Soft Label Based Semi-Supervised Boosting for Classification and Object Recognition

Dingfu Zhou, Benjamin Quost, Vincent Frémont

Soft Label Based Semi-Supervised Boosting for Classification and Object Recognition

Dingfu Zhou, Benjamin Quost and Vincent Frémont
Université de Technologie de Compiègne,
UMR CNRS 7253 Heudiasyc,
Compiègne, France
Email: {*dingfu.zhou, benjamin.quost, vincent.fremont*}*@hds.utc.fr*

***Abstract*—Supervised classification algorithms such as Boosting and SVM have achieved significant success in the field of computer vision for classification and object recognition. However, the performance of the classifier decreases rapidly if there are insufficient labeled training samples. In this paper, a semi-supervised boosting algorithm is proposed to overcome this limitation. First, a few labeled instances are use to estimate probabilistic class labels for unlabeled samples using Gaussian Mixture Models after a dimension reduction step performed via Principal Component Analysis. Then, we apply a boosting strategy on decision stumps trained using the soft labeled instances thus obtained. The performances of our strategy are evaluated on several state-of-the-art classification datasets, as well as on a pedestrian detection and recognition problem. Experimental results demonstrate the interest of taking into account additional data in the training process.**

I. Introduction and Related Work

In the last decade, machine learning techniques have proven their interest for vision-based object detection. The success of these methods own to both the advance in machine learning and the breakthrough of object descriptors. Advances in image processing have established local [1] and context-based [2] features as key approaches for object detection and recognition. Besides, recent works have shown the great interest of using Support Vector Machines (SVMs) [3], Boosting [4], Random Forests [5] and Neural networks [6] for this purpose. Note, however, that all of these methods are sensitive to the size of the training set. More precisely, due to the phenomenon known as "curse of dimensionality", the amount of training data at hand may need to be huge.

In some practical problems there are insufficient labeled training samples, whereas labeling training data is tedious and time consuming. Many approaches have been proposed to overcome this problem by introducing an amount of unlabeled samples to help construct the classifier. This type of approach is referred to as semi-supervised learning (SSL) [7]. The essential idea of SSL is to utilize the knowledge from both labeled and unlabeled training samples to build a better decision boundary between different classes. A number of approaches have been proposed e.g., self-training [8], co-training [9], transductive support vector machines [10], graph-based methods [11], semi-boosting [12], [13], [14] and semi-supervised Random Forests [15]. In most of the SSL approaches, one or several semi-supervised assumptions, such as smoothness, cluster and manifold assumptions, are used to assign pseudo labels to the unlabeled samples. Then pseudo-labeled training instances are used to build the classifier. Note that SSL may be formalized using soft labels [16]: for example, each training instance may be associated with a probability distribution over the set of classes. The probability of a given class label may be interpreted in a frequentist setting as the proportion of similar training instances having this label. Obviously, 0/1 probabilities may be used when the class label is known with certainty.

In a way, assigning soft labels to the (unlabeled) training instances corresponds to integrating some prior knowledge in the training set. In [17] and [18], the authors propose to design an objective function before using boosting. Alternatively, soft labels may be estimated from the distribution of the data. In [16], soft class labels are estimated using a kernel density estimation (KDE) approach.

In this work, we use HOG descriptors to obtain training instances from the raw images. Since clustering techniques are known to be sensitive to the lack of available data, especially in high dimensions, we first select a limited amount of relevant features using PCA. We experimentally show that this step has very little influence on the clustering step. Then, we fit a Gaussian Mixture Model (GMM) [19] to the training data, and we use the posterior probabilities as soft labels. Note that as a parametric density estimation technique, GMM are known to be more robust to the lack of training data than kernel density estimation. Eventually, in the classification step, we propose to boost decision stumps trained from the probabilistic labeled training instances [20].

Our paper is organized as follows. Section II presents the soft label estimation step using PCA and GMM. In Section III, we remind how probabilistic labels may be used in decision tree inference, and we present our boosting strategy for such a base classification algorithm. Section IV details our experiments. Eventually, we conclude the paper with some perspectives of work in Section V.

