Gabor-Boosting Face Recognition

Mian Zhou and Hong Wei
School of System Engineering
University of Reading
Reading RG6 6AY
United Kingdom
m.zhou, h.wei@reading.ac.uk

March 13, 2009

Abstract

In this paper, we present an accurate appearance-based face recognition algorithm with Gabor wavelet transform for feature extraction, AdaBoosting algorithm for feature selection, and Support Vector Machine (SVM) for classification. Within the AdaBoost algorithm, we design a novel weak learner - Potsu which is the only weak learner fulfilling the requirement of AdaBoost. In classification, we found that ensemble classifier made from AdaBoost may prone to overfitting, and the effects can be reduce by SVM.

Keywords: Face Recognition, Gabor Wavelets, AdaBoost, SVM

1 Introduction

In recent years, biometrics has been used to identify individuals using traits based on retina, iris, fingerprints, or face. Personal identification based on face images is of particular interest as they are non invasive and rely lessly on the cooperation of the individual.

Depending on the application context, face recognition is divided into two scenarios: face verification and face identification. In face verification, an individual who desires to be recognised claims an identity, usually through a personal identification number, an user name, or a smart card, so that face verification is to answer question - "Does the face belong to the specified person?". In face identification, the system is to establish an individual's identity without the subject to claim an identity, so that face identification is to answer question - "Whose face is this?".

The face recognition system presented in this paper is based on individual subject in a way of face verification. Face verification in this paper is to verify a claimed identity by face image of the individual, and a classifier labels the image as two classes, i.e. client or impostor [29]. A client is the given individual with claimed identity. Impostors are all other persons except the client. In this way, the face verification can be defined as client-based. By specifying a certain person, all face images belonging to the person are recognised as the client, others are recognised as impostors. In terms of machine learning, the client is considered as positive examples, and impostors are considered as negative examples. For such a two-class classification , techniques developed for object detection and face detection [25, 26] can be used.

The face recognition process consists of three phases: feature extraction, feature selection and classification. Feature extraction is to construct a high dimensional feature space from the image space. Feature selection reduces the dimensionality of the feature space, and spans a low dimensional subspace. Classification verifies a new face image as a client or an impostor. In this paper, we present a new face recognition algorithm called Gabor-Boosting, which is motivated by Boosting face detection [25] and Gabor wavelet representation. Gabor wavelet transform is used to extract Gabor wavelet features from face images. Due to very large number of features available, AdaBoost is used to select small group of features representing individual face. Finally, the verification is conducted by Support Vector Machine (SVM) classifiers with selected features. To evaluate the performance of Gabor-Boosting algorithm, the XM2VTS face database is used.

The paper is organised as follows: Section 2 describes the Gabor wavelet feature extraction; Section 3 presents the feature selection algorithm, in which a novel weak learner - Potsu is proposed; two types of classification are explored in Section 4, i.e. ensemble and SVM classification. It has shown that SVM is appropriate for classification in Gabor-Boosting; and finally Section 5 gives the conclusion.

2 Gabor Wavelet Feature Extraction

This section discusses the Gabor wavelets and how face images are represented by Gabor wavelet features. Firstly, the background of Gabor wavelets is given. Secondly, the Gabor wavelet and its parameters are defined. Thirdly, Gabor wavelet transform is illustrated and the representatives of face images - Gabor wavelet features are presented.

2.1 Background of Gabor Wavelet

Gabor wavelets exhibit desirable characteristics of spatial locality and orientation selectivity. Their kernels are similar to the two-dimensional (2-D) receptive field profiles of the mammalian cortical simple cells [8]. Gabor wavelets capture the local structure corresponding to spatial frequency[1], spatial localisation, and orientation selectivity. As a result, Gabor wavelet representation of face images is robust to variations due to illumination and facial expression changes [16, 19, 22].

Two-dimensional Gabor wavelets are first introduced into biometric research by Daugman [1] for human iris recognition. Lades et al [10] apply Gabor wavelets in face recognition using the Dynamic Link Architecture (DLA) framework. The DLA performs a flexible template comparison between Gabor wavelet representations of different face images. Wiskott et al. [27] expand on DLA by developing a Gabor wavelet-based Elastic Bunch Graph Matching (EBGM) algorithm to label and recognise human faces. Testing done on the FERET database [18]

[1]In somewhere, spatial frequency is referred as scale.

is with a high recognition rate for frontal face images. Liu and Wechsler [13] apply the Enhanced Fisher linear discriminant Model (EFM) to an augmented Gabor feature vector derived from the Gabor wavelet representation of face images. Fan et al. [2] combine Gabor wavelets and Linear Discriminate Analysis (LDA) for face recognition. In their work, a bank of well-chosen Gabor wavelets is applied to an image to construct a group of feature vectors, and then the null space-based LDA (NLDA) is performed simultaneously for each orientation of the Gabor wavelets. The results show that as more Gabor wavelets are selected and applied, higher accuracy for recognition is achieved. Liu [12] presents a Gabor-based kernel Principal Component Analysis (PCA) method by combining the Gabor wavelet representation of face images and the kernel PCA method for face recognition. Wu et al. [28] develop a method for glass detection by applying two types of wavelet features, Haar and Gabor. The glass detector is learnt by using a variant of boosting algorithm. In their results, the Gabor features perform better than the Haar features.

