Face recognition system in Python

***Abstract*—The aim of this research is to explore the application of machine learning in the field of people identity management for security purposes. In this paper, we present a comparative analysis of two classifiers, namely Support Vector Machines (SVM) and K-Nearest Neighbors, for face recognition using Local Binary Patterns (LBP) feature extraction technique. The database comprises 40 individuals, each having 10 images with different poses and facial expressions. Two-fold cross-validation is employed for evaluating the classifiers, and performance metrics such as accuracy, False Acceptance Rate, and False Rejection Rate are computed for each fold. The optimal parameters for SVM are determined using GridSearchCV. The findings suggest that the LBP-SVM technique achieves an accuracy of 95% and FAR of 0.12% in the best-case scenario. However, the study highlights that the accuracy can be further enhanced with larger and more diverse datasets.**

***Index Terms*—Face recognition, machine learning, security control, identification.**

I. Introduction

Facial recognition is a process that involves the analysis of facial features to identify individuals. It has become essential in recent years due to its security, law enforcement, and smartphone authentication applications [1]. Given the potential consequences of falsely identifying an individual, it is crucial to develop accurate algorithms for facial recognition. [2].

The facial recognition process comprises several stages [3] that utilize algorithms to recognize individuals based on facial features. Initially, the system detects the presence of a face in an image or video frame. The detected face is then processed by filtering and enhancement techniques to improve the image's quality. The next stage is feature extraction, where the facial features are identified and represented as a feature vector. This vector captures the unique characteristics of the face, which are used to classify the face into a specific identity.

The system requires training on a set of images to achieve accurate classification. During training, the system learns to identify the unique features of each individual face in the gallery. The facial recognition classifier can range from a simple binary decision of 'match' or 'no-match' to complex machine learning models. In practice, the classifier compares the feature vector of a probe image to the gallery templates to determine whether there is a match. Various factors, such as image quality, feature extraction and classification algorithms, and gallery size, influence the classifier's effectiveness.

Facial recognition involves two critical stages: feature extraction and classification. Researchers have developed and evaluated various methods to assess the effectiveness of different extraction and classification methods. However, many of these methods have suboptimal performance and high computational complexity, resulting in a high false detection rate [4]. Principal Component Analysis, SURF, SIFT, and Gabor Filter are examples of such methods. For our study, we selected the Local Binary Patterns (LBP) filter [5]. The LBP descriptor is employed to extract features due to its computational efficiency and robustness to monotonic grayscale changes.

This project aims to compare the performance of these algorithms in recognizing faces. Furthermore, this project provides the accuracy and effectiveness of different face classifiers by highlighting the biometric system of face recognition with the help of LBP, Support Vector Machines (SVM), and k-nearest Neighbours (KNN).

II. Methodology

This section outlines our proposed approach to developing a facial recognition program in Python. Our approach leverages the LBP technique and SVM and KNN for a comparative analysis of their effectiveness in facial recognition. The visual representation depicted in Figure 1 illustrates the procedural steps undertaken in this project.

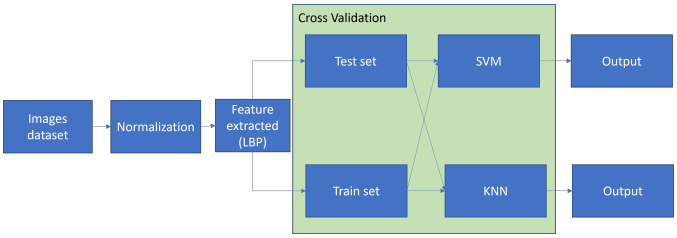

Fig. 1. Project Flow Diagram

A. Data set

The input database for this project was chosen to be the AT&T dataset [6], which includes grayscale images of 40 individuals with 10 images per individual. Since the AT&T dataset comprises images already cropped to show only the face, further face-detection steps were considered unnecessary. Although the images have a uniform size of 92x112 pixels and 256 grey levels per pixel, they vary in angles, expressions, and lighting. Figure 2 provides an illustration of a dataset sample corresponding to person #20.

The dataset was divided into two subsets for training and testing purposes, where 5 images per individual were used for training, and the other 5 images were used for testing. The steps involved in splitting the dataset are described in detail in section III.

B. Prerocessing

A normalization process has been implemented to standardize the intensity values of the grayscale image dataset utilized in this project. The process involves a linear transformation of the input image I, which has intensity values within the range

Fig. 2. Dataset sample for person #20

Min, Max, to a new image $Inew$, which has intensity values within the range $newMin, newMax$. This normalization step is the only preprocessing step required for the dataset.

