Boosting with Side Information

Jixu Chen[1], Xiaoming Liu[2], and Siwei Lyu[3]

[1] GE Global Research, Niskayuna, NY, chenji@ge.com
[2] Michigan State University, East lansing, MI, liuxm@cse.msu.edu
[3] University at Albany, SUNY, Albany, NY, slyu@albany.edu

Abstract. In many problems of machine learning and computer vision, there exists side information, i.e., information contained in the training data and not available in the testing phase. This motivates the recent development of a new learning approach known as *learning with side information* that aims to incorporate side information for improved learning algorithms. In this work, we describe a new training method of boosting classifiers that uses side information, which we term as *AdaBoost+*. In particular, AdaBoost+ employs a novel classification label imputation method to construct extra weak classifiers from the available information that simulate the performance of better weak classifiers obtained from the features in side information. We apply our method to two problems, namely handwritten digit recognition and facial expression recognition from low resolution images, where it demonstrates its effectiveness in classification performance.

1 Introduction

Classification plays a central role in the solutions of many computer vision problems. A classifier is a parametric function that takes input $\mathbf{x} \in \mathcal{R}^d$ and predicts its class label $y \in \{0, 1, \cdots, C\}$. The conventional approach to obtain a classifier is with supervised learning that uses a training set of data $(\mathbf{x}_1, \cdots, \mathbf{x}_N)$ and their corresponding class labels $(y_1, \cdots, y_N)$. The semi-supervised learning approach [6] relaxes on the requirement of a fully labelled training set, and can be used to learn a classifier with a mixture of labelled and unlabelled training data.

However, in many practical applications, there may be extra sources of information other than the training data and class labels, and to which we only have access when training the classifier. For instance, in face recognition, besides images of faces and their corresponding subjects' identities, we can have other information in training, such as the age, gender, skin color, etc. In this work, we call such features as *side information*, and correspondingly, we call features that are present in both training and testing as *available information*.

Side information arises because some aspects of a classification problem cannot be specified via the class labels and the training data[4]. Therefore, it should be differentiated from missing features, which are subsets of composing categories of data that are unavailable in the testing and/or training sets [24, 1, 14,

[4] In the most general sense, labels of the training set can also be regarded as side information.

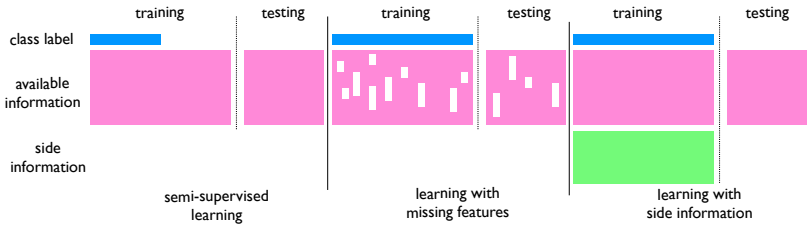


Fig. 1. Graphical illustrations of settings of different learning paradigms.

27, 10]. More importantly, typical techniques addressing missing features such as *data imputation*, where the missing features are replaced with their predictions from the available information, may not be used to handle side information. Predicting side information could be a more difficult task than directly learning the classifier from available information, as in the cases when side information corresponds to more complex and higher dimensional features. On the other hand, predicting side information may not be of particular relevance to the classification task, which could be regarded as unnecessary and wasteful in practical applications.

Incorporating side information into the construction of the classifier may be beneficial because it provides alternative means for the system designer to input more prior knowledge into the learned classifier. Yet, because side information is not present when the classifier is deployed to classify a previously unseen datum, it has been largely ignored in the current practice of automatic learning of classifiers. This motivates the recent development of a new learning approach known as *learning with side information* (also known as learning with privileged information) [33, 32, 25]. The general idea of learning with side information is to use a hypothetical classifier built with available and side information together as an oracle of performance upper-bound to guild the training of the classifier using only available information. Recent results in learning theory [21, 25] have suggested that using side information can improve the learning rate in the training of the classifier, and may be particularly useful when the training data set is relatively small. In practice, learning with side information has shown promising performance gain when it is incorporated in the training of the SVM classifiers (an algorithm known as SVM+) and applied to handwritten digit recognition, time series prediction, and protein classification [33, 32]. Learning with side information can also find abundant applications in computer vision. For instance, to obtain an effective classifier that can recognize faces in low resolution still images from surveillance cameras, we can use high resolution images (e.g., mug-shots) or video clips as side information.