2.2 Gabor Wavelets

A Gabor wavelet $\psi_{\mu,\nu}(z)$ [12] is defined as

$$\psi_{\mu,\nu}(z) = \frac{\|k_{\mu,\nu}\|^2}{\sigma^2} e^{-\frac{\|k_{\mu,\nu}\|^2 \|z\|^2}{2\sigma^2}} [e^{ik_{\mu,\nu}z} - e^{-\frac{\sigma^2}{2}}] \tag{1}$$

where $z = (x, y)$ is a point with x(the horizontal coordinate) and y(the vertical coordinate). The parameters μ and ν define the orientation and scale of the Gabor kernels, $\|\cdot\|$ denotes the norm operator, and σ is related to the standard derivation of the Gaussian window in the kernel. The wave vector $k_{\mu,\nu}$ is defined as follows

$$k_{\mu,\nu} = k_\nu e^{i\phi_\mu} \tag{2}$$

where $k_\nu = \frac{k_{max}}{f^\nu}$ and $\phi_\mu = \frac{\pi\mu}{8}$ if 8 different orientations have been chosen. k_{max} is the maximum frequency, and f is the spatial frequency between kernels in the frequency domain.

The Gabor kernels in Equation (1) are all self-similar since they can be generated from one kernel (a mother wavelet) by dilation and rotation via the wave vector $k_{\mu,\nu}$. Each kernel is a product of a Gaussian envelope and a complex plane wave. The first term in the square bracket in Equation (1) determines the oscillatory part of the kernel and the second term compensates for the DC value, thus making the kernel DC-free, i.e., the integral $\int \psi_{\mu,\nu} d^2z$ vanishes. The effect of the DC term becomes negligible when the parameter σ, which determines the ratio of the Gaussian window width to the wavelength, is sufficiently large.

In our approach, five different scales and eight orientations of Gabor wavelets were used, i.e. $\nu \in \{0, \ldots, 4\}$, and $\mu \in \{0, \ldots, 7\}$. Gabor wavelets are restricted by a Gaussian envelope function with relative width $\sigma = 2\pi$ in previous work [12, 13, 27, 28]. Gabor wavelets were with $k_{max} = \frac{\pi}{2}$, and $f = \sqrt{2}$ in [12, 13, 28]. These parameters yield a Gabor wavelets family having 5 scales and 8 orientations. The real parts of the 40 Gabor wavelets are shown in Figure 1.

2.3 Gabor Wavelet Transform

The Gabor wavelet representation of an image is the convolution of the image with a family of Gabor kernels as defined by Equation (1). The convolution of

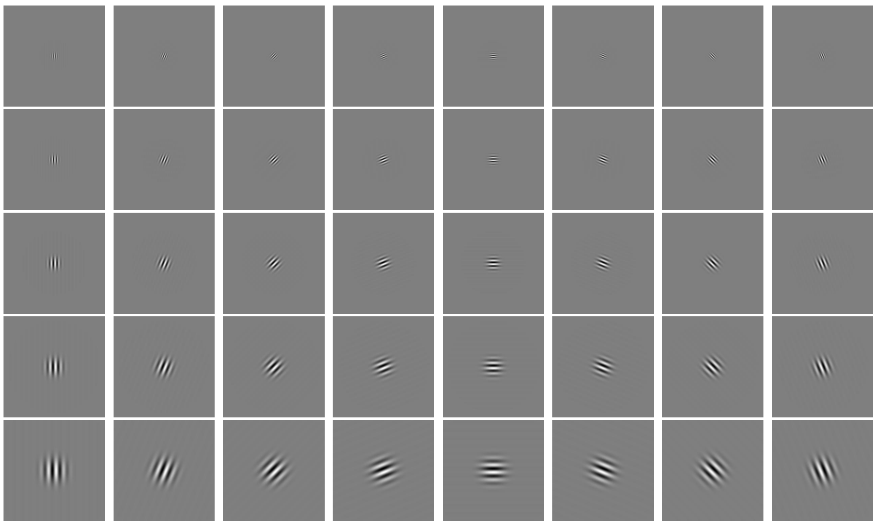


Figure 1: The real part of the 5×8 Gabor wavelets.

image I and a Gabor kernel $\psi_{\mu,\nu}$ is defined as follows

$$O_{\mu,\nu}(z) = I(z) * \psi_{\mu,\nu}(z) \tag{3}$$

The whole set of Gabor wavelet representation of the image $I(z)$ is $S = \{O_{\mu,\nu}(z) : \mu \in \{0, \ldots, 7\}, \nu \in \{0, \ldots, 4\}, z = (x, y)\}$.

The response $O_{\mu,\nu}(z)$ to each Gabor kernel is a complex function with a real part $\mathrm{Re}\{O_{\mu,\nu}(z)\}$ and an imaginary part $\mathrm{Im}\{O_{\mu,\nu}(z)\}$ illustrated as

$$O_{\mu,\nu}(z) = \mathrm{Re}\{O_{\mu,\nu}(z)\} + i\,\mathrm{Im}\{O_{\mu,\nu}(z)\}$$

The magnitude response $\|O_{\mu,\nu}(z)\|$ is expressed as

$$\|O_{\mu,\nu}(z)\| = \sqrt{\mathrm{Re}\{O_{\mu,\nu}(z)\}^2 + \mathrm{Im}\{O_{\mu,\nu}(z)\}^2} \tag{4}$$

In this paper, the magnitude response $\|O_{\mu,\nu}(z)\|$ is used to represent the features. Therefore, a Gabor wavelet feature j is configured by the three key parameters: position z, orientation μ, and scale ν, as illustrated in Equation 5

$$j(z, \mu, \nu) = \|O_{\mu,\nu}(z)\|. \tag{5}$$

For a given image $I(z)$ with $N \times M$ pixels, the number of Gabor wavelet feature representation will be $N \times M \times 40$. The feature space consists of all these features. In this case, it is 40 times larger than the original image space.

3 Feature Selection by AdaBoost

The Gabor wavelet feature representation described above resides in a very high dimensional space. It is necessary to reduce the feature space to a lower dimensional representation by feature selection. Feature selection is a technique, commonly used in machine learning and pattern recognition for reducing the dimension of a feature space. It selects a subset of relevant features from a large number of features to construct robust learning models. By removing irrelevant and redundant features from the training data, feature selection helps in improving the performance of learning models. Feature selection also helps to distinguish features which are important in learning.

