Improving adaboost for classification on small training sample sets with active learning

Lei Wang
Nanyang Technological University, Singapore, leiw@uow.edu.au

Xuchun Li
Nanyang Technological University, Singapore

Eric Sung
Nanyang Technological University, Singapore

Follow this and additional works at: https://ro.uow.edu.au/eispapers

Part of the Engineering Commons, and the Science and Technology Studies Commons

Recommended Citation
Wang, Lei; Li, Xuchun; and Sung, Eric, "Improving adaboost for classification on small training sample sets with active learning" (2004). Faculty of Engineering and Information Sciences - Papers: Part A. 667. https://ro.uow.edu.au/eispapers/667

Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au
Improving adaboost for classification on small training sample sets with active learning

Abstract
Recently, AdaBoost has been widely used in many computer vision applications and has shown promising results. However, it is also observed that its classification performance is often poor when the size of the training sample set is small. In certain situations, there may be many unlabelled samples available and labelling them is costly and time-consuming. Thus it is desirable to pick a few good samples to be labelled. The key is how. In this paper, we integrate active learning with AdaBoost to attack this problem. The principle idea is to select the next unlabelled sample based on it being at the minimum distance from the optimal AdaBoost hyperplane derived from the current set of labelled samples. We prove via version space concept that this selection strategy yields the fastest expected learning rate. Experimental results on both artificial and standard databases demonstrate the effectiveness of our proposed method.

Keywords
sets, active, learning, adaboost, improving, small, sample, training, classification

Disciplines
Engineering | Science and Technology Studies

Publication Details

This conference paper is available at Research Online: https://ro.uow.edu.au/eispapers/667
Abstract

Recently, AdaBoost has been widely used in many computer vision applications and has shown promising results. However, it is also observed that its classification performance is often poor when the size of the training sample set is small. In certain situations, there may be many unlabelled samples available and labelling them is costly and time-consuming. Thus it is desirable to pick a few good samples to be labelled. The key is how. In this paper, we integrate active learning with AdaBoost to attack this problem. The principle idea is to select the next unlabelled sample base on it being at the minimum distance from the optimal AdaBoost hyperplane derived from the current set of labelled samples. We prove via version space concept that this selection strategy yields the fastest expected learning rate. Experimental results on both artificial and standard databases demonstrate the effectiveness of our proposed method.

1 Introduction

Boosting is a general method for improving the classification accuracy of any classification algorithm [5]. The original idea of boosting was introduced by Kearns and Valiant [6]. By using PAC (probably approximately correct) learning theory [10], Boosting directly converts a weak learning model, which performs just slightly better than randomly guessing, into a strong learning model that can be arbitrarily accurate. In 1995, AdaBoost [4] was introduced. In AdaBoost, after each weak learning iteration, misclassified training samples are adaptively given high weights in the next iteration. This forces the next weak learner to focus more on the misclassified training data. Because of the good classification performance of AdaBoost, it is widely used in many computer vision problems and some promising results have been obtained. For example, it has been used for categorization tasks [3], image retrieval [2] and so on. It is known that the good classification performance of AdaBoost is based on the sufficient number of labelled samples. However, in real computer vision applications, collecting enough labelled samples is often costly and time-consuming. In some applications, there may be plentiful unlabeled samples. For example, a hospital may have collected a vast amount of medical images but the medical specialists do not have time to analyse all of them. So in this situation, it is beneficial if one can automate the selection of a small number of samples for the specialist to label. These selected samples must be optimally picked so as to achieve the fastest learning rate. As a result, the number of labelled samples is commonly limited and this constrains the classification performance of AdaBoost. Hence, it is particularly valuable to investigate how to efficiently use the given labelled samples to obtain a classification accuracy as high as possible. In this paper, this problem is focused.

Active learning is a mechanism which aims to optimize the classification performance while minimizing the number of needed labelled samples. Its procedure can be briefly described as follows: given an initially small labelled sample set and an unlabelled sample pool, a classifier is trained by using this labelled sample set. After that, the most informative unlabelled sample to classification is selected from the unlabelled sample pool by using a given selection strategy. Then the teacher is queried to label this sample. After being labelled, this sample will be added to the current training sample set to form a new set. Afterwards, a new training-selecting-querying cycle will begin. The key issue of active learning mechanism is the optimization of selection strategy for fastest learning rate.

