Robust Facial Image Gender Identification Based on a Comparison between Non-Boosted and Ada-Boosted Classification Algorithms

Adil Al-Azzawi¹
¹Electrical and Computer Engineering Dept., University of Missouri, Columbia, MO 65211, USA
aaadn5@mail.missouri.edu

ABSTRACT

Identifying a user’s gender from their face is one of the most important tasks in Computer Vision today. Although many studies have been done on constrained, enhanced, or balanced dataset, little work has been done on real-world “in the wild” photo sets. We propose a PCA-based feature extraction and AdaBoosted (AB) Support Vector Machine (SVM), Naïve Bayes (NB), and K-Nearest Neighbors (K-NN) classification methods, performed on the Labeled Faces in the Wild (LFW) dataset. We preprocess the images using Viola-Jones cropping and Histogram Equalization. We train the classifiers using 5-fold Cross-Validation (CV) and evaluate them on a set of 210 male and 210 female images. We conclude that PCA does provide a good feature set and SVM gives a good accuracy for this problem, when properly tuned.

Keywords: Component; Principal Component Analysis (PCA); Support Vector Machine; Naïve Bayes; Nearest Neighbors.

INTRODUCTION

Gender classification is arguably one of the most important visual tasks, with visual information from human faces providing one of the most important sources of information for gender classification.

“Gender is perhaps the most widely studied facial demographic attribute in the Computer Vision field” [11]. This important piece of information is crucial for many real commercials applications including audience analysis and advertisement [2].

There are many relevant classification algorithms. However, the greatest constraint for this kind of problem is a good feature space to work on. The aim of this paper is to

- Prepare good feature space by using Principal Component Analysis (PCA), an underexplored method for this problem [3].
- Explore gender classification using learning algorithms and compare the results with other algorithms that do not use a learning process.

Background and motivation

Facial features differ from person to person; they are semi-rigid, semi-flexible, culturally significant, and part of our individual identity, and that needs good computing techniques for face recognition and classification. Facial features are considered to be the most acceptable biometric trait because image capturing and prediction of images is easier than other traits. For a real time, system, the size of database is large. In order to reduce the search space of a database, divide it into two halves (male and female), for this classification is needed. Classification reduces the search space at the time of identification. Gender classification can be used as indexing technique to reduce the search space for automatic face recognition [1].
Gender recognition is important because it finds its strong applications in fields of authentication, search engine accuracy, demographic data collection, human computer interaction, access control and surveillance, involving frontal facial images. Gender classification consists of taking an image previously identified as containing a human (most commonly a face) and labeling it as either male or female.

Gender classification using facial images has become an important area of research during past several years. It is easy for humans to identify male or female by looking at faces, but it is a difficult task for the computer. Machines need some meaningful data like a very good feature space to perform the identification. There exist some distinguishable features between male and female that are used by machine learning tools to classify a face image based on gender.

Gender recognition is a learning and pattern recognition problem. Supervised learning as a method of the machine learning and Pattern recognition can deal with that by dividing them into two classes and consist of feature extractor, followed by some form of classifier.

Gender Classification is a binary Classification problem that needs an appropriate data (feature space) and a classifier for Gender Classification. Gender classification approaches are categorized into two classes based on feature extraction. We rely on an approach for feature extraction and dimensionality reduction by using PCA, then feed them to three classifiers. Naïve Bayes (NB), K-Nearest Neighbors (K-NN) and Support Vector Machine (SVM) have been used as our classifiers and we used AdaBoost to enhance the weak classifier.

**Dataset**

We used the LFW dataset collected by Huang et al. [4]. The dataset contains more than 13,000 images of faces collected from the web [2].

Each face has been labeled with the name of the person pictured. 1,680 of the people pictured have two or more distinct photos in the data set. There are a total of 13,233 images and 5,749 people in the database. Each image is a 250×250 jpg [4].

![Original LFW dataset](image)

**Related Works**

Although accuracies of over 95% have been reported in the literature, we expect this to be evidence of overfitting, since this is the accuracy of a human classifier performing the same task [3].