Here, we explore the potential of combining side information to improve boosting classifiers, which we call *AdaBoost+*. Boosting has been widely used in many computer vision problems (such as face or expression recognition), because it is constructed from simple base classifiers, e.g. decision stumps or linear cuts, which facilitates extracting interpretable rules from the decision boundary. In particular, AdaBoost+ does not attempt to impute the side information with

available information, which as we have mentioned, could lead to a more difficult problem. Instead, it uses *classification label imputation* to construct extra weak classifiers from the available information that simulate the performance of better weak classifiers obtained from features in side information. We apply AdaBoost+ to two problems, namely handwritten digit recognition and facial expression recognition from low resolution images, where it demonstrates its effectiveness in classification performance.

2 Related Works

In machine learning, side information has been employed in the context of unsupervised learning, in particular, in distance metric learning [34], constrained clustering [4] and similarity kernel learning [17]. In the unsupervised learning context, the side information is usually cast in the form of pairwise constraints, while few work discusses using side information in the form of extra features.

In comparison, there are fewer works discussing the use of side information for supervised learning. The recent work of Vapnik [33, 25] has shown that a learning algorithm trained with the help of both side information (SVM+), as well as the available information, provides improved performance over a machine trained only on the available information (SVM). In the SVM+ algorithm, side information is used to predict the optimal slack variables in the SVM objective function. In statistical analysis, missing features usually cause biased estimation of statistics or model parameters. To improve the estimation, we can either delete the examples with missing features (list-wise deletion, pairwise deletion) or impute the missing features [24, 13, 27] (regression imputation, hot deck imputation, multiple imputation and EM). A few works focusing on the problem of missing features only in testing data set [14, 10, 27]. In particular, if the missing feature indices are known, it shows that simply imputing the missing features will give an even worse result than using only the observed features [27]. Our work demonstrates that an improved performance can be achieved by using the missing feature in testing as side-information in the training phase.

In computer vision, the attribute-based object recognition [11, 18] uses discrete attribute as side information. These attributes are semantic visual qualities of objects, such as 'red', 'striped', 'wood', etc. Most of the attribute-based methods are similar to imputation, i.e., the missing attributes in the testing data is imputed by attribute classifiers. These methods are different from our work in two significant ways. First, they only focus on low-dimensional discrete attribute labels, while our work is applicable to more general types of side information including discrete labels (Sec. 5.1), high-dimensional continuous data (Sec. 5.2), time series data or 3D structure of proteins [33]. Second, they use the attributes as mid-level features and have to learn attribute classifiers, which may or may not be relevant to the actual classification task. In contrast, we use side-information as extra relevant information with an explicit goal of improving the final classification performance.

There are some prior works on boosting with side information [36] for medical applications where the side information is the actual domain knowledge of a particular application, rather than the data in our work. Side information is also

different from the prior work using 'side' data [29, 7], which performs unsupervised learning using a data set with relative information and an auxiliary 'side' data set with irrelevant information. The irrelative information in the 'side' data is minimized in learning.

3 Side Information and Classification Performance

We start with a simple analysis on the relationship between side information and classification performance. For simplicity, we consider binary classification, and a classifier is defined as $f(\tilde{\mathbf{x}}) : \mathcal{X} \mapsto \{-1, +1\}$, which maps data to the binary class $\tilde{\mathbf{x}} \in \mathcal{X}$ to the corresponding binary class labels. The optimal performance of a binary classification problem is lower bounded by the Bayesian error, $P(y \neq f(\tilde{\mathbf{x}})|\tilde{\mathbf{x}})$ [5], which itself is lower bounded by the conditional entropy $H(y|\tilde{\mathbf{x}})$, a direct corollary of applying the Fano's inequality [9]. In other words, the probability of incorrectly predicting y based on information from $\tilde{\mathbf{x}}$ is constrained by the remaining uncertainty of $H(y|\tilde{\mathbf{x}})$ about y when $\tilde{\mathbf{x}}$ is known. In this sense, $H(y|\tilde{\mathbf{x}})$ provides a general metric of classification performance independent of the particular choice of f.

Next, we consider data as a composite of the available information $\mathbf{x}$ and the side information $\mathbf{z}$, $\tilde{\mathbf{x}} = (\mathbf{x}, \mathbf{z})$. Then, we have $H(y|\mathbf{x}, \mathbf{z}) \leq H(y|\mathbf{x})$, since $H(y|\mathbf{x}) - H(y|\mathbf{x}, \mathbf{z}) = I(y, \mathbf{z}|\mathbf{x})$, where the conditional mutual information $I(y, \mathbf{z}|\mathbf{x})$ is always non-negative [9]. Therefore, including side information may lead to the reduction of the conditional uncertainty about the class label y given input data, which could correspond to a lower Bayesian error. Furthermore, the optimal classification performance when the side information is available in training but withheld in testing is expected to be sandwiched between of the optimal performances of the two other ideal classifiers, one constructed with only available information and the other constructed with both available and side information. In particular, when the performance gap between the two ideal classifiers is significant, the latter can be used as an oracle to guide the training for more effective use of the side information.