II. Soft Class Label Estimation

In our approach, the training data are expressed as $D = D^L + D^U$, where D^L and D^U are the labeled and unlabeled instances respectively. $D^L = \{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_n, y_n)\} \subseteq \mathcal{X} \times \mathcal{Y}$, in which $\boldsymbol{x}_i \in \mathbb{R}^d$ stands for the d- dimensions feature of ith instance, $y_i \in \mathcal{Y}$ is the class label and n is the

number of labeled instances. Here, we only focus on the binary classification problem, therefore $\mathcal{Y} = \{1, 2\}$. The unlabeled samples are expressed as $D^U = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_m\} \subseteq \mathcal{X}$, in which m is the number of unlabeled instances. We also use $N = m + n$ to represent the number of all the training instances.

A. HOG Feature Extraction

HOG features have been recognized as one of the most significant features for pedestrian detection. Owing to its excellent performance, it is also widely used for other object detection in computer vision. As described in [1], the standard HOG descriptor for a 64*128 window size pedestrian sample will have 3780 dimension, in which we chose block-size as 16×16, cell-size as 8×8 and block-stride as $(8, 8)$. Compare to the original image pixels, this high dimension HOG descriptor is also acceptable. However, according to [3], the extracted HOG features also include a lot of redundant information. They have proved that a classifier trained by PCA-based reduced low dimension features has a similar performance with the classifier trained with the high dimension features. Moreover, a lower dimensional features leads to a model with fewer parameters and speeds up the learning and detection algorithms. PCA-HOG feature has also been used in [21] and [22] for pedestrian detection and people counting. In our approach, we first extract standard HOG descriptors from all the training samples, then PCA is applied to generate lower dimension features. Experimental results detailed in Section IV, show that a PCA-HOG feature with $20-$dimension can achieve good clustering results.

B. GMM based Class Label Estimation

A Gaussian Mixture Model (GMM) is a parametric density estimation technique, in which the distribution of the data is supposed to be a mixture of g multivariate Gaussians:

$$p(\boldsymbol{x}|\boldsymbol{\Psi}) = \sum_{k=1}^{K} \theta_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \tag{1}$$

where, θ_k is the prior probability of the kth component of the model satisfying that $\sum_k^K \theta_k = 1$ and $\theta_k > 0$; $\boldsymbol{x}$ is a $d-$dimensional data feature. Each component model is parametrized by a $d\times1$ mean vector $\boldsymbol{\mu}_k$ and a $d\times d$ covariance matrix $\boldsymbol{\Sigma}_k$. Classically, a set of unlabeled instances is considered, and maximum likelihood estimates of the parameters are computed using the EM algorithm [23], [24]. This algorithm considers, for each instance $\mathbf{x}_i$, g latent (random) variables z_{ik} indicating from which component the instance was generated. Given a set of initial parameters $\Psi = \{\theta_k, \boldsymbol{\mu}_k, \Sigma_k\}$, $k = 1, \ldots, K$, the algorithm iterates two successive steps:

- in the E step, the expectation of z_i is computed given the current fit of the parameters: $\pi_{ik} = \frac{\theta_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{l=1}^{g} \theta_l \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}$.
- in the M step, new estimates of the parameters are computed.

When labeled instances are available, the corresponding latent variables are assumed to be known, and the expectations π_{ik} need not to be estimated. The advantage of using such labeled instances is twofold. First, they can be used to compute (nontrivial) starting values for the parameters. Furthermore, they may guide the algorithm towards a desired solution. After the optimal parameters ($\Psi^o = \{\theta_k^o, \mu_k^o, \Sigma_k^o\}$) of the GMMs model have been obtained, the optimal posterior probabilities of the unlabeled samples for both classes can also be computed via Bayes formulas:

$$p(y_i = k|\boldsymbol{x}_i) = \frac{\theta_k^o \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k^o, \boldsymbol{\Sigma}_k^o)}{\sum_{l=1}^{2} \theta_l^o \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_l^o, \boldsymbol{\Sigma}_l^o)}$$

For the covariance matrices Σ, both diagonal and full covariance matrices have been investigated in our study. A GMM using diagonal matrices typically need less training data and is faster than a GMM with full matrices. The experimental results given in Section IV, also show that the diagonal matrices outperforms the full matrices for the class label estimation.