Most work in Gender Identification is done on datasets that are controlled either by constraining, enhancing, or balancing [3]. Constrained datasets are sets of images where each of the subject has been asked to pose at the same angle, with the same expression, under the same lighting conditions [2]; a typical example is the Facial Recognition Technology (FERET) dataset [12]. Enhanced datasets are datasets which contain additional information, such as infrared channels as is the case in the Brain-like Computing & Machine Intelligence (BCMI) dataset [13]. Balanced datasets are datasets which contain a controlled number of images from each gender, and may additionally control other attributes such as ethnicity and age of the subjects, as exemplified by the Park Aging Laboratory (PAL) dataset [14].

Little work has been done on “in the wild” photo sets [3]. These are datasets which are taken from origins considered natural, such as websites, news sources, or personal collections [2, 4]. No constraints, enhancements, or balancing have been enforced. This type of work is of particular interest to us since the results are expected to be more generalizable. Unconstrained datasets also have the advantage of being most like real-world data [2] (more typifying of the classifier’s ultimate use [15]) because they often are real world data.

Castrillón-Santana et al. [2] share this interest, and evaluated an ensemble SVM classifier that used...
a linear kernel, with their feature selection being performed using Local Binary Pattern (LBP). However, atypically for ensemble classifiers, the final result was not obtained by weighted voting, but by NB. They obtain a maximum accuracy of 89.78 percent. We consider this the most similar method to our proposal.

Bekios-Calfa et al. [11] specifically studied the correlation of each of the controls on several datasets and compared it to the performance of their unconstrained counterparts. Their greatest improvement was of 6.05 percentage points when balancing by age, with all other constraints providing improvements between 0.5 and 2.04 percentage points, with a significance level of 0.6 percentage points. Of particular interest to us is their study on the unconstrained datasets, which achieved an accuracy of at most 78.33 percent, when training with the Groups dataset [8] and using the LFW dataset as the testing set. This also suggests that we will not see our measures severely impacted by not using a controlled dataset.

Caifeng Shan [15] also performed a study on the LFW dataset. The feature extraction was performed using LBP and the classification was performed using a boosted SVM with a Radial Basis Function (RBF) as the kernel. The resulting maximum accuracy was 94.81 percent, comparable to human accuracy [3].

At least four other studies have been performed on face gender recognition using PCA for the feature extraction, however, all of these studies used controlled datasets [3] and are therefore not reported here.

A summary of the referred works can be seen in Table I.

**TABLE I** Selected previous studies on “in the wild” gender identification photographs

<table>
<thead>
<tr>
<th>Reference</th>
<th>Method</th>
<th>Dataset</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catrillon-Santana 2013 [2]</td>
<td>LBP+SVM</td>
<td>GROUPS</td>
<td>89.78%</td>
</tr>
<tr>
<td>Shan 2012 [15]</td>
<td>LBP+SVM</td>
<td>LFW</td>
<td>94.81%</td>
</tr>
</tbody>
</table>

**PROPOSED SYSTEM**

Our method consists of a preprocessing phase – where the faces are detected, cropped, labeled, and normalized; features are extracted, and the different classifiers are trained – followed by a classification phase – where the different classifiers are trained, boosted, and evaluated. A summary can be seen in

Fig. 2.
METHODS

Preprocessing

1) Dataset Preparation

The first step of our proposal for the preparation of the dataset is to manually label each of the images in the dataset male and female (using −1 for male images and +1 for female images). We obtained 8,582 images for male and 2,943 images for female. The total number of images is 11,525 each image is 250×250 dimension. This process revealed that there are some images that have two different genders in the same image, which can mean more noise that will be added to our dataset, as can be seen in Fig. 3.

![Fig. 3. Two different gender in the same image](image)

a) Face Detection using Object Detection Cascade Classifier

To solve the issue of different genders in the same image, we suggest to do a face detection on the original dataset by developing the Viola-Jones face detector, known for its high detection rates [5]. This work relies on three key contributions. The first is “Integral Image” which allows the features used by our detector to be computed very quickly. The second is a learning algorithm, based on an AdaBoost classifier [7, 10]. The third one is an object classifier in a “cascade”. The process that we rely on of using cascade object detector is described in the Fig.4. The effects on the image are shown in Fig.5.

![Fig. 4. Face Detection system flowchart](image)

Then, after applying the same detector for the whole dataset, it resulted in a new face detection dataset. We called it the New_LFW dataset as shown in Fig.6.

