ResNet and PCA-Based Deep Learning Scheme for Efficient Face Recognition

Rajendra Kumar Dwivedi, Madan Mohan Malaviya University of Technology, India*
https://orcid.org/0000-0001-6682-1942
Devesh Kumar, Babasaheb Bhimrao Ambedkar University, India

ABSTRACT

Face recognition is an emerging field of research in recent days. With the rise of deep learning, face recognition has become efficient and precise, creating new milestones. The performance, accuracy, and computational time of the existing schemes can be enhanced by devising a new scheme. In this context, a multiclass classification framework for face recognition using residual network (ResNet) and principal component analysis (PCA) schemes of deep learning with Dlib library is proposed in this paper. The proposed framework produces face recognition accuracy of 99.6% and a reduction of computational time with 68.03% using principal component analysis.

KEYWORDS

ResNet and PCA-Based Deep Learning Scheme for Efficient Face Recognition

1. INTRODUCTION

Face recognition is an emerging technology in Artificial Intelligence and computer vision with many research ongoing to develop fast, robust, and more accurate system which can be used in many fields of application like security systems (Othman & Aydin, 2018), video surveillance or verification like done on various online platforms like Facebook and google photos. Apart from this face recognition is being applied in various real-life applications. Various algorithms for 2D and 3D face recognition are being developed (Abate et al., 2007; Masupha et al., 2015). Face recognition can be implemented in various real-life application like security and surveillance (Jaiswal et al., 2020), criminal identification, forensic applications (Kute et al., 2019), threat detection (Sarma et al., 2017), face verification, and attendance systems (Arsenovic et al., 2017).

1.1 Motivation

The performance, accuracy, and computational time of the existing schemes can be enhanced by devising a new scheme. Nowadays, various online platforms use their trade secret algorithms for face

DOI: 10.4018/IJIIT.329957

*Corresponding Author
This article published as an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.
recognition and charge for them. The motivation of this paper is to develop a robust, fast, and optimized system for face recognition with the help of various freely available tools. The system developed thus can be implemented in any existing frameworks which can utilize the objective of face recognition.

1.2 Contribution

We devised a framework using the machine learning library Dlib for face detection and landmark estimation, a Residual Network for feature vector calculation, PCA for dimensional reduction, and various multi-class classifiers for classification. The experiment is done to reduce the computational times of various classifiers after applying PCA and fine-tuning of parameters of the classifiers. The performance of various classifiers is evaluated on the validation set which contains nonoverlapping data from the training dataset.

1.3 Organization

The rest of this paper is organized as follows. Section 2 comprises the related research in the field of learning-based facial recognition. The methodology of the proposed work is discussed in section 3. Section 4 describes the proposed algorithms and implementation is discussed in section 5. Section 6 deals with the experimental results and discussion thereon. Finally, section 7 provides conclusions with some future directions.

2. BACKGROUND AND RELATED WORK

The research area of face recognition continues to evolve these days. It became an important part of various real-life applications. Several face recognition systems have been developed which are fast and robust. This section presents the background and a brief survey in this field.

2.1 Basic Principles in Facial Recognition

Face recognition algorithms are generally based on the feature extraction that is present in images of faces. The facial recognition system will compare the features present in the database with the features extracted from the face images. By defining some confidence value, when this confidence threshold value is crossed then the faces can be said to be similar, the result is then shown. The process of facial recognition can be divided into two categories: One-to-One comparison of image for confirmation; other is One-to-Many comparison for the sake of identification.

In a specialized way, facial recognition systems include a list of interdisciplinary technology to build a system, which includes the acquisition of face images, locating the faces, preprocessing for facial recognition like pose normalization (Al-Obaydy & Suandi, 2020), extraction of features, identity recognition, identity search, etc. In a generalized way, facial recognition is a system for identification either through search or comparison. Apart from this face recognition can be implemented into expression analysis like fatigue detection (Li et al., 2021).