The mathematical expression for the normalization process is as follows [7]:

$$Inew = (I - Min)\frac{(newMax - newMin)}{(Max - Min)} + newMin$$

In the above formula, $Inew$ denotes the new normalized image, I is the original image, Min and Max represent the minimum and maximum intensity values in the original image, and $newMin$ and $newMax$ represent the 0 and 255 intensity values in the normalized image, respectively. The formula represents a linear transformation that scales the intensity values of the original image to fit within the desired range of intensity values in the normalized image.

C. Feature extraction

The LBP algorithm is the most widely used method for feature extraction in face recognition [8]. It is a texture analysis approach that captures the local texture patterns of an image by comparing the pixel values of the central pixel with its surrounding neighbours. LBP is a simple and computationally efficient algorithm that can withstand lighting conditions, facial expressions, and poses changes. It has been shown to achieve high accuracy in face recognition tasks, making it a popular choice for feature extraction in face recognition systems.

In the LBP algorithm, a circular neighbourhood is first defined around each pixel, and the neighbouring pixels are compared to the central pixel. If the neighbouring pixel has a value greater than or equal to that of the central pixel, it is assigned a value of 1, while if it is less than the central pixel, it is assigned a value of 0. This procedure is then repeated for all the pixels within the neighbourhood, generating a binary pattern. Subsequently, the binary values are weighted by powers of two and summed, resulting in the LBP code of the center pixel [9]. Figure 3 illustrates an example of the LBP operator.

Let g_c be the intensity of the central pixel and g_p be the intensity of the p-th neighbouring pixel. Then, the LBP code

Sample			Difference			Threshold		
40	50	60	-26	-16	-6	0	0	0
30	66	70	-36		4	0		1
100	90	80	34	24	14	1	1	1

new value = 1*2^0+1*2^1+1*2^2+1*2^3+0*2^4+0*2^5+0*2^6+0*2^7 = 15

Fig. 3. An example of the LBP operator

for the central pixel is given by [9]:

$$LBP_{P,R}(x_c, y_c) = \sum_{p=0}^{P-1} s(g_p - g_c)2^p$$

Where P is the number of neighboring pixels, R is the radius, (x_c, y_c) represents the center pixel, and s is the step function, defined as:

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

Fig. 4 shows an example of the LBP operator applied to a sample image in our dataset. The left image displays the original image, while the right image shows the LBP image obtained after applying the LBP algorithm. We can observe how the LBP algorithm effectively captures the local texture patterns of the image, which are represented in the resulting binary pattern.

Fig. 4. Image before and after applying LBP

After that, the LBP algorithm calculates the frequency of occurrence of each binary pattern in an image to create a histogram of LBP codes. This histogram represents the unique texture patterns of the face and can be used for face recognition [8].

When creating a histogram of LBP features, the values can be grouped into discrete bins to create a frequency distribution that can be used for further classification. The number of bins used in the histogram determines the level of detail in representing the LBP features.

The selection of the number of bins is crucial as it can affect the discrimination power of the histogram. A small number of bins can result in losing important information, while too

many bins can lead to a sparse and noisy histogram. Therefore, selecting an appropriate number of bins must maintain a balance between information loss and noise.

In our case, the number of bins is set to the maximum possible LBP value plus one, ensuring that all possible values are represented in the histogram. This guarantees that all LBP values are included in the histogram and that the histogram is not truncated due to insufficient bins.

After computing the LBP histograms for the training set, we can use them as a feature vector for further classification.

D. Machine learning

In the context of face recognition, multi-class classification is the task of identifying a person's identity among a set of possible identities, each represented by a class. For example, given a database of face images of 40 individuals, we aim to correctly classify a new input image to its corresponding class, which represents the identity of the person in the image.

One widely used method for face recognition classification is the Support Vector Machine (SVM) algorithm [10], a supervised machine learning algorithm. SVM is trained on a labelled dataset of face images to classify a new face image to a specific individual or class.

A feature vector is used to capture unique facial characteristics to represent a face image in SVM. The algorithm then creates a hyperplane that maximizes the margin between feature vectors of different classes, intending to minimize the classification error on the training dataset. When presented with a new face image, the SVM algorithm computes its feature vector and utilizes the hyperplane to classify it into a specific class or individual. SVM can effectively handle high-dimensional feature spaces and is robust to noise, making it an appropriate choice for face recognition tasks.

The SVM classifier has three parameters [11]: kernel function, C parameter, and gamma parameter. Our objective is to determine the optimal values for these parameters to separate classes within the SVM effectively. The function used to implement this classifier includes only the parameters we are interested in:

- The C parameter regulates the trade-off between correctly classifying training points and smoothing the decision boundary;
- The kernel function selects the type of hyperplane that separates the data and can be linear or non-linear;
- The gamma parameter controls the closeness of fit to the training set and is only relevant for non-linear hyperplanes.