![Fig. 5. Viola-Jones face detection with multiple faces in the same photo](image)

b) Image Cropping

After extracting the face detected dimensions, we used the same box dimension to apply the image
cropping by using Matlab function imcrop to get just the right face image from each one. This size was chosen to exclude as much external cues – such as background or hair color [3] – as possible and focus just on the relevant part of the face.

% checking is the boundary box out of index
New_dim_image= imcrop(I,[Dimensions]);

c) Image scaling to 80×80

By applying the face detector on the whole LFW dataset, we found just the right face candidate box. Each different candidate box has different dimensions. Therefore, we need to resize each image to get the same dimension for the whole dataset as is standard when performing gender classification [3]. We resize each image to 80×80 dimension. For final step our feature space for the whole dataset for face detected images is 11,525×6,400.

% image resizing to 80x80
New_dim = imresize(image,[80,80]);

d) Image converting to gray-scale

We convert each face image from RGB to grayscale by using

% convert images to gray scale
Gray_image= rgb2gray(RGB_image);

e) Histogram Equalization:

Histogram equalization is a widely-used technique [3] for adjusting the intensity values of the image pixels (the “brightness”) to enhance contrast. We used the histeq function to accomplish this. The result can be seen in Fig. 10.

% do Histogram equalization
Enhanced_image= histeq(old_image);

2) Feature Extraction and Dimensionality Reduction

For the feature extraction and dimensionality reduction we need first to normalize features and then do a dimensionality reduction on them.

a) Feature Normalization

Feature normalization (standardization) makes the values of each feature in the data have zero-mean (when subtracting the mean in the numerator) and unit-variance Eq. (1).

\[
\chi' = \frac{x-\bar{x}}{\sigma}
\]  

(1)

b) Principal Component Analysis PCA:

Principal Component Analysis, or simply PCA, is a statistical procedure concerned with elucidating the covariance structure of a set of variables. In particular, it allows us to identify the principal directions in which the data varies. For example, in Fig. 7(a), suppose that the axes U and V represent a two-variable dataset which we have measured in the X-Y coordinate system. The principal direction in which the data varies is shown by the U-axis and the second most important direction is the V-axis orthogonal to it. If we place the U-V axis system at the mean of the data, it gives us a compact representation. If we transform each (X, Y) coordinate into its corresponding (U, V) value, the data is de-correlated, meaning that the co-variance between the U and V variables is zero. For a given set of data, Principal Component Analysis finds the axis system defined by the principal directions of variance (i.e. the U-V axis system in Fig. 7(b)). The directions U and V are called the principal components.

Fig. 7. Principal Component Analysis in (a) original feature space and (b) reduced dimension space

PCA gives us a way of reducing the dimensionality of a data set. Consider two variables that are nearly related linearly as shown in Fig. (7). As in Fig. 7(b), the principal direction in which the data varies is shown by the U-axis, and the secondary direction by the V-axis. However, in this case all the V-coordinates are very close to zero. We may assume, for example, that they are only
non-zero because of experimental noise. Thus in the U-V axis system we can represent the dataset by one variable U, and discard V. Thus, we have reduced the dimensionality of the problem by 1.

![Fig. 8. PCA data projection](image)

c) PCA Data Projection:

Fig. 11 gives a geometric illustration of the process in two dimensions. Using all the data points we find the mean values of the variables \((\mu x_1, \mu x_2)\) and the covariance matrix \(\Sigma\) which is a 2x2 matrix in this case. If we calculate the eigenvectors of the co-variance matrix we get the direction vectors indicated by \(\phi_1\) and \(\phi_2\). Putting the two eigenvectors as columns in the matrix \(\Phi=[\phi_1, \phi_2]\) we create a transformation matrix which takes our data points from the \([x_1, x_2]\) axis system to the axis \([\phi_1, \phi_2]\) system with the Eq. (2):

\[ P_\Phi = (P_x - \mu_x) \Phi \quad (2) \]

Where \(P_x\) is any point in the \([x_1, x_2]\) axis system \((\mu x_1, \mu x_2)\) is the data mean, and \(p_x\) is the coordinate of the point in the \([\phi_1, \phi_2]\) axis system.

d) PCA K parameter:

The K parameter in the PCA does two major things. First one is to select some component from the whole feature space. Second is to select the dimensionality reduction space. There are two important factors that are related with the K selection: the first one is the Average Square Projection (Eq. (3)). This means that we want to minimize the difference between the original feature space and the reduction space. The second factor is the Total Variation in the data (Eq. (4)), or how many features there are.

\[ \frac{1}{m} \sum_{i=1}^{m} \left\| x^{(i)} - x^{(\text{approx})} \right\|^2 \quad (3) \]

\[ \frac{1}{m} \sum_{i=1}^{m} \left\| x^{(i)} \right\|^2 \quad (4) \]

Typically, the machine learning approach chose K to be a sum of the smallest values that satisfy the condition of Eq. (5): The K selection (the number of eigenvalues) that we have selected which gives the difference between the original feature and the reduced features, divided by how many features that we have, should be less than or equal to an \(\alpha\) value 0.01.

\[ \frac{1}{m} \sum_{i=1}^{m} \left\| x^{(i)} - x^{(\text{approx})} \right\|^2 \leq 0.01 \quad (5) \]

In other words, an \(\alpha\) of 0.01 means 99 percent of the variance can be recovered. We can choose 95 percent to be retained with \(\alpha \leq 0.05\), or 90% with \(\alpha \leq 0.10\).

e) PCA K selection:

There are two ways to select K. First, vary \(K=1, 2... n\) until the condition of variance returned has been satisfied. Another way is just pick K number. For example, \(K=3\) which means that we will reduce our feature dimensions to 3. By using the Matlab function SVD, we get an S matrix with a diagonal of eigenvalues.

\% To return 2x2 S matrix with eigenvalues 
\(\{U, S, V\}=\text{svd}(\text{Sigma});\)

For example, if \(\alpha=0.01\) Then, the sum of the K-selected eigenvalues divided by the summation of all eigenvalues should be greater than or equal to 99 percent (Eq. (6)).

\[ \frac{\sum_{i=1}^{K} \lambda_i}{\sum_{i=1}^{n} \lambda_i} > 0.99 \quad (6) \]

After testing multiple K values, we obtained a good balance between complexity and proportion of variance when \(K=90\). Approximately 90 percent of the variance is retained \((\alpha=0.1)\). Therefore, we select \(K=90\) to reduce our feature space as Fig.9 shows.
1) Training and Testing set selection

We use 5-fold cross-validation as proposed by Gherig [9].

Because the dataset has 8,582 male images and only for 2,943 female images, validation, the best way to select the folds is by down sampling [17]. We set our test size to a maximum of ten percent of the dataset. We selected 294 images at random from each of the classes, and then removed duplicates. Our result was a test set of 562 images: 287 males and 275 females. This near-equality allows for a more accurate evaluation of the resulting classifiers.

2) Cross Validation execution

The training set was then divided into five folds, following Gherig [5]; each fold equally having 10,963 randomly selected images. One fold is withheld for the validation step. The same already split folds are used to train (8,771 images per fold) and validate (2,192 images per fold) our classifiers and the performance of the trained classifier is decided by the votes collected from the classifier of each fold.

Classifier Algorithms:

In what follows the variables projected by PCA are to be used in classifier design. The three main classifiers (NB, K-NN, and SVM) are trained and the boosting mechanism is applied with the intention of enhancing the predictive power of our classifiers.

1) Non-Boosting Classifier

In our proposal we have used two kinds of classifiers approach. The first one is a Non-Boosted approach and the second one is the AdaBoosted approach.
a) Naïve Bayes

High-dimensionality in the input space is among the biggest challenges in classifier design. A Naïve Bayes model is a very simple probabilistic-restricted representation to deal with this problem by imposing the assumption that features are conditionally independent given the class label. The solution to our task is to compute the posterior probability $P(Y | X_{1:90})$. Thus, according to Bayes rule (Eq. (7)), this probability is proportional to the joint probability (Eq. (8)), which, by Central Limit Theorem, can be assumed to be normally distributed (Eq. (9)).

$$P(Y | X_{1:90}) = \frac{p(Y)p(X_{1:90}|Y)}{p(X_{1:90})}$$  \hspace{1cm} (7)

$$\propto P(Y) p(X_{1:90}|Y)$$  \hspace{1cm} (8)

$$\propto P(Y) N(\mu_{iy}, \sigma_{iy})$$  \hspace{1cm} (9)