III. Semi-Supervised Boosting with Probabilistic Label

After obtaining class probabilities for all the unlabeled samples, our classification problem can be expressed in the following form: $D = \{(\boldsymbol{x}_1, \boldsymbol{\pi}_1), \cdots, (\boldsymbol{x}_N, \boldsymbol{\pi}_N)$, where $\boldsymbol{\pi}_i = \{\pi_{i,k}\}$, $k = 1, 2$, $\pi_{i,k} = \widehat{P}(Y = k|\mathbf{x})$ being the probabilistic label estimated by GMM for class k.

A. Soft Decision Tree

Decision stumps are commonly used as base classification algorithm (or weak learner) in boosting algorithm for object detection. Here, we describe how such classifiers (and more generally decision trees) may be trained using probabilistic labeled data. More details may be found in [20], [18]. We remind that this classification algorithm aims at classifying the data by splitting the feature space into subsets. For a given node, the frequency of each class is first estimated, as the proportion of the instances belonging to this class falling into the node. Then, the Gini index is computed over this distribution of frequencies. The index thus obtained may be used to evaluate the quality of a split. Let s_d be a candidate split value according to the dth feature (instances such that $x_i^d < s_d$, resp. $x_i^d \geq s_d$, thus fall in the left-hand child node, resp. right-hand one). Then, the best split is the one that maximizes the purity criterion of both left-hand and right-hand children.

In the case of probabilistic labeled instances, the Gini index may still be used: indeed, the frequencies of the classes are obtained by averaging the probabilistic labels of the instances falling into the node. In the following text we take this soft decision stump as the weak learner algorithm.

B. Soft Label based Boosting

Boosting aims at training an accurate classifier (or strong learner) by combining a number of simple classifiers (or weak learners). Each weak learner is chosen so as to optimally reduce the training error. In this paper, we focus on the AdaBoost algorithm [25] for binary classification problems

here. In AdaBoost, the decision $H(\mathbf{x})$ made by the strong learner for a given instance x is obtained by combining the weak learners outputs $h_t(\mathbf{x}) \in \{0, 1\}$ in the following way:

$$H(\mathbf{x}) = \begin{cases} 1, & \text{if } \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}_i) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha_t > 0$ the weight of the tth weak learner ($t = 1, \ldots, T$), and T is the number of weak learners. During the training process, a set of weights are assigned to the train samples. In order to use the soft labeled training instances in the AdaBoost algorithm, several parts should be modified: 1) the weak learner should be able to handle both hard and soft labeled training instances, 2) the classification error of the weak learner should be computed for both hard and soft labeled samples. The first issue may be tackled using the soft decision tree algorithm in Section III-A. In the following, we will introduce how to calculate the classification error for soft labeled instances.

The cost-weighted misclassification error for instance $\mathbf{x}$ can be written as:

$$E_{Y|\mathbf{X}=\mathbf{x}} L(y|h_t(\mathbf{x})) = \eta(\mathbf{x}).\mathrm{I}_{h_t(\mathbf{x}) \leq 0.5} + (1 - \eta(\mathbf{x})).\mathrm{I}_{h_t(\mathbf{x}) > 0.5} \quad (2)$$

where $\mathrm{I}_{\text{condition}}$ stands for the indicator function (it is equal to 1 if the condition given is met, and 0 otherwise). In Eq. (2), we only consider the probability of class 1 estimated by the GMM, noted as $\eta(\mathbf{x}) = \widehat{P}(Y = 1|\mathbf{x})$ (since we consider a two-class problem, the probability of class 2 is directly deduced). In the boosting algorithm, we need to compute the classification error $\epsilon(t)$ for each weak learner, which we define as a weighted version of the misclassification cost:

$$\epsilon(t) = \frac{1}{N} \sum_{i=1}^{N} w_i^t . \epsilon_{\mathbf{x}_i} \quad (3)$$