2.2 Some Face Recognition Methods

Many machine learning and deep learning-based schemes have been developed for face recognition systems. However, their performance, accuracy, and computational overhead can be optimized. Some such face recognition methods are discussed in this section. Table 1 presents a comparative analysis of the existing face recognition schemes.

2.2.1 Based on Geometrical Features of Faces

The calculation of geometrical features from the image of a face is the basis of this method (Ouarda et al., 2014). The entire configuration may be stated as a vector that represents the location and the dimensions of the most important facial features like chin, lips, nose, eyes, mouth, and many others.
In terms of recognition speed and memory, these algorithms perform very well, but the accuracy of recognition is very poor.

2.2.2 Based on Eigen Faces (PCA)

The face recognition based on eigenvectors computed from face images can be done. To escalate the efficiency and to reduce the time taken in recognition, Principal Component Analysis (Kaur & Himanshi, 2015; Zafaruddin & Fadewar, 2019) is done on the eigenvectors and all the irrelevant information is then truncated. A slightly different version of kernel PCA-based dimensional reduction on eigenfaces was done by Kim et al, 2002.

2.2.3 Based on Laplacian Faces (PCA)

The Eigenface-based recognition method intends to preserve the broad structure of the image features, Laplacian Face (He et al, 2005) based recognition method intends to preserve the narrow structure of the image features. In many practical classification problems, the local manifold arrangement is more significant than the global Euclidean arrangement, specifically when nearest-neighbor-like classification is done.

2.2.4 Based on Support Vector Machine

Face image classification based on actual pixel value is also a method for face recognition. Sani et al, 2009 used the Support Vector Machine to classify the faces on the Yale database. unlike any other SVM face recognition approaches they didn’t apply any preprocessing on the data such as feature extraction, dimension reduction techniques, or illumination correction method.

2.2.5 Based on Compact Binary Face Descriptor

Many facial recognition systems used Local Binary Patterns (LBP) like binary feature descriptors and their variants, due to their outstanding robustness and resilient discriminative power. Compact Binary Face Descriptor (CBFD) (Lu et al, 2015) is a more advanced and efficient version in Binary feature descriptor which is based on the extracted pixel difference vectors (PDV) in local patches, a mapping of feature to project these PDVs into lower-dimensional binary vectors in an unsupervised manner is called as CBFD.

2.2.6 Based on Convolutional Neural Network

The artificial neural network is a nonlinear dynamic structure that is known for its great self-organization and adaptive ability. Gou et al., 2019, have compared various deep learning-based face recognition algorithms. After the rise of the convolutional neural network (CNN), the efficiency and robustness of the face recognition systems got a boom (Goswami et al., 2019). With the highest rates of accuracy and precision of recognition, it proved to be the best method so far. DeepID3 (Sun et al, 2015) uses a convolutional neural network for face recognition. Parkhi et al, 2015, produced a web scraped dataset of 2.6M images of 2.6K people and trained a CNN on it. Sharma et al, 2016, developed a face recognition system using CNN and tested it on the Face Recognition Grand Challenge (FRGC) dataset. Coskun et al, 2017, developed a face recognition system using CNN and a softmax classifier which was tested on Georgia Tech Face Database. Liu et al., 2017, developed an open-source face recognition development kit named VIPLFaceNet using deep neural networks.

3. METHODOLOGY OF THE PROPOSED WORK

The principles for implementing various fragments of a face recognition system are described in this section as follows.
Table 1. Comparative analysis of the existing face recognition schemes