So far, our analysis only pertains to the ideal case when we have access to the joint distribution of data and class labels, corresponding to an infinitely large labelled training data set. When finite training data sets are used, recent results in learning theory [21, 25] show that under some fairly general conditions, learning with side information can lead to provably faster learning rate (in some cases, it gives rise to exponential improvement in the learning rates [21]), i.e., more reduction in classification error per additional training example. This makes learning with side information particularly useful in classification problems with high dimensional data and limited training data sets.

4 Boosting with Side Information

In this section, we will describe our algorithm to incorporates side information into an AdaBoost classifier, named *AdaBoost+*. The basic idea of boosting is to produce a strong classifier by linearly combining a set of weak classifiers, each may be learned from one single feature, in an iterative learning procedure. During each iteration, one optimal weak classifier will be selected based on the

Algorithm 1 Boosting with Side Information (Adaboost+)

input: N training samples of $\tilde{\mathbf{x}}_i = (\mathbf{x}_i, \mathbf{z}_i) \in \mathcal{R}^{D+E}$, with available information $\{\mathbf{x}_i\}_{i=1..N}$ and side information $\{\mathbf{z}_i\}_{i=1..N}$ and class labels $\{y_i\}_{i=1..N}$.
output: Boosted classifier $f^T(\mathbf{x}) = \sum_{t=1}^{T} \alpha^t h^t(\mathbf{x})$
Initialization: Set initial classifier : $f^0(\mathbf{x}) = 0$.
for $t = 1$ to T **do**
  Using $\{\tilde{\mathbf{x}}_i\}_{i=1..N}$ as training data, build a *Type-I* classifier for each feature.
  Select top K weak classifiers $h_I^t(\tilde{\mathbf{x}})$ and the corresponding α^t that minimize the training error, Eq.(1).
  for $k = 1$ to K **do**
    if $h_I^t(\tilde{\mathbf{x}})$ is built with a feature from side information **then**
      Train a regressor $\mathcal{R}^t$ from available information to this weak classifier.
      Build a *Type-II* weak classifier $h_{II}^t(\mathbf{x})$ on this regressor (Eq. 4).
      Replace the original *Type-I* classifier with $h_{II}^t(\mathbf{x})$.
      Update the corresponding α^t (Eq. 3).
      Update the corresponding training error.
    else
      Keep the original *Type-I* weak classifier $h_I^t(\mathbf{x})$ and its corresponding error.
    end if
  end for
  Add the updated top one weak classifier into the final boosting classifier $f^t(\mathbf{x}) = f^{t-1}(\mathbf{x}) + \alpha^t h^t(\mathbf{x})$.
end for

strong classifier up to the previous iteration, from one feature in the feature pool. In particular, for our scenario of learning with side information, the feature pool includes features from both available information and side information. In this paper, we assume that during the iterative process, there is at least one weak classifier learned from side information will be considered as optimal for a certain iteration, comparing to all weak classifiers from available information.

If this assumption cannot be satisfied, i.e., the weak classifier from available information always leads to a lower error rate on the training data than that of the side information while combining with the existing strong classifier, our method will be degenerated into the conventional AdaBoost. When this assumption can be satisfied, we assert that there is a possibility for a boosted classifier constructed with features from both available and side information to achieve better classification performance than a classifier built with available information alone. One simple reason for this assertion is that the particular feature of side information chosen during the iteration may serve as an oracle to guide the boosting on how to use the available information more effectively. In our paper such guidance is achieved by classification label imputation, which uses an auxiliary regressor from available information to simulate the binary classification outputs of the weak classifier from side information.

4.1 Algorithm

Let us first introduce the basic notations in the subsequent description. The training data set of the boosting classifier includes corresponding examples of

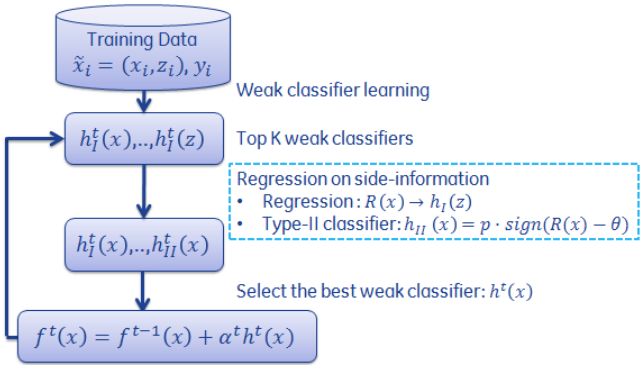


Fig. 2. The algorithm diagram of boosting with side information (Adaboost+).

available information $\{\mathbf{x}_i\}_{i=1..N}$ and side information $\{\mathbf{z}_i\}_{i=1..N}$, with binary class labels $\{y_i\}_{i=1..N}$. Note that for ith training sample, its available information, $\mathbf{x}_i \in \mathcal{R}^D$, and side information, $\mathbf{z}_i \in \mathcal{R}^E$ can be from different feature spaces. We also denote the complete training data by combining available information and side information as: $\tilde{\mathbf{x}}_i = (\mathbf{x}_i, \mathbf{z}_i) \in \mathcal{R}^{D+E}$, where $\mathbf{x}_i = (x_i^1, ..., x_i^D)$ and $\mathbf{z}_i = (z_i^1, ..., z_i^E)$, respectively. From the training data, our goal is to learn a boosted strong classifier $f^T(\mathbf{x}) = \sum_{t=1}^{T} \alpha^t h^t(\mathbf{x})$.