In this paper, active learning is integrated into AdaBoost to improve AdaBoost’s classification performance on small training sample sets. Considering that Version Space is a powerful tool to analyze a learning process, it is adopted in this paper to analyze active learning-based AdaBoost in theoretical sense. We gave a geometrical interpretation of AdaBoost in a version space, and following this direction, we proposed the optimal selection strategy. To the best of our knowledge, there are few papers which analyze active learning-based AdaBoost in the theoretical sense or
optimize the selection strategy. Our experimental results on both artificial and real databases demonstrate that the proposed method can effectively improve the classification performance of AdaBoost, especially on the small training sample sets.

2 Background

In [9], Schapire et al. used a margin as a measure to analyze the classification performance of AdaBoost. The margin of a sample to a classifier is defined as the product of this sample’s label and this classifier’s output for this sample. They gave a classification bound based on the margin and proved that larger margins on the training sample set could lead to higher classification accuracy. In [1], Ratsch and Warmuth proved that AdaBoost could only obtain a fairly large margin, instead of the maximal margin. Hence, they introduced AdaBoost algorithm where \( \rho \) is a pre-specified target margin and each weak learner is assumed to have a training error smaller than \( \left( \frac{1}{2} - \frac{1}{2} \rho \right) \). By iteratively adapting \( \rho \), they used Marginal AdaBoost to achieve the maximum margin. Here, we focus on two classes classification problem. As we will be using these algorithms, a summary of them are described as follows.

Algorithm: The AdaBoost algorithm

1. **Input:** a set of training samples with label \( \{(x_1, y_1), \ldots, (x_N, y_N)\} \), weak learning algorithm, the number of iterations \( T \), target margin \( \rho \).
2. **Initialize:** the weight of training samples: \( w_1^n = 1/N \), for all \( i = 1, \ldots, N \).
3. Do for \( t = 1, \ldots, T \):
   - Train the weighted training sample set to obtain the weak learner \( h_t : \{+1, -1\} \).
   - Calculate the training error of \( h_t \): \( \varepsilon_t = \frac{1}{N} \sum_{n=1}^{N} w_n^t \) \( \{y_n \neq h_t(x_n)\} \). Break if \( \varepsilon_t = 0 \) or \( \varepsilon_t \geq \frac{1}{2} \) and set \( T = t - 1 \).
   - Calculate \( \gamma_t \): \( \gamma_t = \frac{1}{N} \sum_{n=1}^{N} w_n^t y_n h_t(x_n) \) \( \{\gamma_t \text{ is the edge of } h_t\} \).
   - Set weight of weak learner \( h_t \): \( \alpha_t = \frac{1}{2} \log \frac{1 + \gamma_t}{1 - \gamma_t} - \frac{1}{2} \log \frac{1 + \rho}{1 - \rho} \).
   - Update weights of training samples: \( w_n^{t+1} = \frac{w_n^t \exp(-\alpha_t y_n h_t(x_n))}{D_t} \), where \( D_t \) is a normalization constant, and \( \sum_{n=1}^{N} w_n^{t+1} = 1 \).
4. **Output:**
   - Normalize weights of weak learners: \( \alpha_t = \frac{\alpha_t}{\sum_{r=1}^{T} \alpha_r} \).
   - Output the label: \( f(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \), let \( A = [\alpha_1, \alpha_2, \ldots, \alpha_T] \), then \( A \) satisfies \( \|A\|_1 = 1 \).

By iteratively adapting \( \rho \), Marginal AdaBoost is used to maximize the margin and it has the similar convergence rates as AdaBoost. It can be seen as the following optimization problem [8]:

\[
\begin{align*}
\max_{\alpha \geq 0} & \quad (\rho) \\
\text{subject to} & \quad \rho_n(\alpha) \geq \rho, \quad \rho_n(\alpha) = y_n \sum_{t=1}^{T} \alpha_t h_t(x_n), \quad \text{and} \quad \|A\|_1 = 1
\end{align*}
\]

(1)

It means that Marginal AdaBoost finds the classifier which has the maximal margin. AdaBoost [4] can be considered as a special case of the Marginal AdaBoost when the target margin, \( \rho \), is set to 0. Because the Marginal AdaBoost really achieves the maximal margin and it has been reported [1] that it can give better classification result than AdaBoost, we use Marginal AdaBoost in the following theoretical analysis. Our experimental results will show that the selection strategy derived from the theoretical analysis is also effective for AdaBoost which is commonly used in practice.