<table>
<thead>
<tr>
<th>Author</th>
<th>Approach</th>
<th>Outcome</th>
<th>Limitations</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Al-Obaydy et al., 2020</td>
<td>Pose normalization in video surveillance</td>
<td>Substantially improved performance of face recognition system</td>
<td>Still cannot reach 100% accuracy in the face recognition in video surveillance</td>
<td>The limitation can be overcome by improving and optimizing the algorithm</td>
</tr>
<tr>
<td>Li et al., 2021</td>
<td>Fatigue monitoring system using face recognition</td>
<td>High fatigue detection accuracy and great performance in real-time</td>
<td>The algorithm can be made robust and a low light environment should be considered</td>
<td>The algorithm can be improved by accounting for various aspects rather than only relying on eyes and face.</td>
</tr>
<tr>
<td>Ouarda et al., 2014</td>
<td>Face Recognition based on geometric features</td>
<td>A decent working face recognition system</td>
<td>Only accounting geometrical features cannot recognize the same person at different age</td>
<td>High speed but poor recognition rate</td>
</tr>
<tr>
<td>Kaur et al., 2015</td>
<td>Eigen-faces with PCA</td>
<td>A simple yet effective working face recognition system</td>
<td>Poor recognition in different background</td>
<td>An easy method for face recognition but limitations must be accounted</td>
</tr>
<tr>
<td>Zafaruddin et al., 2019</td>
<td>Eigen Faces with PCA and Neural Networks</td>
<td>A face recognition system with 93% accuracy</td>
<td>Works great only on low-resolution images</td>
<td>Can be computationally costly for high-resolution images</td>
</tr>
<tr>
<td>Kim et al., 2002</td>
<td>Eigen-faces with kernel-PCA and SVM</td>
<td>A face recognition model with an error rate reduced to 1.5%</td>
<td>Only works on grayscale images</td>
<td>The recognition rate is great but only limited to grayscale images</td>
</tr>
<tr>
<td>He et al., 2005</td>
<td>Laplacian faces with PCA</td>
<td>A face recognition algorithm with an error rate of 8.17%</td>
<td>Only great performance in a frontal pose</td>
<td>A face alignment pre-processing can be done to make the algorithm more feasible</td>
</tr>
<tr>
<td>Sani et al., 2009</td>
<td>SVM</td>
<td>Reduced pre-processing time as compared to PCA</td>
<td>Poor recognition rate and only works on grayscale images</td>
<td>Only works on a very low resolution of 25x25 and had no real-life implementation</td>
</tr>
<tr>
<td>Lu et al., 2015</td>
<td>CBFD</td>
<td>A new feature learning method called compact binary face descriptor (CBFD)</td>
<td>Only worked on a single layer of the network</td>
<td>High performance can be achieved using deep networks</td>
</tr>
<tr>
<td>Goswami et al., 2019</td>
<td>Detection and mitigation of Adversarial perturbations</td>
<td>Tested adversarial perturbations on various face recognition models and mitigated them</td>
<td>More algorithms can be tested, and algorithms can be optimized</td>
<td>Various deep learning models are included, and great comparison is performed</td>
</tr>
<tr>
<td>Sun et al., 2015</td>
<td>Deep convolutional neural network</td>
<td>DeepID3 is obtained with very high recognition rates and speed</td>
<td>Performed on a small dataset</td>
<td>A neural network-based face recognition system with great accuracy</td>
</tr>
<tr>
<td>Parkhi et al., 2015</td>
<td>Deep Convolutional neural network</td>
<td>A method for large dataset collection and face recognition system with great accuracy</td>
<td>Pre-processing of data can be further improved</td>
<td>A great method for creating the custom dataset</td>
</tr>
<tr>
<td>Sharma et al., 2016</td>
<td>Deep learning and CNN</td>
<td>Face recognition system with 96% accuracy and 0.1 FAR</td>
<td>Performance decrease in pose and intensity variation</td>
<td>An overall good face recognition system</td>
</tr>
<tr>
<td>Coskun et al., 2017</td>
<td>CNN and softmax classifier</td>
<td>A system developed for great face recognition</td>
<td>Time costly and works on low-resolution images</td>
<td>It can be improved with testing on more datasets</td>
</tr>
<tr>
<td>Liu et al., 2017</td>
<td>Deep learning and CNN</td>
<td>An SDK developed and released for the face recognition system</td>
<td>Pre-processing of the data can be done more efficiently</td>
<td>A great development kit for the implementation of face recognition</td>
</tr>
</tbody>
</table>
3.1 Detection of Faces and Landmark Estimation