where w_i^t is the weight associated with instance $\mathbf{x}_i$ at the tth iteration, and $\epsilon_{\mathbf{x}_i} = \pi_{i1} \mathrm{I}_{h_t(\mathbf{x}_i) < 0.5} + (1 - \pi_{i1}) \mathrm{I}_{h_t(\mathbf{x}_i) \geq 0.5}$ is the misclassification cost for instance x_i. Note that $d_{er}(x_i)$ is also used for updating the weight w_i^t along the iterations of the boosting algorithm:

$$w_i^{t+1} = w_i^t \beta(t)^{1 - \epsilon_{\mathbf{x}_i}} \quad (4)$$

where $\beta(t)$ is defined as $\beta(t) = \frac{\epsilon(t)}{1 - \epsilon(t)}$. Eventually, the weight α_t of the weak learner is classically defined as $\alpha_t = log \frac{1}{\beta(t)}$. Algorithm 1 summarizes the main steps of our soft label based boosting algorithm.

Remark that the proposed soft label based boosting algorithm can take both hard and soft labeled samples as inputs. If all the samples are hard labeled, it degrades into the original AdaBoost algorithm. Note furthermore that an instance with a probabilistic label close to the uniform probability distribution will have a significant classification error even if it is well classified. This reflects the difficulty to learn from instances from uncertain information. In order to let the classifier focus on the training instances classified with confidence, we set the initial weights of the instances to $w_i^1 = 2 * abs(\pi_{i1} - 0.5)$.

Algorithm 1 Soft label based Boosting algorithm.

Inputs: Training data $(\boldsymbol{x}_i, \boldsymbol{\pi}_i), i = 1, \cdots, N$, $\pi_{i,1} = \widehat{P}(y_i = 1|\mathbf{x}_i)$. T, number of weak learners, weak learner algorithm .

Initial: sample weight vector: $w_i^1 = 2 * abs(\pi_{i,1} - 0.5)$.

For t=1:T

1) Normalization sample weight: $w_i^t = \frac{w_i^t}{\sum_{i=1}^{N} w_i^t}$;
2) Call $WeakLearner$ with distribution w_i^t , get back a hypothesis $h_t : \mathcal{X} \to \{0, 1\}$;
3) Calculate the expected classification error for sample $\mathbf{x}_i$:

$$\epsilon_{\mathbf{x}_i} = \pi_{i,1} I_{h_t(\mathbf{x}_i) \leq 0.5} + (1 - \pi_{i,1}) I_{h_t(\mathbf{x}_i) \geq 0.5}$$

4) Calculate the expected error for weak learner t:

$$\epsilon_t = \sum_{i}^{N} w_i^t . \epsilon_{\mathbf{x}_i}$$

5) Set $\beta(t) = \frac{\epsilon_t}{1 - \epsilon_t}$ and $\alpha_t = log \frac{1}{\beta(t)}$;
6) Update new weight to each sample:

$$w_i^{t+1} = w_i^t . \beta(t)^{1 - \epsilon_{\mathbf{x}_i}}$$

end

Output: The final decision:

$$H(\mathbf{x}_i) = \begin{cases} 1, & \text{if } \sum_{t=1}^{T} \alpha_t . h_t(\mathbf{x}_i) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0, & \text{otherwise} \end{cases}$$

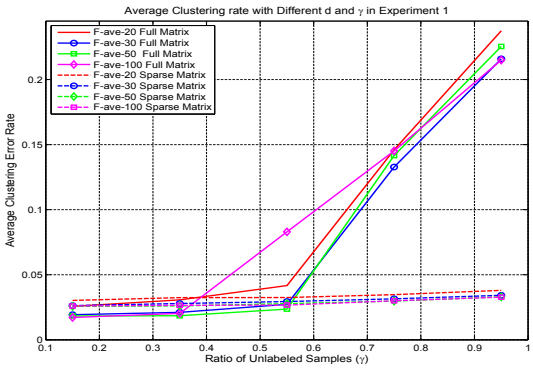
For a hard labeled sample, the initial weight is 1, while a 0 initial weight will be given to a sample who has equal classes probabilities.