3 The proposed method

Our proposal is to apply AdaBoost to obtain the optimal hyperplane for the present set of labelled samples. Then the next sample is chosen to meet the minimum margin criterion. This means that the unlabelled sample that is closest to this optimal hyperplane is chosen. We show that our method yields the fastest expected learning rate for sequential selection. In the following analysis, feature space and version space are firstly defined. Then, the geometrical interpretation of AdaBoost in the version space is given. Finally, the optimal selection strategy is proposed. We will show that, by employing the proposed selection strategy, better classification result can be obtained when a small training sample set is encountered. In addition, active learning-based AdaBoost is called active AdaBoost in short for convenience.

3.1 The version space

Let \( h_1, h_2, \ldots, h_T \) be the \( T \) weak learners used in AdaBoost, and \( H(x) = [h_1(x), h_2(x), \ldots, h_T(x)] \) denotes the output of these \( T \) weak learners. This implies that, for a sample, \( x \), a mapping, \( H \), is constructed from the input space, \( \mathcal{I} \), where the \( x \) lies, to a feature space, \( \mathcal{H} \):

\[
H(x) = \begin{bmatrix} h_1(x) \\ h_2(x) \\ \vdots \\ h_T(x) \end{bmatrix} : \mathcal{I} \to \mathcal{H}.
\]

(2)

In this way, the decision function of AdaBoost can be rewritten as

\[
f(x) = \sum_{t=1}^{T} \alpha_t h_t(x) = A^\top H(x)
\]

(3)

where \( A \) is defined as \( A = [\alpha_1, \alpha_2, \ldots, \alpha_T]^\top \), and it is a vector in the coefficient space, \( \mathcal{A} \). It can be found that
\( f(x) = 0 \) represents a hyperplane in the feature space, \( \mathcal{H} \), with the coefficient vector, \( \mathbf{A} \). The dimensions of both \( \mathcal{H} \) and \( \mathcal{A} \) are equal to the number of weak learners, \( T \).

The version space of AdaBoost is explored as follows. Given a set of training samples, \( I \), the version space can be viewed as a set of hypotheses each of which correctly classifies all the training samples [7]. In the case of AdaBoost, the version space, \( V_I \), is defined as a set of linear classifiers, \( f \), in \( \mathcal{H} \) under the condition that each classifier can correctly classify all the training samples in \( I \):

\[
V_I = \{ f | f(x) = \mathbf{A}^\top \mathbf{H}(x), y f(x) > 0, \forall x \in I, y \in \{+1, -1\}, \mathbf{A} \in \mathcal{A}, \mathbf{H}(x) \in \mathcal{H} \} \tag{4}
\]

It is known that each classifier \( f \) in version space \( V_I \) corresponds to a coefficient vector \( \mathbf{A} \). Hence, the version space, \( V_I \), can be redefined based on the coefficient vector \( \mathbf{A} \), instead of \( f \), under the condition that each of the corresponding classifier can correctly classify all the training samples:

\[
V_I = \{ \mathbf{A} | \| \mathbf{A} \|_1 = 1, y \| \mathbf{A}^\top \mathbf{H}(x) \| > 0, \forall x \in I, y \in O, \mathbf{A} \in \mathcal{A}, \mathbf{H}(x) \in \mathcal{H} \} \tag{5}
\]

Recall that \( \| \mathbf{A} \|_1 \) is equal to 1 in AdaBoost. Similarly, we define \( V_I \), as the version space corresponding to the training sample set, \( I_t \), formed in the \( i \)-th learning cycle.