However, it is also proportional to the prior probability $P(Y)$, which, if not set, would be empirically determined from the data. Since we know that our data is not representative of the global population, we must set a better prior. We know that there are 0.97 men for every woman in the United States [18] (where most of our images originate [4]). However, it is of note that, at birth, there are 1.04 boys for every girl [18]. We therefore use a uniform prior as an acceptable compromise.

b) K-Nearest Neighbors (K-NN)

Unlike Naïve Bayes, which makes strong assumption about both conditional independence and the distribution of the feature space in order to simplify the computation, K-NN considers, for each test point the K closest data points from the training set and assigns the new test point to the class having the largest number of representatives among the set. The parameter K controls the degree of smoothing of the approximation function and has to be determined empirically.

We choose K by enumerating from 1 to 100 the values of K and find that K=9 gives the best performance from the training set then 9-NN classifier is used for the prediction on the test set.

Unlike in Naïve Bayes, setting a prior is not required, however, we found that it increased accuracy by 20 percent. The use of a prior changes how the representatives are weighted: instead of considering each of the K neighbors equally, the largest class (male faces) finds each one of its representatives valued at 1/2.92, since there is 2.92 times as many male faces as there are female faces in the dataset. Thus, for every female neighbor found, 2.92 male neighbors are necessary for the test point to be classified as male, rebalancing the odds based on the data distribution.

c) SVM

Our input space is not linearly separable and we need to rely on soft margin SVM which both maximizes the margin w and minimizes the errors (Eq. (10)) subject to Eq. (11) and Eq. (12).

$$\min_{w} \frac{1}{2} |w|^2 + C \sum_{i} \xi_i$$  \hspace{1cm} (10)

s.t.  \hspace{1cm} $y_i (w^T x_i + b) \geq 1 - \xi_i$  \hspace{1cm} (11)

$$\xi_i \geq 0 \ \forall i$$  \hspace{1cm} (12)

The final Lagrangian dual formulation becomes Eq. (13):

$$\max_{\alpha \geq 0} \mathcal{L}(\alpha) = \sum_{i} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j$$  \hspace{1cm} (13)

s.t.  \hspace{1cm} $\sum_{i} \alpha_i y_i = 0$  \hspace{1cm} (14)

$$0 \leq \alpha_i \leq C \ \forall i$$  \hspace{1cm} (15)

Now $\alpha_i$’s upper bound is C and the solution is Eq. (16).

$$w = \sum_{i \in N_y} \alpha_i y_i x_i$$  \hspace{1cm} (16)

Then we use Sequential Minimal Optimization (SMO) to solve for each pair of $\alpha_i$ and $\alpha_j$ by freezing other variables. C is left at its default value of C=1.

The three kernel functions that have been used in the experiments to compute the inner product in the Lagrangian dual formulation (Eq. (13)) are the
Linear (Eq. (17)), Gaussian Radial Basis Function (Eq. (18)), and Polynomial Kernels (Eq. (19)).

**Linear:**

\[
G(x_i, x_j) = x_i^T x_j K(x^{(i)}, x^{(j)}) = x^{(i)T} x^{(j)}
\]  

**Gaussian RBF:**

\[
G(x_i, x_j) = e^{-\|x_i-x_j\|^2}
\]  

**Polynomial:**

\[
G(x_i, x_j) = (1 + x_i^T x_j)^2
\]

The Gaussian Radial Basis Function (RBF) tuning can be achieved by scaling the input vectors by a scalar value \( \sigma \) before the kernel transformation (Eq. (20)), resulting in Eq. (21).

\[
x' = \frac{x}{\sigma}
\]

\[
K'(x'^{(i)}, x'^{(j)}) = e^{-\|x'^{(i)}-x'^{(j)}\|^2}/\sigma^2
\]

Alternatively, Matlab can automatically select the optimal scaling via heuristic procedure using subsampling. [16]

2) **Boosted Classifiers Approach**

Finally, we employ AdaBoost to try improving the performance of our classifiers by training an ensemble of classifiers on the re-weighted training set, and the new weights are updated according to Eq. (22)

\[
D_{t+1}(i) = \frac{D_t(i) e^{-\alpha_t h_t(x^{(i)})}}{Z_t}
\]

where \( Z_t \) is a normalization factor to make \( D_{t+1} \) a valid distribution. By choosing \( \alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t} \) we can prove that the new weights (Eq. (30)) only depend on the current weights and the error term \( \epsilon_t \). Eq. (23) through (30) show the derivation of this proof.