In this paper, we use AdaBoost to select significant features from the pool of Gabor wavelet features, hence to reduce the dimension. Firstly, a brief introduction to AdaBoost is given. Secondly, the key element in AdaBoost - weak learner is described and discussed. Thirdly, a novel weak learner - Potsu is proposed. Finally, there is the AdaBoost feature selection algorithm and its results conducted in the XM2VTS face database.

3.1 AdaBoost

AdaBoost (an abbreviation for Adaptive Boosting) is formulated by Freund and Schapire [4] for "boosting" the accuracy of classification. It is a relatively efficient, simple, and easy learning strategy for improving the performance of learning algorithms by combining a set of rough and moderate learners (or classifiers). In AdaBoost, these learners are called "weak" learners which have low discrimination power to assign the true label correctly. Hence, AdaBoost refers to a method of combining a set of weak learners into a strong classifier which gives a high accuracy for prediction. In the term of AdaBoost, the final strong classifier is called "ensemble", which refers to a collection of statistical classifiers. By giving a set of examples with initial weights, AdaBoost builds an initial weak learner from the training data. It then focuses those examples in the training set which are misclassified. A new weak learner is then built with a modified training set which boosts the weights of these misclassified examples in the training process. Finally, an ensemble is combined linearly by these generated weak learners with corresponding weights to achieve higher recognition rate.

AdaBoost has been successfully applied to face detection by Viola and Jones in [25], in which images are firstly processed by *integral image representation* for obtaining Haar wavelet features. Due to enormous number of Haar wavelet features, a variant of AdaBoost is used to select a small set of features and train the classifier to distinguish face or non-face. For a single feature, a weak learner is built and evaluated by the training data. In the variant of AdaBoost, an iteration is an exhaustive search on the whole set of features for the most significant features for representing faces. Each iteration selects one significant feature. The final ensemble is a combination of weak learners built on these selected features. The face detector is the most rapid and accurate face detection system compared to other systems, and it pushes face detection research and development into a new era.

The successful AdaBoost application is used in the mode of two-class classification. Since the face verification scenario is introduced into a way of two-class classification, our face recognition algorithm also adopts AdaBoost to select significant features representing individuals. Unlike the face detection, the two classes are client and impostor. A client is the given individual with claimed identity. Impostors are all other persons except the client. In Gabor-Boosting, face images belonging to a client are defined as positive examples, and face images of impostors are defined as negative examples. In Gabor-Boosting, each client has its own feature representatives derived from AdaBoost training and classification mechanism.

3.2 Weak learners in AdaBoost

In AdaBoost training, the corresponding weak learner h_t for each selected feature is collected and built into an ensemble classifier. In the original AdaBoost, the ensemble classifier is

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

where, α_t is the importance factor indicating how importantly the weak learner h_t contributes to an ensemble classifier, and T is the number of iterations (the number of weak learners in ensemble). The importance of a weak learner is evaluated by

$$\alpha_t = \log \frac{1}{\beta_t} \tag{6}$$

The parameter β_t is defined as $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$, where the parameter ε_t is the classification error of the selected weak learner in each iteration. The importance is also calculated by

$$\alpha_t = \log \frac{1-\varepsilon_t}{\varepsilon_t}$$

Because the error ε_t of a weak learner from each iteration should be less than 1/2, the importance α_t will be greater than zero. The smaller the error ε_t is, the larger is the importance α_t. This indicates that a weak learner with slightly "stronger" discriminating power, plays a more important role in the ensemble classifier to make a decision.

The training of weak learners is by finding the minimum error ε with giving the training data. Such the error is evaluated by

$$\varepsilon = \sum_{i=1}^{n} \omega_{t,i} |h(x_i) - y_i|^2 \tag{7}$$

where $\omega_{t,i}$ is the weight of example x_i in the t-th iteration. Generally, training a weak learner is equivalent to finding a criterion which can predict labels of the training examples correctly. Different criteria could induce different results on labelling the examples, *i.e.*, the number of correct predictions. Normally, a classifier is required to achieve as many correct predictions as possible. However, the aim of training a weak learner in AdaBoost is to find an optimal criterion with a different requirement, which is not the maximum prediction, but a weighted prediction.

Normally different criteria are required by different classifiers, i.e. weak learners, to meet optimal classification. Linear Discriminant Analysis (LDA) classifiers require maximising between-class variance and minimising within-class variance. Naive Bayes classifiers use the method of maximum likelihood or maximise a posterior probability. Support Vector Machine (SVM) classifiers are required to maximise the margin between classes and minimise a quantity proportional to the number of mis-classification errors. k-nearest neighbour (k-NN) classifiers are based on the nearest training examples in the feature space.

Artificial Neural Network (ANN) classifiers are required to minimise the ratio between misclassified examples and total examples.

The requirement for weak learner in AdaBoost training is that the training criterion must satisfy the minimal ε. The classifiers mentioned above, such as LDA, Naive Bayes, SVM, k-NN, and ANN, do not satisfy the requirement. None of these classifiers can be directly introduced as the weak learner in the training, so that a new weak learner is needed.

3.3 Potsu weak learner

The new weak learner using an example based heuristic approach to threshold the data for AdaBoost is proposed in this section. The proposed weak learner is named **Potsu** weak learner, because it combines the concept of Perceptron with Otsu's algorithm.