To ensure the existence of the version space, an assumption has to be taken that the two classes are linearly separable over the \( \mathbf{H}(x) \) (\( x \in I_t \)) in \( \mathcal{H} \). According to the VC-dimension theory [11], when the samples are independently and identically distributed, a linear classifier can always be found in an \( n \)-dimensional space which correctly classifies \( (n+1) \) samples irrespective of how they are labeled. Hence, in the case of AdaBoost, provided you can find \( n+1 \) independent classifiers, when the dimension of \( \mathcal{H} \) is larger than the number of training samples minus one, there must exist a linear classifier in \( \mathcal{H} \) which can correctly separate all the training samples. Namely, the existence of version space can be ensured. Because this paper focus on small training sample sets, the dimension of \( \mathcal{H} \), which is equal to the number of weak learners, \( T \), can be easily larger than the number of training samples. Besides this, the most effective learning in active learning commonly happens at the first several cycles where the training sample is less. Due to these reasons, we can guarantee the existence of the version space here.

A duality can be found between the feature space, \( \mathcal{H} \), and the coefficient space, \( \mathcal{A} \). As mentioned before, \( f(x) = \mathbf{A}^\top \mathbf{H}(x) = 0 \) can be seen as a hyperplane in \( \mathcal{H} \), where \( \mathbf{A} \) is the coefficient vector and \( \mathbf{H}(x) \) is the variable; Meanwhile, it can be found that it is also a hyperplane in \( \mathcal{A} \) where \( \mathbf{H}(x) \) is the coefficient vector while \( \mathbf{A} \) is the variable. Hence, a training sample, which corresponds to a point in \( \mathcal{H} \), also corresponds to a hyperplane in \( \mathcal{A} \), and vice versa.

\[ f(x) = 0 \]

**Figure 1.** A version space in 3D space. (This figure illustrates the version space in a 3D space. Version space is a region on the 2D triangle plane, while the 2D plane intersecting with this triangle denotes the hyperplane induced by a sample.)

### 3.2 The geometrical interpretation of AdaBoost in a version space

Recall that, in AdaBoost, the coefficient vector, \( \mathbf{A} \), is constrained by \( \| \mathbf{A} \|_1 = 1 \). This equality represents a hyperplane in coefficient space \( \mathcal{A} \). Besides this, each component of \( \mathbf{A} \), \( \alpha_t (t = 1, 2, \cdots, T) \), should be a positive number. From equation (5), we know that version space of AdaBoost is a collection of coefficient vectors. Hence, geometrically, the version space is a connected region on the hyperplane \( \| \mathbf{A} \|_1 = 1 \), and this region is restricted in the first quadrant of the coefficient space. After mapping (equation (2)), a sample \( x \) converts to \( \mathbf{H}(x) \). In the coefficient space, because \( \mathbf{H}(x) \) is a vector composed of 1 and -1, \( \mathbf{H}(x) \) lies on the vertex of a hypercube. The center of this hypercube is the origin of the space \( \mathcal{A} \) and its edge length is 2. Therefore, the coefficient vector \( \mathbf{H}(x) \) of the hyperplane induced by \( x \) is a vector pointing to one of the vertexes of the above hypercube. Figure 1 illustrates the version space in a 3D space. Version space is a region on the 2D triangle plane, while the 2D plane intersecting with this triangle denotes the hyperplane induced by a sample. The 3D cube with the dashed line denotes the hypercube mentioned above. Before giving the geometrical interpretation of AdaBoost in a version space, the definition of margin has to be introduced to facilitate the analysis. In AdaBoost, given a set of training samples \( \{(x_1, y_1), \cdots, (x_N, y_N)\} \), the margin of a sample, \( x_k \), to a decision hyperplane, \( f = 0 \), is defined as

\[
\rho_f(x_k) = y_k \frac{f(x_k)}{\| \mathbf{A} \|_1} \tag{6}
\]

The margin of \( f \) is then defined to be the minimal value among the margins of all the training samples to \( f \) as

\[
\rho_f = \min_{x_k \in I} \rho_f(x_k) \tag{7}
\]
Due to the duality between the feature space and the coefficient space, in $H$, the sample, $x_k$, through the mapping will induce a hyperplane in $A$. Only those $A$s satisfying $y_k H(x_k)^\top A > 0$ will be favored because they correspond to those decision hyperplanes in $H$ which can correctly classify $x_k$. The distance of such a favored point, $A$, from the hyperplane induced by $x_k$ can be computed as

$$d_{x_k}(A) = \frac{H(x_k)^\top A}{\|H(x_k)\|_2} = \frac{f(x_k)^\top A}{\sqrt{T}}$$