This algorithm is summarized in Algorithm 1 and the algorithm diagram is shown in Figure 2. As in the general framework of boosting, our algorithm starts with an initial classifier $f^0(\mathbf{x}) = 0$. Then from the whole set of training data, $\{\tilde{\mathbf{x}}_i\}_{i=1..N}$, we learn a *Type-I* classifier, $h_I(\mathbf{x}) = p \cdot (sign)(x^d - \theta)$, for each feature corresponding to available information and side information. Specifically, $h_I(\mathbf{x})$ is a decision stump, where d is the index of the selected feature, θ is the threshold, and $p \in \{+1, -1\}$ is the polarity of the weak classifier. We then select the top K weak classifiers $h_I^t(\tilde{\mathbf{x}})$ and their corresponding α^t that minimize the training error in Eq. 1.

$$E = \sum_{i=1}^{N} e\left(y_i, f^{t-1}(\mathbf{x}_i) + \alpha^t h_I^t(\tilde{\mathbf{x}}_i)\right), \tag{1}$$

where $e(\cdot, \cdot)$ is the error function in comparing the class label with the classifier output. The choice of function $e(\cdot)$ differs among different boosting algorithms, and we adopt the one for Adaboost [12], which is given by:

$$e\left(y_i, f(\mathbf{x}_i)\right) = \exp(-y_i f(\mathbf{x}_i)). \tag{2}$$

Given a weak classifier $h^t(\mathbf{x})$, its corresponding α^t to minimize Eq. 1 is computed as:

$$\alpha^t = 0.5 \log\left((1 - \epsilon^t)/\epsilon^t\right), \tag{3}$$

where $\epsilon^t = \frac{\sum_{i=1}^{N} w_i \cdot [y_i \neq h^t(\mathbf{x}_i)]}{\sum_{i=1}^{N} w_i}$ is the weighted error of this classifier, and the weight $w_i = \exp(-y_i f^{t-1}(\mathbf{x}_i))$ depends on the previous strong classifier.

In the second step of the training, if any one of the top K *Type-I* classifiers is from side information, we replace it with a regressor from available information that optimally match the binary outputs of that *Type-I* weak classifier. In actual

testing when side information is not available, such regressor can mimic the effect of side information since the outputs of the *Type-I* weak classifier have been approximated from this regressor. Specifically, for each one of top K *Type-I* weak classifier that is from side information, we train an auxiliary regressor, $R^t(\mathbf{x}) \rightarrow h_I^t(\mathbf{z})$, that uses available information to predict the binary output of this weak classifier. Note that our method is different from using a regression to predict all the features in side information from available information, which is the typical practice of data imputation. In contract, we call our method as classification label imputation, which is applicable to side information with much higher dimensions than the available information (see the experiment in Sec. 5.2). Based on the regressor's output, we construct a *Type-II weak classifier*, which is defined as:

$$h_{II}(\mathbf{x}) = p \cdot (sign)[R(\mathbf{x}) - \theta]. \tag{4}$$

In principle, any regression method can be used in our algorithm. Considering the efficiency and effectiveness, we use the Gaussian Process Regression (GPR) [26] in our experiment. GPR is a non-parametric regression which assumes the regression function following gaussian process. Given the training data of input/output pairs $\{\mathbf{x}_i, y_i\}_{i=1..N}$, the regression output of a new input $\mathbf{x}^*$ can be derived as a Gaussian distribution, and we use its mean as the regression output:

$$y* = R(\mathbf{x}_*) = \mathbf{K}_*^T(\mathbf{K})^{-1}\mathbf{Y}, \tag{5}$$

where $\mathbf{K}$ is a $N \times N$ matrix whose entries are the kernel functions (RBF kernel [26] in our experiment) of the training data: $k(\mathbf{x}_i, \mathbf{x}_j)$. $\mathbf{K}_*$ and $\mathbf{Y}$ are N dimensional vectors whose entries are $k(\mathbf{x}_i, \mathbf{x}^*)$ and y_i respectively. To train a GPR, we only need to estimate the covariance matrix $\mathbf{K}$ on the training data.