Face detection is the method to detect faces in the provided image. Dlib provides mainly two built-in methods for face detection viz. Histogram of Oriented Gradient (HOG) (Cerna et al, 2013; Nigam et al, 2018; Li & Lin, 2017) and Convolutional Neural Network (CNN) (Mliki et al, 2020). However, CNN-based face detection is more advanced and provides the best results, but it is computationally very costly and cannot run in real-time CPU. So for the scope of this system, we will be using Dlib’s Frontal Face Detector with HOG, the working of the detector is like first of all the provided RGB image is converted into grayscale for the sake of simplicity, the distribution of the directions of the gradients is used as features, then the features extracted by HOG is fed through a linear SVM. The Frontal Face Detector returns a ‘Rectangle Dlib object’ which contains information like coordinates, area, center point, etc. Figure 1 presents HOG description of an image.

After the coordinates are obtained the next step is the estimation of landmarks of the faces. Landmark estimation is a tool for facial feature point extraction. There is a pre-trained face landmarks detector provided in the Dlib toolkit which is used to approximate the locations of 68(x,y)-coordinates that map to facial structure on the face the conception of the landmarks is shown in Figure 2. These landmarks are part of the 68 points iBUG 300-W dataset which was used in the training of the Dlib facial landmark predictor. Figure 3 presents the predicted landmarks of an input image.

3.2 Face Feature Vector Computation

After the facial landmark points are acquired, they are then used to align face position by affine transformations, if not aligned, the accuracy of face recognition gets affected if it is not frontal pose. After the alignment, the face gets fed through the face recognition residual network. 128-dimensional feature vectors are then produced, which is used as the base of face recognition. The Euclidian distance or similarities in Cosine of vectors of two faces is computed to check their similarity, with some predefined confidence threshold.

How the Face feature vectors are calculated? Dlib library uses Residual network (He et al., 2015; Khan et al., 2018) followed by DNN to generate feature vectors. Residual Network is a deep convolution network with feedback loops proposed by He et al., 2015. Once it was created, it became the champion in three competitions in image classification, detection, and location in ImageNet.
Figure 2. Conception of the 68 facial landmark points from the iBUG 300-W dataset

Figure 3. Input image and predicted landmarks
Dlib Library Uses a slightly modified version of the ResNet-34 network model proposed by Kaiming He and team, with 29 Convolutional layers and the number of filters in layers reduced by half, the ResNet is then added with another Dense(128) layer, so it can generate 128-dimensional feature vector. The resultant network is then trained on about 3 million faces of 7485 individual identities. The Network model obtained is so powerful which then can be used to generate a feature vector of any human face. Figure 4 shows the workflow of the face feature vector calculation.

3.3 Face Recognition System

It is quite complex involving various steps and methods, which in summary can be stated as detecting face, locating face, pre-processing, extracting of features, training of features, and classification task. In addition to mentioned steps data dimension reduction methods is also a key step in the face recognition process. In this paper Principal Component Analysis (PCA) is used for dimension reduction. The framework used in this system is of two portions. The first portion consists of training and model generation and the second portion works on face recognition. The workflow of this framework is shown in the flowchart in Figure 5.

3.4.1 Machine Learning Model Training

To train a machine learning model, first of all, the training data must be imported. For each image in the training set, the face on the image is detected and located, after all the facial feature points are located and fed through the ResNet+DNN network for the generation of the 128-D feature vector, each vector is then associated with the corresponding label which is the name identity of the person. After these steps, the trained ML model is obtained.

3.4.2 Recognition Section

For input image or video, the first step is of face detection and location, after that, the face feature vectors are calculated for each faces present in the image or frame of the video, after this, using the trained ML model obtain the classification outcome and mark it down on the screen.