(8)

$$= \frac{y_k f(x_k)}{\|A\|_1} = \frac{1}{\sqrt{T}} \rho_f(x_k)$$

This indicates that $d_{x_k}(A)$ in $A$ is proportional to the margin, $\rho_f(x_k)$, in $H$. In $A$ space, we further define the minimal distance, $D(A)$, between this $A$ to these hyperplanes induced by all the samples in a training set, $I$, as follows.

$$D(A) = \min_{x_k \in I} [d_{x_k}(A)] = \frac{1}{\sqrt{T}} \min_{x_k \in I} [\rho_f(x_k)]$$

(9)

Hence, it can be found that $D(A)$ is proportional to $\rho_f$ defined in equation (7). It is known that, based on a training sample set, $I$, marginal AdaBoost finds the classifier, $f^*_i$, which has the maximal margin. Let $A^*_i$ be the coefficient vector of $f^*_i$. Based on the above relationship between $D(A)$ and $\rho_f$, it can be shown that the marginal AdaBoost correspondingly finds the coefficient vector, $A^*_i$, which has the maximal distance, $D(A^*_i)$, in the coefficient space, $A$. Now, the geometrical interpretation of $A^*_i$ in a version space can be described as follows. The training sample set induces a set of hyperplanes in the coefficient space. In this space, there exist some hyperspheres which do not intersect with any hyperplane above. $A^*_i$ lies in the current version space and is the center of the hyperspheres whose radius is maximal among all the hyperspheres. The radius can be computed to be $\frac{1}{\sqrt{T}} \rho_f$ (see equation (9)). Furthermore, those hyperplanes which touch this hypersphere have the minimal distance to $A^*_i$ and they correspond to the training samples with the minimal margin in the feature space.

### 3.3 The optimal selection strategy for active AdaBoost

Let $f^*$ be the optimal classifier which gives the minimal test error on all possible training sample sets. It is the classifier that active learning aims to approach in the minimal number of learning cycles. The coefficient vector of $f^*$ is denoted as $A^*$. Recall that, in the $i$-th active learning cycle, the training sample set is $I_i$. The classifier obtained by training AdaBoost on $I_i$ is denoted as $f^*_{i}$ and its coefficient vector is denoted as $A^*_i$. Because $A^*$ is the coefficient vector corresponding to the optimal classifier, $f^*$, it always lie in the series of version spaces formed in active learning cycles in general. Furthermore, the Marginal AdaBoost is purposely forced to achieve zero-training error in each active learning cycle. In this paper, this means that, in the $i$-th learning cycle, $f^*_i$ will correctly classify the training sample in $I_i$. Therefore, $A^*_i$ will lie in the corresponding version space, $V_{I_i}$. Hence, $A^*_i$ can well approach to $A^*$ if the size of $V_{I_i}$ is small enough. This implies that the version space has to be greedily reduced to make $A^*_i$ converge to $A^*$ as fast as possible.

Recall that $V_{I_i}$ denote the version space given $I_i$, and let $S_{V_{I_i}}$ be the size of $V_{I_i}$. The optimal selection strategy should have the version space reduced as much as possible. Hence, the expectation of the size of the next version space, $V_{I_{i+1}}$, in the $(i+1)$-th learning cycle, $E(S_{V_{I_{i+1}}})$, should be minimized after the selected unlabelled sample is labelled. Let $x_q (x_q \not\in I_i)$ denote the query sample, and its label is $y_q$. To correctly classify the sample, $x_q$, it has to satisfied that $y_q H(x_q)^\top A > 0$. In the coefficient space, this formula can be rewritten as $y_q H(x_q)^\top A > 0$. As mentioned before, only those $A$s satisfying $y_q H(x_q)^\top A > 0$ will be favored. This means that, by adding the query sample, $x_q$, to the current training sample set, a part of version space will be excluded from the current version space and a new smaller version space will be formed. The optimal query sample, $x^*$, which minimizes the expected version space can be expressed as

$$x^* = \arg \min_{x_q \not\in I_i} E(S_{V_{I_{i+1}}})$$

(10)