To train a weak learner rapidly, it is necessary to keep the design of weak learner as simple as possible. As a simple linear classifier, *perceptron* [6] is chosen as the prototype of weak learner. A perceptron consists of observation f_j on a feature j, a threshold θ_j, and a parity p_j as below

$$h_j(x) = \begin{cases} 1 & \text{if} \quad p_j f_j(x) < p_j \theta_j \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

The observation f_j is numerical, hence the weak learner is dealing with one-dimensional data. The parity p_j indicates the direction of the inequality sign, so that there are only two options for the parity p_j - either 1 or -1. The parity of Potsu weak learner is determined by comparing the mean of all positive examples with the mean of all negative examples. It is assumed when majority of positive examples are below the threshold, the parity p_j should be equal to 1, otherwise -1.

The training of a Potsu weak learner is to label examples by finding the optimal threshold θ_j. The threshold θ should be a quantitative value within the range between the minimum and the maximum of examples' feature value $f_j(x)$. If there is a quantitative value θ' close to the threshold θ, the error ε' with respect to θ' will be also close to the minimal error ε with respect to θ. Since weak learners do not require "strong" discriminating power, it is more likely to collect a set of possible thresholds to approximate the true threshold. It is very similar to the Otsu method in which some possible thresholds are collected beforehand. These possible thresholds are called *candidates*. The classification error is calculated according to each candidate, finally an optimal threshold is found with the minimum error among these candidates. A candidate θ'_j is defined between two adjacent feature values, *i.e.* the mean of the two values. If there are n training examples for training a Potsu weak learner, it will be $n-1$ candidates available $(\theta'_1, \ldots, \theta'_{n-1})$. For each candidate, the accumulated error of Potsu weak learner h with the candidate θ'_k is calculated by

$$\varepsilon'_k = \sum_{i=1}^{n} \omega_i |h(x_i) - y_i|^2 \tag{9}$$

Once accumulated ε'_k is greater than 0.5, the calculation will stopped, because further accumulating would induce a large error which is worse than random guess [4]. Finally, the candidate with the minimum error will be selected to be the threshold θ. The number of candidates should be large, otherwise the candidates could not cover all possible values of threshold. On the other hand, the number should not be too large for computation.

The Potsu weak learner adopts a heuristic approach to find an optimal threshold reasonably close to the best possible solution. With a large number of examples, the accuracy on predicting the threshold is very high, due to a large number of candidates in selection. In addition, finding an optimal threshold in the Potsu weak learner is very fast due to the perceptron prototype. Perceptrons are especially suited for simple problems in pattern classification. The classification in perceptron is based on thresholding, which is one of the fastest algorithms for segmentation and recognition in pattern classification. The Potsu weak learners adopt a threshold selection technique. In this technique, the algorithm firstly is to collect a set of threshold candidates, and then evaluate the training performance on each candidate, and finally the candidate with the minimal error is selected as the optimal threshold. The algorithm for training a Potsu weak learner is given in Table 1. In [5, 11, 21], weak learners are modeled as the log likelihood ratio (LLR), which requires to predict probability density functions. Obviously, a Potsu weak learner is faster than those LLR based weak learners which demand the complicated prediction on the probability density function.

3.4 Boosting Feature Selection

The algorithm of AdaBoost is presented in Table 2 for Gabor wavelet feature selection. AdaBoost maintains weights ω_t as a probability distributed over training examples. The weights represent how hard the corresponding examples to be recognised correctly. At the beginning, the weights belonging to a same class (either client or impostor) are set equally. In the course of iteration, the weights are updated according to the performance of the selected weak learner. If an example is recognised incorrectly by the weak learner, the corresponding weight increases (indicating the example is hard to be recognised), and vice versa.

In the AdaBoost training, a weak learner is constructed upon a single feature, so that there are numerous weak learners representing all features. In each iteration, all weak learners are evaluated by the training set. A weak learner with the minimum error ε_t is selected, and its corresponding feature is selected as a significant feature. If the training has T iterations, T features are selected from the original feature set.

The basic idea of Boosting is to use a group of weak learners to construct a strong classifier, such that the ensemble classifier $H(x)$ is a linear combination of selected weak learners $h_t(x)$ with the corresponding importance α_t. The factor α_t specifies how important the corresponding weak learner is in making decision of $H(x)$.

The experiment is carried out on eight clients from the XM2VTS face database. A small group of features are selected for each client. For each client, the first four images are taken for training, and the other four images are for testing. For 200 clients, the training set has 800 face images, and the testing set has 1560

Table 1: The algorithm for training Potsu weak learners

```
 1: Given a feature $j$ and example $(x_1, y_1), \ldots, (x_n, y_n)$ and $y_i \in Y = \{0, 1\}$ for
    negative and positive respectively, $f_j(x)$ is the value of the feature $j$ on the
    example data $x$. For an example $(x_i, y_i)$, a weight $\omega_i$ is given.
 2: Construct a Potsu weak learner according to Equation 8.
 3: Calculate the mean feature value $\bar{\mu}_1$ on all positive examples and the mean
    feature value $\bar{\mu}_0$ on all negative examples
 4: if $\bar{\mu}_1 < \bar{\mu}_0$ then
 5:    the parity $p = 1$
 6: else
 7:    the parity $p = -1$
 8: end if
 9: Sort all $f_j(x_i) \quad i \in \{1, \ldots, n\}$ into $f'_j(x_1), \ldots, f'_j(x_n)$
10: for every two adjacent sorted examples $f'_j(x_i)$ and $f'_j(x_{i+1})$ do
11:    Compute the $i$th candidate $\theta'_i = \frac{f'_j(x_i)+f'_j(x_{i+1})}{2}$.
12: end for
13: for all $n-1$ candidates $\theta'_1, \ldots, \theta'_{n-1}$ do
14:    Suppose $\theta'_k$ as the threshold $\theta_j$ in Equation 8.
15:    Calculate the error $\varepsilon'_k = \sum_{i=1}^{n} \omega_i |h_j(x_i) - y_i|^2$ with respect to $\omega_i$.
16:    if $\varepsilon'_k > 0.5$ then
17:       reject the candidate $\theta'_k$.
18:    end if
19: end for
20: Select the corresponding candidate $\theta'_j$ with the lowest error $\varepsilon$ as the threshold
    $\theta_j$.
21: A trained Potsu weak learner is $h_j(x) = \begin{cases} 1 & \text{if} \quad pf_j(x) < p_j\theta_j \\ 0 & \text{otherwise} \end{cases}$
```