4. PROPOSED ALGORITHMS

The proposed scheme has main four algorithms as follows. Algorithm 1 is for the face detection process. Algorithm 2 describes the Landmark Estimation process. Methodology of Face Feature Vector Calculation is explained in Algorithm 3 and the procedure of Face Recognition System is discussed in Algorithm 4. Figure 6 presents the flowchart for face detection, landmark estimation, and face feature vector calculation process. Flowchart for the face recognition system is shown in Figure 7.

Algorithm 1: Face Detection

Input: Image
Output: Dlib Rect Object

Figure 4. Workflow of face feature vector calculation
Algorithm 1 performs the task of face detection which is done using HOG features of image. The input of this algorithm is an image file and the output is a Dlib ‘rect’ object which contains the information about corners of the bounding box, center of the face, and area of the bounding box.

Algorithm 2: Landmark Estimation

**Input:** Dlib ‘Rect’ object and image

**Output:** 68 coordinate points of face landmarks

Begin
1. Input Dlib Rect object and image.
2. Feed landmark estimator model with rectangle coordinates and image.
3. Obtain 68 coordinates of face landmarks.
End
Algorithm 2 is used for the estimation of the 68 landmark points that are present in face images, the input is the Dlib ‘rect’ object and the image file which then passed through the pre-trained model for landmark estimation, the output of this algorithm is coordinates of all the 68 landmark points that are present in the image of the face.

Algorithm 3: Face Feature Vector Calculation

**Input:** 68 coordinate points of face landmarks and image

**Output:** 128-D face feature vector

Begin

1. Input 68 coordinate points and image.
2. Using 68 points align face using Affine Transformation.
3. Feed aligned face through Resnet + DNN model.
4. Obtain a 128-D face feature vector.

End

The task of face feature vector computation is done in Algorithm 3, the 68 landmark coordinates along with the image are provided as input, using the landmark points and affine transformation the faces are aligned, the aligned face is then fed into a residual network for feature vector generation which is also the output of this algorithm.

The face recognition system is presented in Algorithm 4, which takes the image dataset with proper labels as input and gives performance results in the output. For every image present in the dataset the task of face detection, landmark estimation, and feature vector generation is performed after that the obtained 128D feature vector along with the proper label is then stored into a separate CSV file, on this newly created dataset of feature vectors and labels PCA is performed and the dataset is reduced and then it is split into training and test set, now grid search and cross-validation is performed on the training set for the sake of fine-tuning of hyper-parameters, after that the trained model is then tested on the test set. For validation purposes, another dataset of non-overlapping data from the original set is taken and the performance of the model is obtained.

Figure 6. Flowchart for face detection, landmark estimation, and face feature vector calculation
Algorithm 4: Face Recognition System

**Input:** Dataset and validation set of face images with proper labels

**Output:** Performance of the model

**Begin**

1. Input dataset.
2. For each image in the dataset.
   2.1. Face detection and location.
   2.2. Facial landmarks estimation.
   2.3. 128-D facial feature vector generation.
   2.4. Storing encodings and labels into a CSV file.
3. Principal Component Analysis of new dataset of encodings and labels.
4. Split data into a training set and a test set.
5. Apply grid search and k-fold cross-validation on the training set to optimize hyper-parameters of the multiclass classifier.
6. Train model using parameters obtained from step 5.
7. Obtain the model and test performance on the test set created in step 4.
8. For each image in the validation set.
   8.1. Face detection and location.
   8.2. Facial landmarks estimation.
   8.3. 128-D facial feature vector generation.
   8.4. Storing encodings and labels into a CSV file.
**End**

5. IMPLEMENTATION

5.1 Dlib Toolkit

Dlib toolkit is a modern C++ library that consists of a range of various tools and algorithms of deep learning, machine learning, image processing, and face recognition. This toolkit is extensively used in automation and robotics, mobile devices, and various big high-performing computational areas. Dlib toolkit can certainly help us in the building of a complete high-performance facial recognition scheme.

