSVM, P-Ising, CNN [16], and Proposed denote linear SVM, nonlinear SVM, Pseudo-Ising classifier [8], a convolutional neural network method, and the proposed method. Note that "NA" means that a routine cannot return a value.

Full	Weighed loss (3)	Hamming	F_1-score				
LSVM	NA	NA	NA				
NSVM	NA	NA	NA				
P-Ising	NA	NA	NA				
CNN	.0025	.035	.093				
Proposed	.0022	.025	.228				
Evaluation loss: L in (1) for 100 random samples							
	Person	Car	Motorcycle	Bus	Truck	Train	Overall
LSVM	.094(.005)	.069(.005)	.008(.002)	.012(.002)	.037(.004)	.006(.001)	.038(.001)
NSVM	.041(.003)	.013(.002)	.001(.000)	.002(.000)	.004(.001)	.001(.000)	.010(.001)
P-Ising	.179(.008)	.250(.009)	.320(.007)	.316(.010)	.263(.012)	.328(.009)	.276(.005)
CNN	.013(.002)	.005(.000)	.002(.000)	.002(.000)	.003(.000)	.001(.000)	.004(.000)
Proposed	.011(.000)	.005(.000)	.002(.000)	.002(.000)	.003(.000)	.001(.000)	.004(.000)
Evaluation loss: Hamming for 100 random samples							
	Person	Car	Motorcycle	Bus	Truck	Train	Overall
LSVM	.175(.006)	.123(.005)	.022(.002)	.027(.002)	.067(.004)	.014(.001)	.071(.002)
NSVM	.118(.003)	.081(.002)	.018(.001)	.022(.001)	.044(.001)	.012(.000)	.049(.001)
P-Ising	.251(.006)	.306(.01)	.339(.008)	.338(.01)	.298(.012)	.347(.011)	.313(.006)
CNN	.385(.008)	.107(.000)	.032(.000)	.038(.000)	.050(.000)	.031(.000)	.107(.001)
Proposed	.299(.007)	.107(.000)	.032(.000)	.038(.000)	.050(.000)	.031(.000)	.093(.001)

Table 9: Test errors for COCO object detection image data as well as corresponding standard errors in parentheses based on a subsample of 600 observations involving six categories. Here LSVM, NSVM, P-Ising, CNN, Proposed, and Integration denote linear SVM, nonlinear SVM, Pseudo-Ising classifier [8], a convolutional neural network method, the proposed method using only complete data, and the proposed method integrating additional labels. The best performer is marked in **bold**.

Evaluation loss: L in (1)							
	Person	Car	Motorcycle	Bus	Truck	Train	Overall
LSVM	.2364	.3026	.1593	.1997	.3147	.0380	.2084
NSVM	.4322	.2504	.0168	.0797	.2591	.0010	.1732
P-Ising	.3601	.4412	.3529	.3252	.3638	.3322	.3626
CNN	.0915	.0312	.0016	.0047	.0043	.0012	.0224
Proposed	.2316	.1564	.0013	.0021	.0222	.0012	.0691
Integration	.0645	.0543	.0013	.0017	.0029	.0012	.0210
Evaluation loss: Hamming							
	Person	Car	Motorcycle	Bus	Truck	Train	Overall
LSVM	.2884	.3224	.1712	.2102	.3290	.0468	.228
NSVM	.4344	.2718	.0298	.0890	.2692	.0250	.1865
P-Ising	.4424	.4780	.3728	.3516	.3802	.3580	.3971
CNN	.2218	.0954	.0310	.0306	.0510	.0314	.0768
Proposed	.2770	.1770	.0246	.0276	.0610	.0314	.0997
Integration	.0442	.0602	.0013	.0018	.0027	.0012	.0186

Table 10: Averaged test errors for the full PASCAL object detection image data (2007 version), as measured by the weighed loss (3), the Hamming distance, and the F_1-score. Here LSVM, NSVM, P-Ising, CNN [16], and Proposed denote linear SVM, nonlinear SVM, Pseudo-Ising classifier [8], a convolutional neural network method, and the proposed method.

Method	Weighted loss (3)	Hamming	F_1-score
LSVM	.018	.038	.690
NSVM	.007	.031	.399
P-Ising	.329	.390	.784
CNN	.010	.073	.453
Proposed	.006	.038	.757

Table 11: Averaged test errors for the full Mediamill data [25], as measured by the weighed loss (3), the Hamming distance, and the F_1-score. Here LSVM, NSVM, P-Ising, and Proposed denote linear SVM, nonlinear SVM, Pseudo-Ising classifier [8], and the proposed method. Here NA means that the method did not produce a result after running for one week.

Method	Weighted loss (3)	Hamming	F_1-score
LSVM	NA	NA	NA
P-Ising	NA	NA	NA
NSVM	.0063	.0302	.5406
Proposed	.0032	.029	.5714