**Correct**

\[
D_{t+1}(i) = \frac{D_t}{Z_t} \begin{cases} e^{-\alpha_t}, & \text{Correct} \\ e^{\alpha_t}, & \text{Wrong} \end{cases}
\]

**Wrong**

\[
D_{t+1}(i) = \frac{D_t}{Z_t} \begin{cases} \sqrt{\frac{\epsilon_t}{1-\epsilon_t}}, & \text{Correct} \\ \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}, & \text{Wrong} \end{cases}
\]

\[
D_{t+1}(i) = \sqrt{\frac{\epsilon_t}{1-\epsilon_t}} \left( 1 - \epsilon_t \right) + \sqrt{\frac{1-\epsilon_t}{\epsilon_t}} \left( \epsilon_t \right)
\]

\[
Z_t = 2 \sqrt{(1-\epsilon_t)\epsilon_t}
\]

\[
\sqrt{\frac{1-\epsilon_t}{\epsilon_t}} \sum_{i \in \text{Wrong}} D_t(i)
\]

\[
D_{t+1}(i) = \frac{D_t(i) e^{-\alpha_t h_t(x^{(i)})}}{Z_t}
\]
\[ D_{t+1}(i) = \frac{D_{t+1}(i)}{2} \left( \frac{1}{1-\epsilon_t} \right) \]

Epsilon sub t \((\epsilon_t)\) is computed by summing all the weights of the misclassified samples Eq. (31).

\[ \epsilon_t = \sum_{i \in \text{Wrong}} D_t(i) \]  

Therefore, the final classifier is simply the sign of weighted votes from all learners in the ensemble Eq. (32).

\[ H(x) = \text{sign} \left( \sum_{t=1}^{T} a_t h_t(x) \right) \]

We evaluate each classifier’s performance after all possible number of \(l\) rounds of boosting up until 16 rounds, reflecting the performance of an \(l\)-round boosted classifier, for \(l=1\ldots16\).

**RESULTS**

In our experiments, we compare each of our classifiers in term of 5-fold CV performance for both Non-boosted and boosted settings.

We observe that all SVM methods attain above 80 percent accuracy on both the training and test set, with the highest training accuracy of 89.68 percent for the 3rd order polynomial kernel SVM and the highest test accuracy of 87.37 percent for Gaussian RBF kernel as shown in Table II.

After applying AdaBoost for 16 iterations on Gaussian RBF SVM, the test accuracy improves only by 0.2 percentage points. While performance gain is up by 1.42 percentage points at the fourth boosting iteration of the 3rd order polynomial SVM, which settled at about 86.48%. The Naïve Bayes classifier stopped improving at 79 percent accuracy after 7 iterations. We have also shown that boosting doesn’t have any effect at all as far as the 9-Nearest Neighbors classifier is concerned.

Although the training accuracies for Gaussian and 3rd order polynomial kernel SVM increase exponentially when boosted – and peak at 100 percent starting from the third iterations of the test accuracies fluctuating – linear kernel SVM is unaffected by boosting.

**TABLE II Non-Boosted classifier accuracy**

<table>
<thead>
<tr>
<th>SVM</th>
<th>Training</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian RBF SVM</td>
<td>89.32%</td>
<td>87.37%</td>
</tr>
<tr>
<td>3rd order Polynomial</td>
<td>89.68%</td>
<td>86.48%</td>
</tr>
<tr>
<td>Linear</td>
<td>82.34%</td>
<td>81.67%</td>
</tr>
<tr>
<td>9-Nearest Neighbors</td>
<td>82.34%</td>
<td>77.76%</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>77.10%</td>
<td>76.16%</td>
</tr>
</tbody>
</table>