IV. Experiments and Analysis

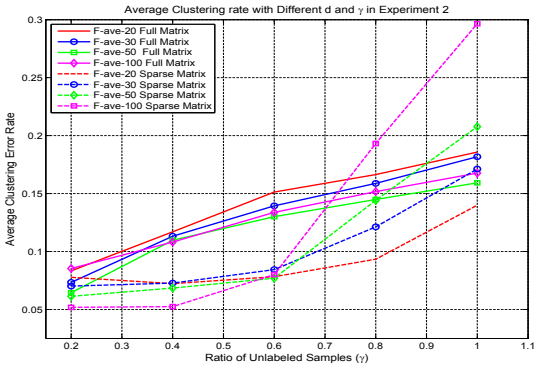
A. Performance of Probabilistic Label Estimation

In order to evaluate the performance of the GMM based probabilistic class labels estimation, we test our algorithm on the CVC [26] and INRIA [1] pedestrian dataset. The CVC dataset has 3172 positive (pedestrian) and 15150 negative (non-pedestrian) samples while the INRIA dataset has 3542 positive and 4560 negative samples (randomly extracted from the negative images). In these experiments, all the samples are randomly separated into labeled and unlabeled with a ratio value γ, which is defined as: $\gamma = \frac{m}{n+m}$, where, n and m represent the number of labeled and unlabeled samples respectively. First, the HOG features [1] are computed for each sample, then PCA is applied to reduce the features dimension to a lower one denoted as d. In this experiment, we also compare the GMM performances with diagonal and full covariance matrices. Details about the two different experiments are described below:

- Experiment 1: Only CVC dataset is used. The γ values range from 0.15 to 0.95. At each value, four different feature dimensions d have been used.
- Experiment 2: Both the CVC and INRIA datasets are used in this experiment. We always consider the samples from



(a) Experiment 1



(b) Experiment 2

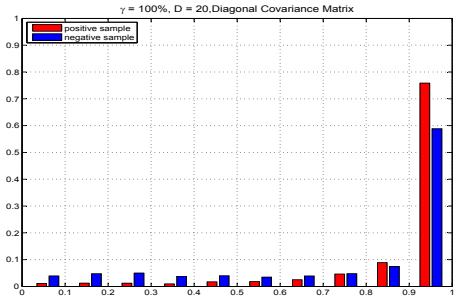
Figure 1: Performances of GMM based probabilistic class labels estimation.

the CVC dataset as the labeled data. Then we randomly divide the INRIA dataset into two parts with a ratio γ. We keep a proportion of γ as the unlabeled data, while the rest is considered as labeled data.

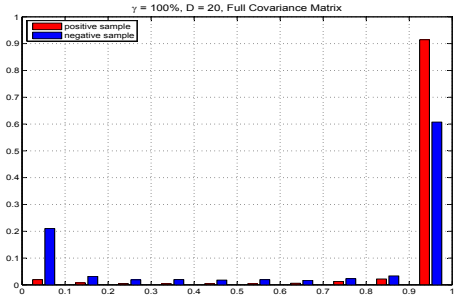
In order to evaluate the performance of the label estimation, we use a simple and rough method. A class label with the highest probability is assigned to each unlabeled sample. All the experiments have been repeated five times, and the average error rate is defined as: $F\_ave = \frac{m_{mis}}{m}$, where, m_{mis} is the number of samples that have been misclassified by the GMM.

Fig. (1) describes the average error rates of the two experiments. In the figure, the solid and dotted lines represent the results with full and diagonal covariance matrix respectively. In experiment 1, the full matrix performs a little better than the diagonal matrix when there are enough labeled samples, but it decreases rapidly with the decreasing of labeled samples. However, the diagonal matrix keeps stable with the decreasing of labeled samples. In the experiments 2, we have found something different. The performance of the diagonal matrix highly depends on the feature dimension, and we obtained that the diagonal matrix together with $20-$dimension feature gives the best result.