5.2 Dataset Preparation

For the performing following experiment a custom dataset of 6800 face images of 657 persons of a different race, age and gender is taken. To create this dataset, some data has been taken from the LFW dataset, and the rest is scraped from the web.

5.3 Face Detection and Landmark Estimation Experiment

In this phase of the experiment the dataset is fed through face detection and landmark estimation algorithm, Figure 8 shows 6 random person’s landmark estimation through the algorithm. According to the figure, the landmark estimated does not depend on the angle, pose, or position of the face. Different landmarks are essentially in the corresponding positions of the faces.

To check the accuracy of the algorithm the whole dataset of 6800 images is fed through it, among them, 6459 can be effectively extracted with feature points of faces with an accuracy of 94.98%. Here,
we can conclude that if the face is not clear and visible due to small size, occultations on faces or the picture being not clear can result in the failure of the algorithm to extract the face feature points.

5.4 Face Feature Vector Calculation

All the extracted feature points are then fed through the ResNet+DNN Network to obtain the 128-D feature vector. After this the feature vector along with the label is stored in a new dataset and then Principal Component Analysis (PCA) is performed for the reduction of dimensions on the dataset. Out of 128 features, the Principal component is found around 64 features with 0.98 variances retained as shown in Figure 9.

5.5 Process of Face Recognition

It can be described as detecting faces, estimating facial landmarks, feature vector computation, and classification based on the face distance. The detailed workflow is shown in Figure 10. The classifier whether be it SVM, KNN, or LR compares the input data with the training data and makes the prediction based on the internal algorithm used. The model derived from training was then tested on the validation set. 227 non-overlapping images data of 104 people at random is selected and prediction is made.

6. PERFORMANCE EVALUATION

We have used various confusion matrix-based performance metrics viz., Accuracy (A), Precision (P), Recall (R), and F1-score (F1) to evaluate the performance of the schemes. We also compared the computational time of the schemes to measure their performance.
Figure 8. Extracted face landmark points of six random persons

6.1 Performance Metrics
Figure 9. Principal component analysis on the dataset

![Principal Component Analysis](image)

Figure 10. The work-flow of face recognition

![Face Recognition Workflow](image)
Various performance metrics based on the confusion matrix (A, P, R, F1) are described as follows. The confusion matrix is a matrix represented in terms of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN).

6.1.1 Accuracy (A)
Accuracy is the measurement of precisely classified data. The procedure of computing Accuracy is given in Eq. (1).

$$A = \frac{TP + TN}{TP + FP + FN + TN} \times 100$$

(1)

6.1.2 Precision (P)
Precision is the rate of true-positive prediction over total-positive prediction. It is calculated according to Eq. (2). Eq. (3) depicts the weighted average precision of the scheme.

$$P = \frac{TP}{TP + FP} \times 100$$

(2)

$$weightedP = \frac{\left(\frac{P_{c1} \times |c1| + P_{c2} \times |c2| + \ldots}{|c1| + |c2| + \ldots}\right)}$$

(3)

Here $P_{c1}, P_{c2}, \ldots$ are the precisions for class 1, class 2, and so on, and $|c1|, |c2|, \ldots$ are the number of instances in class 1, class 2, and so on.

6.1.3 Recall (R)
It is the rate of true-positive-predictions over the sum of true-positive-predictions and false-negative-predictions. Sometimes, it is also called Sensitivity. It is calculated as Eq. (4). The weighted average recall is can be calculated with the help of Eq. (5).

$$R = \frac{TP}{TP + FN} \times 100$$

(4)

$$weightedR = \frac{\left(\frac{R_{c1} \times |c1| + R_{c2} \times |c2| + \ldots}{|c1| + |c2| + \ldots}\right)}$$

(5)

Here $R_{c1}, R_{c2}, \ldots$ are the recalls for class 1, class 2, and so on, and $|c1|, |c2|, \ldots$ are the number of instances in class 1, class 2, and so on.