**TABLE III Boosted classifier accuracy**

<table>
<thead>
<tr>
<th>Boosting iteration</th>
<th>Gaussian RBF SVM</th>
<th>3rd order Polynomial SVM</th>
<th>9-NN</th>
<th>Naïve Bayes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>87.37%</td>
<td>86.30%</td>
<td>77.76%</td>
<td>76.16%</td>
</tr>
<tr>
<td>2</td>
<td>84.88%</td>
<td>87.01%</td>
<td>77.76%</td>
<td>76.16%</td>
</tr>
<tr>
<td>3</td>
<td>86.48%</td>
<td>86.65%</td>
<td>77.76%</td>
<td>78.11%</td>
</tr>
<tr>
<td>4</td>
<td>84.52%</td>
<td>87.90%</td>
<td>77.76%</td>
<td>77.94%</td>
</tr>
<tr>
<td>5</td>
<td>85.94%</td>
<td>87.19%</td>
<td>77.76%</td>
<td>78.29%</td>
</tr>
<tr>
<td>6</td>
<td>85.77%</td>
<td>86.65%</td>
<td>77.76%</td>
<td>78.29%</td>
</tr>
<tr>
<td>7</td>
<td>84.88%</td>
<td>86.65%</td>
<td>77.76%</td>
<td>79%</td>
</tr>
<tr>
<td>8</td>
<td>85.94%</td>
<td>86.12%</td>
<td>77.76%</td>
<td>79%</td>
</tr>
<tr>
<td>9</td>
<td>86.30%</td>
<td>86.65%</td>
<td>77.76%</td>
<td>79%</td>
</tr>
<tr>
<td>10</td>
<td>86.65%</td>
<td>86.65%</td>
<td>77.76%</td>
<td>79%</td>
</tr>
<tr>
<td>11</td>
<td>86.48%</td>
<td>86.83%</td>
<td>77.76%</td>
<td>79%</td>
</tr>
<tr>
<td>12</td>
<td>86.30%</td>
<td>85.94%</td>
<td>77.76%</td>
<td>79%</td>
</tr>
<tr>
<td>13</td>
<td>87.01%</td>
<td>86.30%</td>
<td>77.76%</td>
<td>79%</td>
</tr>
<tr>
<td>14</td>
<td>87.01%</td>
<td>86.30%</td>
<td>77.76%</td>
<td>79%</td>
</tr>
<tr>
<td>15</td>
<td>87.19%</td>
<td>86.48%</td>
<td>77.76%</td>
<td>79%</td>
</tr>
<tr>
<td>16</td>
<td>87.54%</td>
<td>86.48%</td>
<td>77.76%</td>
<td>79%</td>
</tr>
</tbody>
</table>

**Fig. 12.** Training and Testing Accuracies for Boosted SVMs
CONCLUSION

We were able to successfully classify the photographs with an accuracy comparable to state-of-the-art methods [3].

Of note is the particular complexity of the feature extraction process. We found that PCA is indeed a suitable dimension reduction for this type of problem, but acknowledge that determining the appropriate K for an optimal data recovery is an extra step on the part of the investigators.

K-Nearest Neighbors and Naïve Bayes proved to indeed be weak learners, especially with K-NN’s tendency to favor a different class for even values of K than it did for odd values of K. We note that setting a prior provides a huge improvement to accuracy.

The Support Vector Machine proved to be the best method in our experiments. Once again, finding the appropriate tuning parameters is acknowledged to be extra troublesome on the part of the researchers.

Finally, we find that boosting does not offer much improvement in this case. Despite its highly touted improvement power [7, 10] and reported high accuracy in this problem [3], we found that our greatest improvement was only of three percentage points.

FUTURE WORK

Although we took our number of cross-validation folds from Gherig’s benchmarking protocol [9], Gherig also suggests ensuring that each subject appears in one and only one of each of the cross-validation folds “To prevent algorithms from learning the identity of the persons in the training set”. We believe that enforcing this constraint could increase our test set accuracy. Doing this automatically is possible with the LFW dataset, since each subject has a unique name [4] but is left as future work.

We also believe our approach could benefit from using other feature extractors, such as LBP or LDA which seem to be in wide use [3] as well as other kernels for SVM.

We believe that an Ensemble learner made up of the three classifiers we experimented with here could also provide great benefits, as well as ensemble learners consisting of deep learning classifiers. However, this is also left as future work.

REFERENCES