Fig. (2) shows more details about the GMM based probability label estimation results. Here, due to space limitations, we just show the results in experiment 2 with $d = 20$ and $\lambda = 1.0$. The red and blue columns represent the actual class of the



(a) Diagonal Matrix



(b) Full Matrix

Figure 2: Distribution of estimated probabilistic class labels.

sample that can be positive or negative. The X-axis values are the estimated probabilities of a sample obtained by the GMMs model. In order to properly display the distribution of the estimated class probabilities, we uniformly divide interval $[0, 1]$ (the class probability belongs to $[0, 1]$) into 10 equal parts and record the number of samples falling into each sub-interval. The Y-axis represents the ratio of the samples that fall into the corresponding sub-interval. From Fig. (2), we can see that the full matrix works well for the positive samples compared to the diagonal matrix. However, it gives worse performances for the negative samples, especially when some of them have been set with high probabilities for the incorrect class (left-most side of Fig. (2)-(b)). Compared to the full matrix, one can notice that the diagonal matrix gives an eclectic result. It has a weaker ability to classify the positive samples while it makes less clustering errors on the negative samples. Additionally, the diagonal matrix gives less average error for all the samples (see in Fig. (1)).

B. Pedestrian Recognition

We also took the INRIA pedestrian dataset to evaluate the algorithm for pedestrian recognition. Firstly we randomly separated the data into training and testing with a ratio $3 : 1$. Then we randomly selected a part of the training data as the labeled sample and left the remaining data as unlabeled data. We followed the method in [1] to represent each sample by a $3780-$dimensional HOG feature. Then a PCA has been applied to the resulting vector to obtain a $20-$dimension feature for each sample. Guided by the labeled samples, the probabilistic class labels for unlabeled training samples are estimated by GMM using the EM algorithm. Here, the reduced $20-$ di-

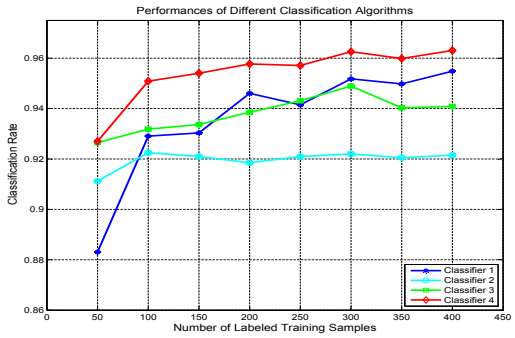


Figure 3: Recognition rates of four different classifiers. We choose $T = 500$ for all boosting classifiers in this experiment.

mension features are only used for estimating the soft class labels. In boosting algorithm, the original HOG features are used for recognition. The AdaBoost algorithm is used as the baseline and is trained with few labeled samples. Four different kinds of classifiers have been designed in this experiment: 1) classifier 1 is a classical AdaBoost classifier trained using only few labeled samples, 2) classifier 2 is a GMM classifier using both the labeled and unlabeled samples, 3) classifier 3 is the proposed semi-supervised boosting classifier trained using all the hard and soft labeled samples, 4) classifier 4 is also the proposed semi-supervised boosting classifier trained using all the labeled and some selected unlabeled samples. Classifier 1, 3 and 4 use stump decision tree as the weak learner. Hard label decision tree is used in Classifier 1, while soft decision tree is applied in both Classifier 3 and 4.

All the experiments have been repeated five times and we calculated the average recognition rate to draw the Fig. (3). From this figure, we can see that the recognition rate of Classifier 1 (blue line) increases obviously with the increasing of labeled training samples, however, the recognition rate of Classifier 2 does not increase with the increase of the labeled samples. The green line represents the performance of Classifier 3. Although it increases with the increasing of labeled samples, it gives worse performance than Classifier 1 which has been trained using only few labeled samples. Through additional experiments we found that the weak performance of Classier 3 is mainly caused by the training samples who have been assigned with wrong probabilistic class labels. The GMM can give the right probabilistic class label for most of the unlabeled samples, while it also gives wrong label to some training samples. These wrong probabilistic labeled samples have a great influence on the recognition rate of Classifier 3.