face images[2]. Given a specific client in the training set, four images belong to the client, and the other 796 images are impostors. The ratio between the number of positive and negative examples is 4 : 796. Inside the XM2VTS, an image size is of 25×24 pixels. Each image is convolved with 40 Gabor wavelets. Consequently, the total number of features for each image is $25 \times 24 \times 40 = 24,000$. In each iteration, a large number of features are rejected by control the error ε. If the error ε exceeds 0.5, the corresponding feature will be rejected and not taken into the next iteration.

After AdaBoost feature selection, a small set of features are selected as significant features for the eight clients in XM2VTS. The number of selected features from each client is different. From the 1st client to the 8th client, the number of selected features are varying from 30 to 80. Figure 2 displays these selected features projected on the corresponding client face image in the top row and the first Gabor wavelet features for each client in the bottom row. It has been noticed that these selected features are randomly distributed on faces, and they are also distributed on common facial feature areas, such as eyes, eyebrows, nose, and mouth. Interestingly, there are features located on the mole of the

[2]200 clients with 4 images and 95 impostors with 8 images

Table 2: Feature selection algorithm

```
1:  Given example $(x_1, y_1), \dots, (x_n, y_n)$, where $x_i$ is the data of the $i$th example,
    which consist of $k$ features $\{j_1, \dots, j_k\}$, and $y_i \in Y = \{0, 1\}$ for negative
    examples and positive examples respectively..
2:  Initialise the weights $\omega_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where $m$ and $l$
    are the number of negative and positive examples respectively.
3:  for $t = 1, \dots, T$ do
4:    Normalise the weights, $\omega_{t,i} \leftarrow \frac{\omega_{t,i}}{\sum_{i=1}^{n} \omega_{t,i}}$ so that $\omega_{t,i}, i = \{1, \dots, n\}$ form a
      probability distribution.
5:    for all $\{j_1, \dots, j_k\}$ do
6:      Train a weak learner $h_j$ built with one single feature $j$ with the weights
        $\omega_{t,i}$.
7:      The error is calculated as $\varepsilon_j = \sum_{i=1}^{n} \omega_{t,i} |h_j(x_i) - y_i|^2$ with respect to
        $\omega_{t,i}$
8:    end for
9:    Choose the optimal weak learner $h_t$ with the lowest error $\varepsilon_t$ from all $h_j$.
10:   Select the corresponding feature $j_t$ of the weak learner $h_t$ as a significant
      feature.
11:   Remove the feature $j_t$ from the feature set $\{j_1, \dots, j_k\}$.
12:   Update the weights $\omega_{t+1,i} = \omega_{t,i} \beta_t^{1-e_i}$, where $e_i = 0$ if example $x_i$ is
      classified correctly and $e_i = 1$ otherwise, and $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$.
13:   Choose the importance $\alpha_t = \log \frac{1}{\beta_t}$, and $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$.
14: end for
15: The ensemble classifier
```

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

4th client's face.

4 Classification

After feature selection, an ensemble classifier is spontaneously reminded as the classification solution. In the first part of this section, the recognition results from the ensemble solution is presented, and exhibits with overfitting. Hence, in the second part of this section, an alternative classifier - SVM is used to reduce overfitting. It shows that SVM is appropriate for Gabor-Boosting face recognition algorithm.

4.1 Ensemble classification

In the XM2VTS face database, the features are selected and the ensemble classifiers are built for each client. The classification results of the ensemble classifier H from the first client to the eighth client are shown in Table 3. All ensemble classifiers have the best performance on the training set, where all classifiers

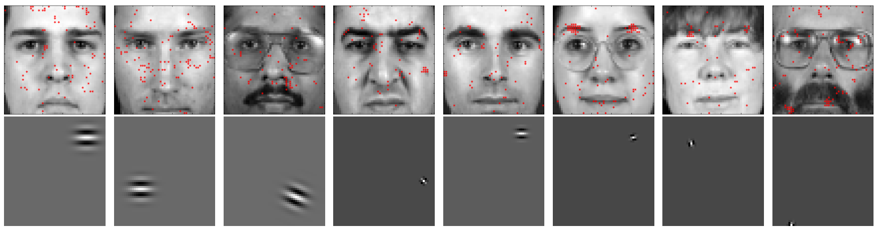


Figure 2: The selected features on face images and the first selected Gabor wavelet feature.

Table 3: The classification performance of AdaBoost Classifier

Client No	Number of Features	Training Set (%)		Testing Set (%)	
		$\mathcal{FN}$	$\mathcal{FP}$	$\mathcal{FN}$	$\mathcal{FP}$
1	48	0.0	0.0	100	0.0
2	74	0.0	0.0	100	0.0
3	37	0.0	0.0	100	0.0
4	44	0.0	0.0	100	0.0
5	48	0.0	0.0	75	0.0
6	57	0.0	0.0	100	0.0
7	40	0.0	0.0	100	0.0
8	75	0.0	0.0	100	0.0

have achieved zero error, *i.e.*, no mis-classification. Both false positive rate $\mathcal{FP}$ and the false negative rate $\mathcal{FN}$ are equal to zero so that the ensemble classifiers have achieved the perfect results in the training set. However, in the testing set, the ensemble classifiers reveal poor classification performance on positive examples, i.e. high $\mathcal{FN}$. Most positive examples are misclassified except the 5th client, but all negative examples are classified correctly.