In the coefficient space, $V_{I_i}$ is partitioned into two sub-regions by the hyperplane induced by the selected unlabelled sample. Let $R_{i,j}$ and $R_{i,j+1}$ be the two sub-regions, and $S_{R_{i,j}}$ and $S_{R_{i,j+1}}$ denote the sizes of them. It can be shown that $V_{I_i} = \bigcup_{j=1}^{N_i} R_{i,j}$ and $S_{V_{I_i}} = \sum_{j=1}^{N_i} S_{R_{i,j}}$. Because the optimal normal vector, $A^*$ always lies in one of the two sub-regions, the version space $V_{I_{i+1}}$ will be the sub-region $R_{i,j+1}$ if and only if $A^*$ lies in this sub-region. In this way, the expectation of the size of $V_{I_{i+1}}$ can be expressed as

$$E(S_{V_{I_{i+1}}}) = \sum_{j=1}^{N_i} [S_{R_{i,j}} P(A^* \in R_{i,j})] = \sum_{j=1}^{N_i} [S_{R_{i,j}} \int_{R_{i,j}} p(A^* | V_{I_i}) dA^*]$$

(11)

where $P(A^* \in R_{i,j})$ is the probability of $A^*$ falling into $R_{i,j}$. Considering that $A^*$ can lie anywhere in the version space $V_{I_i}$ with equal probability, it can be shown that $p(A^* | V_{I_i}) = \frac{1}{S_{V_{I_i}}}$. Thus,

$$\int_{R_{i,j}} p(A^* | V_{I_i}) dA^* = \frac{1}{S_{V_{I_i}}} \int_{R_{i,j}} dA^* = \frac{S_{R_{i,j}}}{S_{V_{I_i}}}$$

(12)
Hence, equation (11) becomes

\[
E(S_{V_{i+1}}) = \frac{1}{S_{V_i}} \left( \sum_{j=1}^{2} S_{R_{i,j}}^2 \right) \\
\geq \frac{1}{S_{V_i}} \left( \sum_{j=1}^{2} S_{R_{i,j}} \right)^2 = \frac{S_{V_i}}{2}
\]

(13)

Based on the Cauchy inequality, Cauchy inequality above becomes an equality if and only if \( S_{R_{i,1}} = S_{R_{i,2}} = \frac{S_{V_i}}{2} \). This result indicates that the optimal selection strategy is to select such an unlabelled sample, which corresponds to the hyperplane that can partition the current version space into equally-sized two sub-regions.

According to the above geometrical interpretation of AdaBoost in a version space that \( A_i^* \) is the center of the hypersphere, \( A_i^* \) will be on the geometrical center of the current version space if this space is symmetrical. Hence, selecting the sample inducing the hyperplane closest to \( A_i^* \) can still approximately bisect the current version space. It has been obtained that, in the coefficient space, the distance of \( A_i^* \) from the hyperplane corresponding to the training sample, \( x_k \), is \( \frac{1}{\sqrt{f_i^*}} \rho f_i^*(x_k) \) (see equation (8)), which is proportional to the margin of \( x_k \) to \( f_i^* \). If we want to select an unlabelled sample which corresponds to the hyperplane closest to \( A_i^* \), we can select the one whose margin to \( f_i^* \) is the minimal in the feature space.

In summary, it can be concluded that by selecting the sample with the minimal margin in each active learning cycle, the version space can be reduced quickly. The algorithm of the proposed method is described as follows.

1. **Input:** the initial training sample set \( I = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\} \), the query sample set \( Q = \{(x_1, y_1), (x_2, y_2), \cdots, (x_q, y_q)\} \)

2. **Initial training:** use AdaBoost to train the initial training sample set and obtain a classifier, \( f \).

3. **Repeat:**
   (a) Compute: compute the margin values of all unlabelled samples with \( f \).
   (b) Query: select the sample, \( x_q^* \), which can greedily decrease the expectation of the next version space. Here we select the sample corresponding to the minimal margin:

   \[
   \rho f(x_q^*) = \arg \min_{x \in Q} \rho f(x)
   \]

   (14)

   (c) Train: Add the query sample to the current training sample set and train AdaBoost on this new set.