After the top K classifiers are updated, we select the top one weak classifier into the final classifier for the current iteration. Notice that if all the top classifiers are selected from available information, they do not need to update and our algorithm degenerates to conventional Adaboost. We choose $K > 1$, because the imperfection of the regression may change the ranking of the top K weak classifiers. Since very low ranked classifiers has little chance to be top ranked, we set $K = 5$ in our experiments, which seems working well in practices.

5 Experiments

We now demonstrate the effectiveness of our algorithm on two applications, handwritten digit recognition and facial expression recognition from low resolution images, each of which uses different types of side information. Note that in both cases, rather than developing methods of performance better than the state-of-the-art, our emphasis here is to showcase that the boosting with side information can lead to better performance in comparison to the conventional boosting algorithm.

5.1 Handwritten Digits Recognition

In the first set of experiments, we consider the problem of handwritten digit recognition from images. As shown in Fig. 3, we use the data set provided by [33],

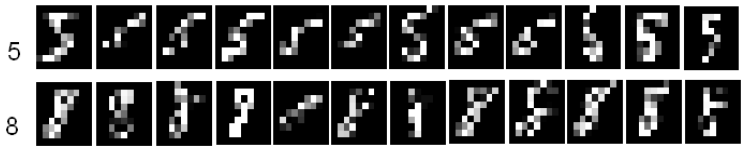


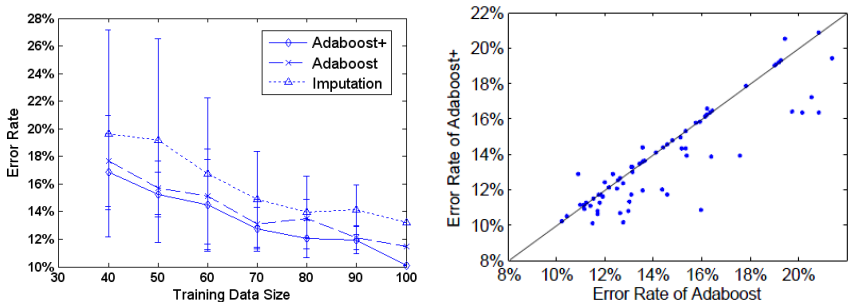
Fig. 3. Examples of handwritten digits in 10×10image.

which consists of low-resolution images (10×10 pixels) of handwritten digits 5 and 8. As in [33], we use 100 images (maximally available to the public) as training set and 1866 images as the test set. Furthermore, for every training image, 21 holistic (poetic) descriptions are provided by an independent expert. Each description can be translated to a discrete value. Some of these descriptions (with range of possible values) are: two-part-ness (0-5); tilting to the right (0-3); uniformity (0-3), etc. Note that these descriptions are not available for the test images and thus are treated as side information in our method.

Our goal is to learn a boosted classifier using the available information (100 dimensional vectors of the vectorized pixel values) and the side information (21 dimensional vectors from the quantized textual descriptions). The classification errors using different training data sizes are shown in Fig. 4(a). For each training data size smaller than 100, 12 different subsets are randomly selected from the training data. Thus, we perform train and test 12 times and report the average and standard deviation of these 12 testing errors. We also compare with the regression imputation [24], i.e., the missing side information in the testing data is recovered by the predicted value from a regression, and the augmented features with both side information and available information are used for classification. Here, we learn a Gaussian process regression to predict the side information from the available information. When the training data size is 100 (only test once), the classification error is 13.23%, 11.47% and 10.08% for imputation, Adaboost and Adaboost+ respectively. We can see that the imputation result is even worse than Adaboost. This is consistent with the previous imputation result [24] and the attribute-based methods [30, 11] which show that using side information/attributes in an unselective manner does not necessarily improve the classification. In [11], the classification performance can be improved only when good discriminative attributes are carefully selected.

Using the side information, Adaboost+ outperforms Adaboost in general. The scatter plot in the top panel of Fig. 4 compares the Adaboost and Adaboost+ testing performances of the classifiers trained using each randomly selected training subset. We can see that Adaboost+ outperforms Adaboost in most tests.

A direct comparison with SVM+ [33] is difficult. On one hand, SVM and boosting are two very different classification techniques. On the other hand, [33] used 4000 images as an extra validation set for tuning the parameters in SVM and SVM+. This validation set does not have accompanying textual descriptions as side information. In order to have a fair comparison with SVM+, we also use this



(a) (b)

Fig. 4. (a) Performance comparison with various training data sizes. (b) Scatter plot of error rates of 73 tests. Adaboost+ is equal or better than Adaboost in 63 tests.