Experimental results suggest the performance of the ensemble classifier on the training set does not represent its overall performance. The ensemble classifier is trained to perform perfectly on the training set, but reveals poor performance on other data sets, *e.g.*, the testing set. This phenomenon is so called *overfitting*. The overfitting is a wide research topic in pattern recognition, machine learning, data mining, etc. There are many issues that lead to overfitting. In this case, the issues causing the overfitting are categorised into two main sources: inferior training data and classification algorithm.

The inferior training data may lead to overfitting. Inside this source, the reason is **curse of dimensionality**. The overfitting in this case is partly due to the small number of positive examples in the training set. Generally, the number of training examples is related to the number of features used in classification. In [9], it is stated that the number of training examples per class should be at least ten times the number of features. If the number of training examples does not satisfy this requirement, the performance of classification will be degraded. This phenomenon is called "curse of dimensionality". The number of features used in the classification is around 30 to 80. The number of training examples per class should be around 300 to 800. In this case, the number of training examples of positive class is only four, which is far less than the required number.

A phenomenon that AdaBoost is prone to overfit has been observed in [7, 15]. In [17], it concludes that ensemble classifiers in AdaBoost may suffer from overfitting in the presence of noise which may explain some of the decreases in performance. Inside the source, there are two issues pertinent to overfitting. The first issue is that the AdaBoost training over-emphasises noisy examples, and the second issue is due to the weighted voting used by ensemble classifiers. Freund and Schapire [3] suggested that the poor performance of AdaBoost ensemble classifier results from overfitting the training dataset since later training may be over emphasising examples which are actually noise. In AdaBoost, an example with a higher weight is more emphasised in the training than other examples with lower weights. After each iteration of AdaBoost, the weights of some mis-classified examples are increased. In the next iteration, the training will place more emphasise on these examples which are often referred to as "hard" examples. However, if these "hard" examples include high level noise, the AdaBoost training learns the noise rather than the nature of the data itself. For a problem with noise, focusing on misclassified examples may cause the ensemble classifier to focus on boosting the margins of noise, but not on boosting the margins of examples. In fact, the weight updating misleads the AdaBoost training in overall classification. In [17], noises from different levels are added into data sets, and the results demonstrate the error rate of the AdaBoost may indeed increase as the number of weak learners increases. In [20], it suggests that later weak learners focus on increasing the margins for examples. Early stop is useful in AdaBoost training for the purpose of increasing performance. Within the ensemble classifier, each weak learner is associated with an importance α_t, which is also considered as a weight for that weak learner. Hence, the output of ensemble classifier is the combination of weighted voting from all weak learners. Previous work [23] has shown how different weight voting schemes can generally be resilient or lead to overfitting. In the work, optimising the weights is the method of choice for achieving minimal generalisation error. In general, uniformly distributed weights may cause the overfitting, while with optimised weights the ensemble classification is insensitive to overfitting.

4.2 SVM classification

Since AdaBoost is prone to the overfitting, an alternative classifier - SVM is used to classify the examples. In this section, an SVM classifier is trained with the given selected features on the training set.

SVMs are a collection of learning approaches used for generalised linear or non-linear classification. SVMs are to minimise the classification error and maximise the geometric margin, so that SVMs are also known as maximum margin classifiers. For two-class classification, it is assumed that there is a hyperplane (also called a *decision boundary*) separating the clusters of two classified data. SVMs project input vectors to a higher dimensional space where a maximal separating hyperplane is created. The separating hyperplane maximises the distance between the two classes. An assumption is made that the larger the margin or distance between two classes, the lower the generalisation error of the classifier will be. The original optimal hyperplane algorithm is a linear classifier. However, non-linear SVMs can be created by applying the kernel method [24] to maximum-margin hyperplanes. The algorithm is allowed to fit the maximum-margin hyperplane in the modified feature space. The classifier is a hyperplane in the high-dimensional feature space, which may be non-linear in the original input space. The kernels can be adopted as a polynomial kernel, a radial based kernel, or a Gaussian radial basis kernel. In this paper, a Matlab SVM toolbox - OSU Support Vector Machines (SVMs) Toolbox [14] is used to perform the task of classification.

Experiments are also carried out on the eight clients from the XM2VTS. A non-linear SVM classifier with a polynomial kernel of degree three is constructed from the 800 training examples, which are the first four images across 200 clients.

Freund and Schapire [3] suggest that AdaBoost training emphasises on noisy examples at the later iterations, so that the overfitted result is acquired at the end of training. The training of SVM is the optimisation process which is to find the primal form of the Lagrange formalism. By optimising the Lagrange function, a small group of Support Vectors (SVs) are obtained, which defines the hyperplanes between classes. The SVs are examples with non-zero Lagrange multipliers. In this case, 5 to 11 examples are taken as SVs from totally 800 training examples. The number of SVs for the eight clients is shown in Table 4. Hence, SVMs use a small number of examples to construct classifiers instead

Table 4: Number of SVs in the SVMs based on the eight clients

Client	No of SVs	No of Positive SVs	No of Negative SVs
1	7	1	6
2	11	2	9
3	5	1	4
4	10	1	9
5	9	1	8
6	7	1	6
7	7	2	5
8	5	2	3

of the whole set of examples (including the noisy examples). This optimisation process is equivalent to finding some examples, *i.e.*, SVs, which contain the lowest level of noise. By only using SVs, noisy examples are excluded in the construction of classifier. The training and classification only emphasise on those examples with the lowest level of noise, so that the overfitting caused by noisy data is reduced to the minimum level. In addition, when SVMs are designed originally, the overfitting issues are taken into account. SVMs avoid overfitting by choosing a specific hyperplane among the many that can separate the data in the feature space. SVMs find the maximum margin hyperplane, which maximises the minimum distance from the hyperplane to the closest SVs.