4. **Output:** classifier, \( f \), after learning cycles finish.

### 4 Experimental Results

An artificial database, an OCR database and two UCI Repository databases are used in the experiment. Each of the databases has two classes. The artificial database is generated by the *majority rule* (the label of a sample is 1 if the majority of bits in this sample are +1 and it is -1 if the majority of bits in this sample are -1). Training samples and query samples are from 200 artificial samples while the test samples are another 200 artificial samples with the same underlying distribution. The OCR database is USPS database of handwritten postal codes, and one-against-all classification with respect to digit 0. When the digit is 0, the label is 1, otherwise, the label is -1. Training samples and query samples are from 7,291 digits and test samples are from 2,007 digits. Each digit consists of a 16×16 grey level image and the grey values are scaled to \([1, -1]\). The ionosphere database and banana database of UCI Repository database are also used. In the banana database, the number of training samples and query samples is 400 while the number of test sample is 4900. In the ionosphere database, the number of training samples and query samples is 200 while the number of test samples is 200. In the experiments, the weak learner of AdaBoost is RBF network. The number of RBF’s centers is set to 10, and the total number of weak learners is fixed to 200.

### 4.1 Experimental procedure

The following procedure is for conducting experiments to evaluate the proposed method on the four databases.

1. \( k_0 \) training samples are randomly chosen as the initial training samples before active learning starts. Among the \( k_0 \) sample, at least, there are one positive and one negative training samples.
2. Train AdaBoost with the \( k_0 \) initial training samples and a classifier, \( f \), is obtained.
3. Compute the margins of all the unlabelled samples by using \( f \). For active AdaBoost, select the sample with respect to the minimal margin as the query sample. After being labelled according to the grand truth, this sample will be added to the current training sample set to form a new set. For AdaBoost, randomly select a query sample from all the unlabelled samples. Label this sample according to the ground truth and add it to the current training set. Correspondingly another new training sample set is obtained.
4. Based on the two newly formed training sample sets, train two AdaBoost classifiers, respectively.
5. Redo step 3 to step 4. Until active learning stops.
6. To accumulate statistics, redo step 1 to step 5 one hundred times and calculate the average classification errors of the two methods. Plot the error values for AdaBoost and active AdaBoost and compare them.
4.2 The experimental results and discussions

Figure 2(a) shows the comparison between AdaBoost and Active AdaBoost on the artificial database. The comparison is to show how "intelligent" selection of the next sample can outperform one which is randomly selected. Our comparison is based on $m+n$ samples. The number of initially labelled samples is $m$ for both active AdaBoost and AdaBoost. For the $n$ samples, these are chosen by our algorithm whereas for AdaBoost, these are randomly selected from the unlabelled pool. The X-coordinate is the number of query samples and the Y-coordinate is the test error values. The number of initial training samples is 20 and in each active learning cycle, one query sample is selected. From this figure, we can see that to achieve the same accuracy, for example, 6% test error, nearly 60 training samples are saved by using active AdaBoost. Based on the same number of training samples, active AdaBoost obtains the maximal improvement of 6.53% over AdaBoost. Similar results are also obtained from the other databases with the same number of initial training samples. In Figure 2(b), the USPS database, at most 1.7% improvement is achieved. In Figure 2(c), the Ionosphere database, the maximal improvement on the test error reaches 3.16%. In Figure 2(d), the improvement of 4.5% is achieved on Banana database. Hence, the experiment results on both artificial databases and real databases demonstrate the effectiveness of the proposed method. Based on a given number of training data, active AdaBoost can provide better classification performance than AdaBoost. On a large scale, the fewer the number of initial training samples, the higher the improvement obtained.

5 Conclusion

This paper proposed active learning-based AdaBoost to accelerate the learning rate of AdaBoost on small training sample set. Based on version space, the proposed method is analyzed in theoretical sense and an optimized selection strategy is derived. The experimental results on both artificial and real databases demonstrated that, for a given small training sample set, the proposed method can provide better classification performance than the original AdaBoost. The smaller the size of training sample set, the more significant the improvement on classification accuracy. The real applications in computer vision can benefit from the proposed method, because it can effectively reduce the demands of labelled samples while maintaining a satisfactory classification performance.

References