6.1.4 F1 Score (F1)
It is computed with the help of Recall and Precision. The method of computing the F1 Score is shown in Eq. (6) and Eq. (7) shows the calculation of the weighted average F1 score.

$$F1 = \frac{2 \times R \times P}{R + P} \times 100$$

(6)
Here $F_{1c1}, F_{1c2}, \ldots$ are the F1 scores for class 1, class 2, and so on, and $|c1|, |c2|, \ldots$ are the number of instances in class 1, class 2, and so on.

6.2 Experimental Results

The performance of the classifiers does not change after applying PCA but in account of the computational cost, it performs very well. SVM or KNN either can be used as classifiers for the face recognition system. But if the demand is for lightweight with a faster classification such as in video feed or live stream KNN is the go-to algorithm, however, SVM also performs very efficiently but with the trade-off of computational time.

6.2.1 Performance Evaluation Based on Confusion Matrix Parameters

Table 2 presents the results of the various performance metrics used for performance analysis of the proposed and the existing schemes. The performance of the three classifiers viz., SVM, KNN, and LR on the training set is shown in Figure 11.

6.2.2 Performance Evaluation Based on Computational Time

Table 3 contains the computation time of training and prediction. Figure 12 contains the information about the time taken by the various algorithms in milliseconds and the prediction time of various

Table 2. Performance metrics of the various algorithm on the validation set

<table>
<thead>
<tr>
<th>Approach</th>
<th>Accuracy %</th>
<th>Precision %</th>
<th>Recall %</th>
<th>F1-Score %</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA+SVM</td>
<td>99.6</td>
<td>99.7</td>
<td>99.6</td>
<td>99.5</td>
</tr>
<tr>
<td>PCA+KNN</td>
<td>99.1</td>
<td>100</td>
<td>99.1</td>
<td>99.1</td>
</tr>
<tr>
<td>PCA+LR</td>
<td>98.2</td>
<td>100</td>
<td>98.2</td>
<td>98.6</td>
</tr>
</tbody>
</table>

Figure 11. Performance metrics of various classifiers
algorithms is presented in Figure 13. The overall best performing algorithm is PCA+KNN in terms of both successful prediction and computational time combined.

6.2.3 Performance Evaluation Based on M-Fold Cross-Validation

The various m-fold cross-validation performed on the reduced dataset with PCA and using the various algorithms is shown in Table 4. The grid search method applied on the reduced dataset for obtaining the fine-tuning of hyper-parameters of various algorithms provides, for SVM the values of \( \text{cost}=100, \ gamma=0.1 \) and \( \text{kernel}='\text{linear}' \) is found, for KNN the values of \( \text{metric}='\text{euclidean}', \ n\_\text{neighbors}=2 \) and \( \text{weights}='\text{distance}' \) is found and for LR the value of \( \text{cost}=1000 \). The obtained models are then tested for over-fitting in which they passed.

Table 3. Computation Time of training and prediction

<table>
<thead>
<tr>
<th>Approach</th>
<th>Training Time (mSec)</th>
<th>Prediction Time (mSec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>6152.7</td>
<td>19215.9</td>
</tr>
<tr>
<td>PCA+SVM</td>
<td>3608.1</td>
<td>18209.7</td>
</tr>
<tr>
<td>KNN</td>
<td>216.2</td>
<td>3062.8</td>
</tr>
<tr>
<td>PCA+KNN</td>
<td>69.1</td>
<td>1090.3</td>
</tr>
<tr>
<td>LR</td>
<td>42220.7</td>
<td>31.2</td>
</tr>
<tr>
<td>PCA+LR</td>
<td>21760.4</td>
<td>15.6</td>
</tr>
</tbody>
</table>

Figure 12. Training time of various algorithms

Figure 13. Prediction time of various algorithms
6.3 Discussion
This section presents the experimental results of the proposed and the existing learning-based face detection schemes. It also provides a discussion on the performance of the schemes. For multiclass classification problems like face recognition, Support Vector Machine (SVM), K-Nearest Neighbour (KNN), and Logistic Regression (LR) perform well, in this experiment we will compare the result of the above-mentioned classifiers with normal and PCA reduced dataset. The obtained dataset of 6459 face encodings is now split into a training set of 4869 encodings and a testing set of 1590 encodings.