The training process of SVM is to find a linear discriminant function

$$g(x) = \omega^T \mathbf{x} + b \tag{10}$$

where $\mathbf{x}$ is the vector $\{x_1, x_2, \ldots, x_n\}$ from n features, ω is the weight vector $\{\omega_1, \omega_2, \ldots, \omega_n\}$ for corresponding n features, and b is a bias. The weights also imply how importantly the corresponding features play roles in classification. When these weights are normalised into a probability distribution, the weights vary dramatically. The probability distributions of weights from eight clients are shown in Figure 3. For SVMs, a few features have their weights much higher than others. This indicates that only a small number of features play more important roles in classification. However, in ensemble classification the weights (*i.e.* importance α_t) for each feature are distributed nearly uniformly. Hence, when the weights for selected features are optimised, the overfitting effect is reduced in SVMs.

After the SVM classifiers are trained, the training set and the testing set are given for the testing purpose. Table 5 shows the performance of SVM classification belonging to eight clients. On the 1st, 2nd, 4th, 5th and 6th

Table 5: The performance of SVM classification on eight clients with all selected features

	Number of	Training Set (%)		Testing Set (%)	
Client	Features	$\mathcal{FN}$	$\mathcal{FP}$	$\mathcal{FN}$	$\mathcal{FP}$
1	48	0.0	0.0	0.0	0.0
2	74	0.0	0.0	0.0	0.06
3	37	0.0	0.0	50	0.0
4	44	0.0	0.0	0.0	0.0
5	48	0.0	0.0	0.0	0.0
6	57	0.0	0.0	0.0	0.0
7	40	0.0	0.0	100	0.13
8	75	0.0	0.0	100	0.0

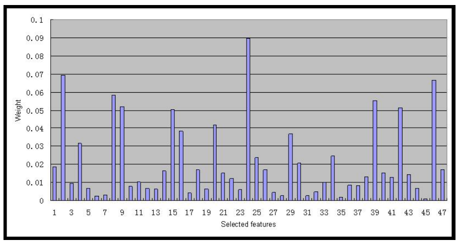
clients, SVMs achieve perfect performance, i.e. zero error. All positive examples in both the training set and the testing set are classified correctly. For the 2nd client, only one negative example is falsely recognised as positive, and the false positive rate $\mathcal{FP}$ is 0.06% which is very small and can be neglected. Hence, the SVM can achieve low error rate comparing with Table 3, the result from SVM reveals the better performance than ensemble classifier.

4.3 Summary

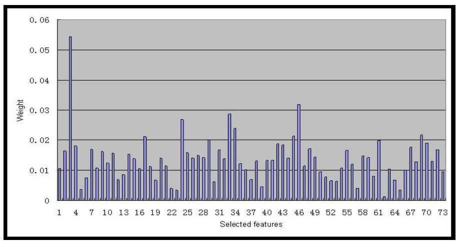
Both inferior training dataset and ensemble classifier in AdaBoost lead to overfitting. When the classifier is changed to SVM, the performance of classification improves for some clients, which demonstrates using an SVM classifier is an appropriate choice for classification. In the real implementation under a restricted face database, it is impossible to improve the training dataset, i.e. impossible to increase the number of examples per class. Hence, the only way to solve overfitting is to use alternative classifiers, such as SVM. In the proposed algorithm, significant features are selected by AdaBoost algorithm, while the verification is performed by SVM.

5 Conclusion

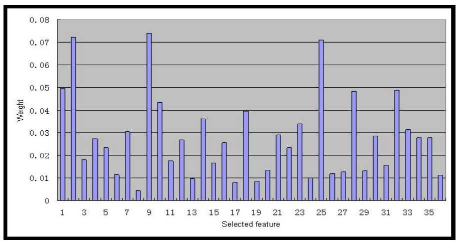
In this paper, a highly accurate appearance-based grey scale frontal face recognition algorithm - Gabor-Boosting face recognition, is presented. The strong recognition performance in the face recognition is highlighted by *Gabor wavelet*