validation set in our boosted classifier. Specifically, after applying Algorithm 1 on the training set, we obtain a set of selected features (some features are from regression). We then extract these features for the validation data, from which we build a Adaboost classifier. Given the same training, validation and testing data as [33], the classification results of Adaboost+ and SVM+ are shown in Fig. 5. These results suggest that it is easy to incorporate the validation data without side information into the Adaboost+ algorithm, and compared to Fig. 4(a) the variance of the results reduces significantly due to the large amount of validation data. More importantly, note that the side information improves the classification performances of both SVM and Adaboost. For instance, using 90 training samples, the average error rate of SVM is reduced from 8.2% to 6.7% (SVM+), and the average error rate of Adaboost is reduced from 6.5% to 5.3% (Adaboost+). Lastly, we notice that the errors of Adaboost and Adaboost+ decrease given larger training data size, but this decrease is not as fast as those of SVM-based classifiers. The reason is that Adaboost directly combines a large amount of validation data (4000 samples) with training data (100 samples) in training, while SVM uses training data to learn support vectors and uses validation data to tune the hyper-parameter. Therefore, the training data size plays a more important role in SVM. It is difficult to directly compare SVM+ and Adaboost+ because the ways of using validation data are different. However, we can see Adaboost+ is achieving better performance than SVM+ with the same training and validation data, and the gap between Adaboost and Adaboost+ is significant.

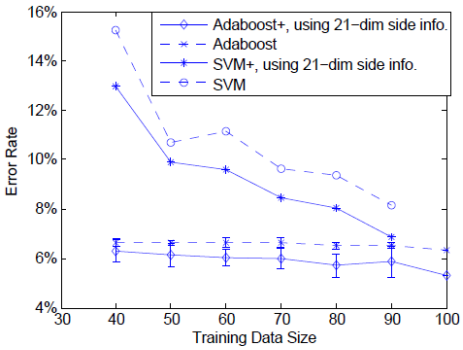


Fig. 5. Digit recognition with validation data. SVM and SVM+ results are from [32].

Fig. 6. HR (a) and LR (b) images of six prototype emotions in the Cohn-Kanade database (©Jeffrey Cohn) . LR is re-scaled to the same size (128×128) as HR.

5.2 Facial Expression Recognition

Facial expression provides critical cues for the internal emotions of a human subject, and expression recognition is an important problem in computer vision. Most of the current works [35, 3, 8] assumes high resolution face images. However, many practical scenarios call for expression recognition based on low-resolution images, for instance when the images are captured with low resolution web-cams or surveillance cameras. In this section, we consider a system of facial expression recognition that uses low-resolution images, but is trained with high resolution images as side information.

We used the Cohn-Kanade (CK) Facial Expression database [20], which is considered today's de-facto standard for comparative studies in facial expression analysis. This database consists of 100 subjects who are instructed to perform a series of 23 facial displays, six of which are prototype emotions, i.e., anger, disgust, fear, joy, sadness and surprise. For our experiment, we select 300 image sequences from the database. The length of the sequences varies between 9 and 60 frames. Each of the sequences consists of expressions from neutral to one of the six prototype emotions. In the original CK database, only the peak expression of each sequence is labeled. Here, we perform recognition on every frame based on the frame-by-frame expression label in [31]. We randomly separate the subjects into two folds, and perform two-fold cross-validation to make sure the training and testing images are from different subjects.

Given the original image resolution of 640×490, we simulate the low-resolution (LR) image recognition by normalizing the face region to a 16×16 small image (Fig. 6). Previous experiment [22] on CK database has shown that the recognition rate decreases significantly in low-resolution images. To address this problem, previous work focus on extracting robust features from LR images [23, 28]. Here, we suppose the corresponding high-resolution (HR) face images are available as training data, and we propose to use these HR images as side information to learn a better classifier. This application is related to the popular super-resolution approaches [16, 15, 19] in face recognition, which uses HR images in training data to learn a mapping from LR image to HR images. The test LR image is mapped to a HR image for recognition. However, our approach is fundamentally different from those methods because we extract complimentary and discriminative side information from HR images rather than reconstructing the HR images themselves. Specifically, we extract 128×128 HR face images

Table 1. Area under ROC for six expressions.

Expression	LR (Adaboost)	HR	LR interpolation	LR+ (Adaboost+)
Anger	0.786	0.910	0.705	0.815
Disgust	0.855	0.919	0.833	0.897
Fear	0.688	0.732	0.634	0.721
Joy	0.954	0.959	0.521	0.944
Sadness	0.832	0.886	0.802	0.875
Surprise	0.976	0.992	0.773	0.980
Average	0.849	0.899	0.711	0.872