6.3.1 Support Vector Machine
By applying grid search on hyper-parameters cost, gamma and kernel, and m-fold cross-validation we can obtain the fine-tuned values of the hyper-parameters of the Support Vector Classifier, after this the model is then trained on the training set the accuracy of 98.9% and precision of 99.2% is obtained on the test set. The training time after dimensional reduction using PCA is reduced by 41.35% and prediction time is reduced by 5.23%.

6.3.2 K-Nearest Neighbour
By applying grid search on hyper-parameters metric, n_neighbors and weights, and m-fold cross-validation we can obtain the fine-tuned values of hyper-parameters of Logistic Regressor, the model is then trained on the training set the accuracy of 98.5% and precision of 99% is obtained on the test set. The training time after dimensional reduction using PCA is reduced by 68.03% and prediction time is reduced by 64.4%.

6.3.3 Logistic Regression
By applying grid search on hyper-parameter cost, and m-fold cross-validation we can obtain the fine-tuned values of hyper-parameters of Logistic Regressor, the model is then trained on the training set the accuracy of 97.8% and precision of 98.7% is obtained on the test set. The training time after dimensional reduction using PCA is reduced by 48.46% and prediction time is reduced by 50%.

7. CONCLUSION AND FUTURE DIRECTIONS
This paper presents an experimental assessment of the face recognition system using Dlib library, ResNet. In this experiment, dimensionality reduction is done using PCA which provides the maximum reduction of computational time by 68.03%. The proposed algorithm is tested on a custom-created database of 6800 faces of 657 people with 3 different multiclass classifiers. The best performing classifier in terms of accuracy is the support vector machine with 99.6% accuracy on the validation dataset of 227 faces of 104 people non-overlapping with the original dataset. However, in terms of computational time, the best performing classifier is KNN with an accuracy of 99.1%. So, for the real-time application like surveillance KNN classifier works best in terms of computational time and accuracy with a slight trade-off of accuracy.
In the future, this face recognition system can be implemented using GPU. The rate at which the face feature extraction and feature vector calculation are done can be drastically improved using GPU processing powers. We can also use the convolutional neural networks for the face detection process.

ACKNOWLEDGMENT

This research is partially funded by the Technical Education Quality Improvement Program (TEQIP III).
REFERENCES


Rajendra Kumar Dwivedi is an Assistant Professor in the Department of Information Technology and Computer Applications at Madan Mohan Malaviya University of Technology, Gorakhpur (U.P.), India. He joined this institute in 2009. He received his B. Tech Degree in 2004 from Pt Ravishanker Shukla University, Raipur and M.Tech. from Indian Institute of Technology, Roorkee in 2015. He has done his PhD from Madan Mohan Malaviya University of Technology, Gorakhpur (U.P.) in 2021. Before joining Madan Mohan Malaviya Engineering College (under state government of U.P.), he worked in K.V. Lansdowne U.K. (under central government of India). He has supervised a large number of M. Tech. students. He has published a large number of research papers in various international and national journals and conferences of high repute (h-index=9, i-10 index=9, citations=256). He is a member of IEEE and also life member of Institution of Engineers (India). His main research interests lie in Wireless Sensor Networks, Network Security, Cloud Computing, Internet of Things, Blockchain, Machine Learning, and Deep Learning.

Devesh Kumar is a Post Graduate student in the Department of Computer Science at Babasaheb Bhimrao Ambedkar University, Lucknow, Uttar Pradesh, India. He is doing a collaborative project with Madan Mohan Malaviya University of Technology, Gorakhpur, Uttar Pradesh, India. His main research interests lie in Wireless Sensor Networks, Network Security, Cloud Computing, Internet of Things, Blockchain, Computer Vision, Machine Learning, and Deep Learning.