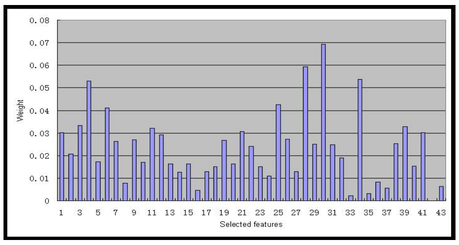
(a) The 1st client



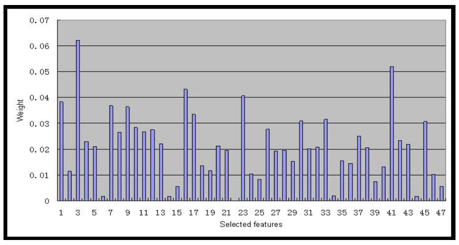
(b) The 2nd client



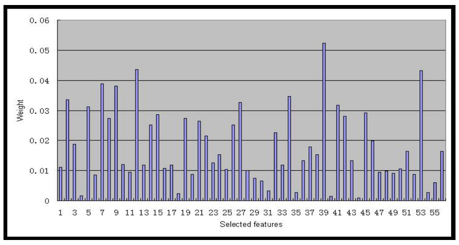
(c) The 3rd client



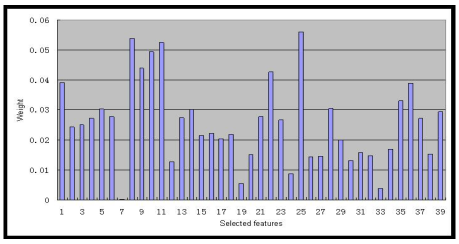
(d) The 4th client



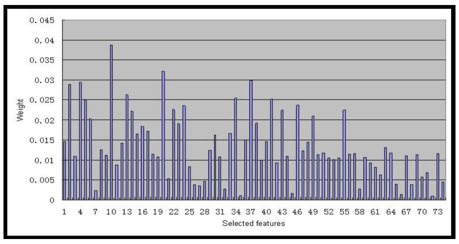
(e) The 5th client



(f) The 6th client



(g) The 7th client



(h) The 8th client

Figure 3: The weight distribution on eight clients of XM2VTS

transform, *AdaBoost*, *Support Vector Machine (SVM)*. The Gabor wavelets are of similar shape as the receptive fields of simple cells in the primary visual cortex, and demonstrate significant potential in biometrics such as iris recognition and face recognition. AdaBoost is one of the most successful and popular learning algorithms. The classification is designed to construct a "strong" classifier from "weak" learning algorithms. The SVM is a powerful learning algorithm for solving a variety of learning and function estimation problems, such as pattern recognition and regression estimation. The SVM provides non-linear classification by mapping the input into high-dimensional feature space where a special type of hyperplane is constructed. The overall recognition performance is benefited by the combination of the three key components. The Gabor wavelet transform extracts features which describe texture variations in human faces. The AdaBoost algorithm selects the most significant features which represent different individuals. The SVM constructs a hyperplane for classification with respect to selected features. In the AdaBoost feature selection, a heuristic example-based weak learner - Potsu weak learner is proposed. Potsu weak learner takes the perceptron as its prototype, and use a heuristic thresholding technique to learn decision rules from examples in training data.

The Gabor-Boosting face recognition algorithm is robust under small number of example images per subject in training data. In the situation of small number of positive examples, the system has achieved nearly 0 false positive rate and moderate false negative rate. Potential applications of the algorithm are possibly in highly secure face authentication systems dedicated to a small group of clients. The systems thus have "no false detections on impostors" and "an appropriate acceptance detection rate on clients".

References

[1] J. Daugman. High confidence visual recognition of persons by a test of statistical independence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:1148–1160, November 1993.

[2] W. Fan, Y.H. Wang, W. Liu, and T.N. Tan. Combining null space-based gabor features for face recognition. In *Proceedings of International Conference on Pattern Recognition*, 2004.

[3] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, 1996.

[4] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory: Eurocolt ?5*, pages 23–37. Springer-Verlag, 1995.

[5] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28(2):337–374, 2000.

[6] S. I. Gallant. Perceptron-based learning algorithms. *IEEE Transactions on Neural Networks*, 1(2):179–191, 1990.

[7] A. J. Grove and D. Schuurmans. Boosting in the limit: Maximizing the margin of learned ensembles. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 1998.

[8] D. Hubel and T. Wiesel. Functional architecture of macaque monkey visual cortex. *Proceedings of Royal Society on Biology (London)*, 198:1–59, 1978.

[9] A.K. Jain, P.W. Robert, and J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[10] M. Lades, J. Vorbruggen, J. Buhmann, J. Lange, C.V.D. Malsburg, and R. Wurtz. Distortion invariant object recognition on the dynamic link architecture. *IEEE Transactions on Computers*, 42:300–311, 1993.

[11] Stan Z. Li and ZhenQiu Zhang. FloatBoost learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Learning*, 26(9):1112–1123, 2004.

[12] C. Liu. Gabor-based kernel pca with fractional power polynomial models for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:572–581, 2004.

[13] C. Liu and H. Wechsler. Gabor feature based classification using enhanced fisher linear discriminant model for face recognition. *IEEE Transactions on Image Processing*, 11:467–476, 2002.

[14] J. Ma, Y. Zhao, S. Ahalt, and D. Eads. OSU SVM: A support vector machine toolbox for Matlab. Online, 2001.

[15] L. Mason, J. Baxter, P. Bartlett, and M. Frean. Boosting algorithm as gradient descent in function space. Technical report, Department of Systems Engineering, Australian National Univeristy, 1999.

[16] B. Olshausen and D. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

[17] David Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

[18] P. Jonathon Phillips, Hyeonjoon Moon, Syed A. Rizvi, and Patrick J.Rauss. The FERET evaluation methodology for face-recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1090–1104, October 2000.

[19] R. Rao and D. Ballard. An active vision architecture based on iconic representations. *Artificial Intelligence*, 78:461–505, 1995.

[20] R. Schapire, Y. Freund, P. Bartlett, and W. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. In *Proceedings of the Fourteenth International Conference on Machine Learning*, 1997.

[21] R.E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. In *Proceedings of The 11th Annal Conference on Computational Learning Theory*, number 80-91, 1998.

[22] B. Schiele and J. Crowley. Recognition without correspondence using multidimensional receptive field histograms. *International Journal on Computer Vision*, 36:31–52, 2000.

[23] P. Sollich and A. Krogh. Learning with ensembles: How over-fitting can be useful. In D. Touretsky, M. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 190–196. MIT Press, 1995.

[24] John Shawe Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge University Press, 2004.

[25] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 511–518, 2001.

[26] P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision*, 57(2):137–154, May 2004.

[27] L. Wiskott, J.M. Fellous, N. Kruger, and C. Malsburg. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:775–779, 1997.

[28] X.J. Wu, J. Kittler, J.Y. Yang, K. Messer, and S.T. Wang. A new direct LDA (D-LDA) algorithm for feature extraction in face recognition. In *Proceedings of International Conference on Pattern Recognition*, 2004.

[29] Mian Zhou and Hong Wei. Face verification using gabor wavelets and Adaboost. In *Proceedings of 18th International Conference on Pattern Recognition*, 2006.