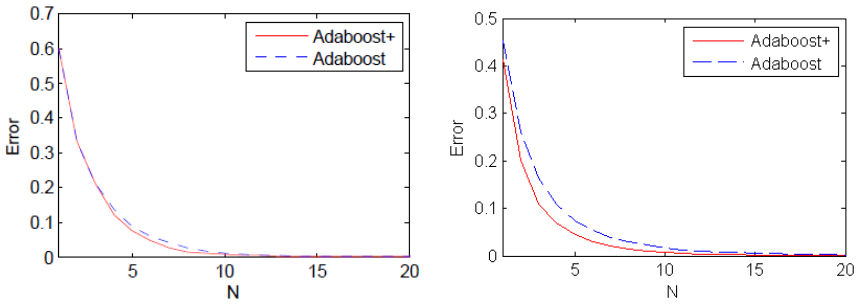
from the training image sequences (Fig. 6). Local binary pattern (LBP) feature [2] is extracted from the HR and LR images respectively. We chose LBP due to its demonstrated effectiveness for expression recognition in low-resolution image [28]. Similar to [28], we divide the HR image into 64 regions of 16×16 pixels [5] and apply the $LBP^{u2}_{8,2}$ operator to extract a 59-bin histogram from each region. Original LBP combines these histograms to form a long feature vector (59×64) for the face image. For efficiency, we map the 59 dimension vector to a two-dimensional LDA space. Thus, the final feature vector has $2 \times 64 = 128$ dimensions. Similarly, the feature from the side information (one HR image) also has 128 dimensions.

Using Adaboost or Adaboost+ classifier, we compare the result of different settings as follows:

- LR recognition: Adaboost is trained on LR and tested on LR. This is the baseline in LR-based expression recognition.
- HR recognition: Adaboost is trained on HR and tested on HR. This ideal case provides an upper bound for LR-based recognition.
- LR-interpolate recognition: Adaboost is trained on HR and tested on re-scaled LR though interpolation. This is an naive approach to test on LR by using HR as training data.
- LR+ recognition: Adaboost+ is trained on LR with HR as side information and tested on LR.

In [28, 22], only the average recognition rates are reported without false alarm. To provide a comprehensive evaluation, the area under ROC (AUC) for six expressions are shown in Table 1. As expected, the performance of HR recognition is the upper-bound (average AUC=0.899). Using HR as side information, the LR recognition rate can be improved towards the upper-bound (average AUC is improved from 0.849 to 0.872). We observe that this improvement is more significant for subtle expressions, i.e., anger, disgust, fear and sadness. But for expressions such as surprise and joy, the improvement is marginal. We also notice that the naive approach of interpolating the LR image does not work well. One possible reason is that the classifier trained on HR relies more on high frequency features, which are not presented in interpolated LR images.

[5] Before feature extraction, LR image is firstly re-scaled to 128×128 through cubic interpolation.



(a) Digit recognition (b) Expression recognition

Fig. 7. Comparison of training error for the top N features.

5.3 Side Information in Different Applications

In different applications, side information are extracted from different sources, have different meanings and have difference influences on the final classifier. Here, we compare the side information in the above two applications.

In AdaBoost+, a feature from side information is selected only when it can further decrease the error function in Eq. 2. We show the average error function of the top N features in Fig. 7. As expected, the error of Adaboost+ decreases faster than Adaboost indicating a more economic use of weak classifiers based on effective side information. We also observe that the side information (HR image) in expression recognition is more effective. Compared to the poetic descriptions in digit recognition, HR image provides more information to the available feature (LR image). This can also be observed in the number of features selected from side information. In the top 20 features, average 5.5 features are selected from side information in expression recognition, while only 0.71 features are selected in digit recognition. These results are consistent with the theoretical analysis [21, 25], which suggests that using side information can accelerate the convergence of learning errors.

6 Conclusions

Side information that is contained in the training data but not available in testing exists in many problems in machine learning and computer vision. This motivates the recent development of a new learning approach known as *learning with side information* that aims to incorporate side information for improved training of learning algorithms. In this work, we describe a new training method of boosting classifiers that uses side information, known as AdaBoost+. In particular, we propose a novel *classification label imputation* method to construct extra weak classifiers from the available information that simulate the performance of better weak classifiers obtained from features in side information. The experiments on two vision problems demonstrate the effectiveness of our method in improving classification performance compared to AdaBoost classifier trained without using side information.

There are several important extensions of the current work that we would like to further pursue. First, note that in the most general sense, learning with

side information is similar to semi-supervised learning and learning with missing features, where a classifier is obtained with partially missing information. In semi-supervised learning, this corresponds to missing labels, and in learning with side information, this corresponds to the absence of side information in the testing phase. Therefore, we are interested in extending the current work to a more general setting that combines missing label, feature and side information. This would provide a unification of all these different learning algorithms, and can also find many applications in machine learning and computer vision. Second, we believe that many problems in computer vision can benefit from the incorporation of side information, and we like to extend the boosting with side information framework to other computer vision problems.