In order to reduce this negative effect, we should remove some of unlabeled samples with wrong probabilistic labels. A effective strategy has been proposed as: we can predict the class labels for all the unlabeled training samples using GMM method (Classifier 2) and at the same time we can also predict another class label for each unlabeled training sample using Classifier 1 (because it is trained using only the labeled samples). For an unlabeled sample, we consider its probabilistic class label is reliable if its predicted class labels from Classifier 1 and 2 are consistent. Otherwise, it is considered as unreliable. Classifier 4 is trained using all the labeled samples and some selected unlabeled samples with reliable probabilistic class label. Although our strategy was not to get rid of all the wrong probabilistic samples from the training data, it still improve the recognition rate obviously. This improvement can be seen from the red line of Fig. (3). By comparing the red and blue lines in Fig. (3), we can find that our proposed boosting algorithm with additional soft labeled sample gives a better recognition rate than AdaBoost trained with only labeled samples.

C. Data Classification

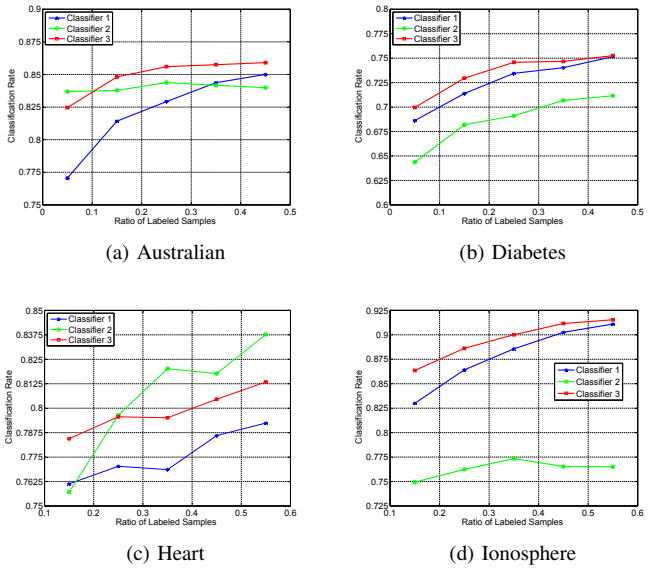
In order to evaluate the proposed semi-supervised boosting algorithm further, we test our approach on public binary classification datasets in the LibSVM repository [27]. Four different dataset have been chosen to test our algorithm. A general description of the data is in the following table.

Dataset	Size of the data	Attribute dimension
Australian	690	14
Diabetes	768	8
Heart	270	13
Ionosphere	351	34

As in the previous section, all the data have been randomly divided into training and testing data with a proportion 3:1. In the training data, a small part of the training samples are selected as labeled samples and the rest are considered as unlabeled samples. In this experiment, only three classifiers are designed: 1) classifier 1 is a classical AdaBoost classifier trained using only few labeled samples, 2) classifier 2 is a GMM classifier using both the labeled and unlabeled samples, 3) classifier 3 is the proposed semi-supervised boosting classifier trained using all the labeled and some selected unlabeled samples and the selection strategy is the same as in Sec. IV-B. For each dataset, the ratio of labeled samples changed from 0.05 to 0.55. Fig. (4), show the classification rate of the four different datasets in each sub-figure respectively. From the figure we can easily find that our proposed semi-supervised boosting algorithm (red line) has a higher classification rate than classical AdaBoost algorithm (blue line) most of the time. The additional unlabeled samples help to improve the classification rate in our semi-supervised boosting algorithm.

V. Conclusions and Future Work

In this paper, we proposed a semi-supervised boosting algorithm that uses few labeled samples and an amount of unlabeled samples. Our approach provides a novel framework to take hard and soft labeled samples into the boosting algorithm. In this paper, the soft class labels of the unlabeled samples are estimated from GMM using the labeled samples. Experimental results on public datasets have shown that the performance of a boosting classifier could be improved by adding soft labeled samples. We also have found that PCA-based HOG features dimension reduction has a little influence



(a) Australian (b) Diabetes

(c) Heart (d) Ionosphere

Figure 4: Classification rate of four different Dataset. All the experiments have been repeated 50 times. We choose $T = 100$ for all boosting classifiers in this experiment.

on the results of probabilistic class labels estimation. This interesting phenomena makes it is possible to estimate soft class labels for pedestrian samples with HOG features. Finally, the results show that the recognition rate can be improved by adding an amount of soft labeled samples.