References

1. Subutai Ahmad and Volker Tresp. Some solutions to the missing feature problem in vision. In *NIPS*, 1998.
2. T. Ahonen, A. Hadid, and M. Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE T-PAMI*, 28(12):2037 –2041, 2006.
3. M.S. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, and J. Movellan. Recognizing facial expression: machine learning and application to spontaneous behavior. In *CVPR*, 2005.
4. Sugato Basu, Mikhail Bilenko, and Raymond J. Mooney. A probabilistic framework for semisupervised clustering. In *KDD*, 2004.
5. Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1 edition, 2007.
6. O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.
7. Gal Chechik and Naftali Tishby. Extracting relevant structures with side information. In *NIPS*, 2002.
8. Ira Cohen, Nicu Sebe, Ashutosh Garg, Lawrence S. Chen, and Thomas S. Huang. Facial expression recognition from video sequences: temporal and static modeling. *CVIU*, 91(1-2):160 – 187, 2003.
9. T. Cover and J. Thomas. *Elements of Information Theory*. Wiley-Interscience, 2nd edition, 2006.
10. O. Dekel, O. Shamir, and L. Xiao. Learning to classify with missing and corrupted features. *Machine learning*, 2010.
11. Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. In *CVPR*, 2009.
12. Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28(2):337–407, 2000.
13. Pedro J. Garcia-Laencina, Jose-Luis Sancho-Gomez, and Anibal R. Figueiras-Vidal. Pattern classification with missing data: a review. *Neural Comput. Appl.*, 19(2):263–282, 2010.
14. A. Globerson and S. Roweis. Nightmare at test time: robust learning by feature deletion. In *ICML*, 2006.
15. B.K. Gunturk, A.U. Batur, Y. Altunbasak, III Hayes, M.H., and R.M. Mersereau. Eigenface-domain super-resolution for face recognition. *IEEE T-IP*, 12(5):597 – 606, 2003.

16. P.H. Hennings-Yeomans, S. Baker, and B.V.K.V. Kumar. Simultaneous super-resolution and feature extraction for recognition of low-resolution faces. In *CVPR*, 2008.
17. Steven C. H. Hoi, Rong Jin, and Michael R. Lyu. Learning nonparametric kernel matrices from pairwise constraints. In *ICML*, 2007.
18. Sung Ju Hwang, Fei Sha, and Kristen Grauman. Sharing features between objects and their attributes. In *CVPR*, 2011.
19. Kui Jia and Shaogang Gong. Multi-modal tensor face for simultaneous super-resolution and recognition. In *ICCV*, 2005.
20. Takeo Kanade, Jeffrey F. Cohn, and Yingli Tian. Comprehensive database for facial expression analysis. In *FG*, 2000.
21. P. Kuusela and D. Ocone. Learning with side information: PAC learning bounds. *J. Comput. Syst. Sci.*, 68(3):521–545, 2004.
22. Ying li Tian. Evaluation of face resolution for expression analysis. In *CVPRW*, 2004.
23. Shu Liao, Wei Fan, A.C.S. Chung, and Dit-Yan Yeung. Facial expression recognition using advanced local binary patterns, tsallis entropies and global appearance features. In *ICIP*, 2006.
24. Roderick J. A. Little and Donald B. Rubin. *Statistical Analysis with Missing Data*. John Wiley and Sons, Inc., 1987.
25. D. Pechyony and V. Vapnik. On the theory of learning with privileged information. In *NIPS*, 2010.
26. Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2005.
27. Maytal Saar-Tsechansky and Forster Provost. Handling missing values when applying classification model. *Journal of Machine Learning Research*, 8:1623–1657, 2007.
28. Caifeng Shan, Shaogang Gong, and P.W. McOwan. Robust facial expression recognition using local binary patterns. In *ICIP*, 2005.
29. Amnon Shashua and Lior Wolf. Kernel feature selection with side data using a spectral approach. In *ECCV*, 2004.
30. Yu Su, Moray Allen, and Frederic Jurie. Improving object classification using semantic attributes. In *BMVC*, 2010.
31. Yan Tong, Wenhui Liao, and Qiang Ji. Facial action unit recognition by exploiting their dynamic and semantic relationships. *IEEE T-PAMI*, 29(10):1683 –1699, 2007.
32. V. Vapnik and A. Vashist. A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(5-6):544–557, 2009.
33. Vladimir Vapnik, Akshay Vashist, and Natalya Pavlovitch. Learning using hidden information (learning with teacher). In *IJCNN*, 2009.
34. Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart. Russell. Stuart. distance metric learning, with application to clustering with side-information. In *NIPS*, 2003.
35. Zhihong Zeng, Maja Pantic, Glenn I. Roisman, and Thomas S. Huang. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *IEEE T-PAMI*, 31(1):39–58, 2009.
36. Lei Zhang, Dimitris Samaras, Dardo Tomasi, Nora Volkow, and Rita Goldstein. Machine learning for clinical diagnosis from functional magnetic resonance imaging. In *CVPR*, 2005.