On the other hand, the soft class labels could be replaced by other prior information. In the future, we plan to design our approach to improve the classifier's performance by using additional samples with weak prior knowledge (e.g., human beliefs or uniform class probabilities). For the pedestrian detection problem, we can apply our approach to improve a generic pedestrian detector (which is trained on one or several generic public datasets) by adding plentiful unlabeled samples from some specific scenes.

REFERENCES

[1] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005*, volume 1, pages 886–893, 2005.

[2] Yuanyuan Ding and Jing Xiao. Contextual boost for pedestrian detection. In *Computer Vision and Pattern Recognition, 2012*, pages 2895–2902. IEEE, 2012.

[3] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, 2010.

[4] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001.*, volume 1, pages 5–11. IEEE, 2001.

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

[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, volume 1, page 4, 2012.

[7] Olivier Chapelle, Bernhard Schölkopf, Alexander Zien, et al. *Semi-supervised learning*, volume 2. MIT press Cambridge, 2006.

[8] John Besemer, Alexandre Lomsadze, and Mark Borodovsky. Genemarks: a self-training method for prediction of gene starts in microbial genomes. implications for finding sequence motifs in regulatory regions. *Nucleic Acids Research*, 29(12):2607–2618, 2001.

[9] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM, 1998.

[10] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *ICML*, volume 99, pages 200–209, 1999.

[11] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[12] Christian Leistner, Helmut Grabner, and Horst Bischof. Semi-supervised boosting using visual similarity learning. In *Computer Vision and Pattern Recognition, 2008*, pages 1–8, 2008.

[13] Pavan Kumar Mallapragada, Rong Jin, Anil K. Jain, and Yi Liu. Semiboost: Boosting for semi-supervised learning. *PAMI*, pages 2000–2014, 2009.

[14] Ke Chen and Shihai Wang. Semi-supervised learning via GENOMES regularized boosting working on multiple semi-supervised assumptions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, pages 129–143, 2011.

[15] Christian Leistner, Amir Saffari, Jakob Santner, and Horst Bischof. Semi-supervised random forests. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 506–513. IEEE, 2009.

[16] Weihong Wang, Yang Wang, Fang Chen, and Arcot Sowmya. A weakly supervised approach for object detection based on soft-label boosting. In *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*, pages 331–338. IEEE, 2013.

[17] Robert E. Schapire, Marie Rochery, Mazin Rahim, and Narendra Gupta. Boosting with prior knowledge for call classification. *Speech and Audio Processing, IEEE Transactions on*, 13(2):174–181, 2005.

[18] Riwal Lefort, Ronan Fablet, and Jean-Marc Boucher. Weakly supervised classification of objects in images using soft random forests. In *Computer Vision-ECCV 2010*, pages 185–198. Springer, 2010.

[19] Jeff A Bilmes et al. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. *International Computer Science Institute*, 4(510):126, 1998.

[20] Paula Gabbert. Decision trees from uncertain learning sets. In *Systems, Man, and Cybernetics, 1994. Humans, Information and Technology., 1994 IEEE International Conference on*, volume 1, pages 913–918. IEEE, 1994.

[21] Takuya Kobayashi, Akinori Hidaka, and Takio Kurita. Selection of histograms of oriented gradients features for pedestrian detection. In *Neural Information Processing*, pages 598–607. Springer, 2008.

[22] Chengbin Zeng and Huadong Ma. Robust head-shoulder detection by pca-based multilevel hog-lbp detector for people counting. In *Pattern Recognition (ICPR), 2010 International Conference on*, pages 2069–2072. IEEE, 2010.

[23] Arthur P Dempster, Nan M Laird, Donald B Rubin, et al. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal statistical Society*, 39(1):1–38, 1977.

[24] Richard A Redner and Homer F Walker. Mixture densities, maximum likelihood and the em algorithm. *SIAM review*, 26(2):195–239, 1984.

[25] Yoav Freund and Robert E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37, 1995.

[26] David Gerónimo, Angel D Sappa, Daniel Ponsa, and Antonio M López. 2d-3d-based on-board pedestrian detection system. *Computer Vision and Image Understanding*, 114(5):583–595, 2010.

[27] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.