In the field of face recognition, the KNN algorithm is another popular classification method. It is a simple yet effective algorithm that can handle non-linear decision boundaries and can handle noisy data. However, KNN can be computationally expensive and requires a large amount of labeled data to achieve high classification accuracy.

In KNN, a new face image is classified by comparing it to the labeled face images in a training dataset. The algorithm selects K-labeled face images that are closest to the new face image in feature space, where K is a user-defined parameter.

The algorithm assigns the new face image to the class that is most common among the K-selected face images. In case of a tie, the algorithm may use a voting scheme or assign the new face image to the class of the closest face image. The choice of K has a significant impact on the accuracy of the algorithm.

To evaluate the performance of the classifiers, we use a test set to extract LBP histograms and apply the classifier to predict the corresponding class labels. By comparing the predicted labels to the true labels, we can assess the accuracy of the classifier.

III. Results

A. Test/train data set

In this project, the dataset is divided into two distinct groups, namely A and B, with each group consisting of 5 images from each individual. The evaluation of the model is based on 2-fold cross-validation, where each subset A and B is alternately used for training and testing.

Four classifiers are compared, namely SVM-AB, KNN-AB, SVM-BA, and KNN-BA. The SVM-AB and KNN-AB classifiers were trained using the A-group and tested using the B-group, whereas the SVM-BA and KNN-BA classifiers were trained using the B-group and tested using the A-group. The performance of each fold is evaluated by measuring accuracy, FAR, and FRR. However, the confusion matrix is shown for the best-performing classifier.

B. The cross-validation Grid

Optimizing the performance of classifiers in accurately recognizing faces is an important task, and tuning hyperparameters is a crucial step in achieving this objective. A Cross-validation Grid approach was employed using GridSearchCV [12] in Python to optimize the performance of the SVM and KNN classifiers.

For the SVM classifier, a set of hyperparameters was defined in Table I.

TABLE I
THE SVM PARAMETERS FOR CROSS-VALIDATION GRID

C	0.1	1	100	1000
kernel	rbf	poly	sigmoid	linear
gamma	1	0.1	0.01	0.001

Using the Cross-validation Grid approach, the optimal parameters for SVM-AB were determined to be (C=0.1, gamma=1, kernel='linear'), and the optimal parameters for SVM-BA were (C=0.1, gamma=1, kernel='poly').

Similarly, the KNN classifier's K parameter was evaluated with a range of values: [2, 3, 5, 8, 11, 15, 18, 21, 25]. Based on the assessment, the optimal value for KNN was (K=2).

Using GridSearchCV allowed for a systematic and thorough search of the parameter space to find the best parameters that resulted in the highest accuracy and lowest FPR and FNR. The optimal parameters found for SVM and KNN will be used in the final evaluation of the classifiers to determine which is most effective for facial recognition.

C. Evaluation

In facial recognition systems, performance can be evaluated using various metrics, including accuracy, false positive rate (FPR), and false negative rate (FNR) [13].

Accuracy refers to the proportion of correctly classified instances, both true positives and true negatives, divided by the total number of instances. It measures the model's ability to classify data accurately into their respective classes.

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \quad (1)$$

FPR measures the proportion of negative instances incorrectly classified as positive out of the total negative instances. This metric is critical in applications where false positives can have serious consequences.

$$FPR = \frac{FP}{FP + TN} \quad (2)$$

Similarly, FNR measures the proportion of positive instances incorrectly classified as negative out of the total number of positive instances. This metric is also essential in applications where false negatives can result in significant errors.

$$FNR = \frac{FN}{FN + TP} \quad (3)$$

An effective facial recognition model should have high accuracy and low FPR and FNR, indicating that it can accurately classify data into their respective classes while minimizing the number of false predictions.

The results of these evaluations are summarized in Table II.

TABLE II
PERFORMANCE OF THE FACE RECOGNITION CLASSIFIERS

Method	Accuracy %	FPR %	FNR %
SVM-AB	**95**	**0.12**	**4.5**
SVM-BA	90	0.26	10
KNN-AB	79	0.53	20.5
KNN-BA	83	0.44	17

Based on the results of the four classifiers, it can be concluded that SVM-AB and SVM-BA have the highest accuracy scores, with 95% and 90%, respectively. Both classifiers have relatively low FPR. However, the FNR of SVM-BA is higher than SVM-AB, with 10% compared to 4.5%.

KNN-AB and KNN-BA have lower accuracy scores compared to SVM. Also, both KNNs have a relatively high FPR and FPR.

SVM-AB is the best classifier for accuracy and false positive rate while maintaining a relatively low false negative rate. This indicates that SVM-AB can correctly classify a high percentage of the face images in the test set while minimizing false positives. This is important in applications where false positives can have serious consequences, such as security systems.

A confusion matrix is also presented for SVM-AB in Figure 5. The confusion matrix shows the number of instances correctly classified and misclassified by the model for each class.

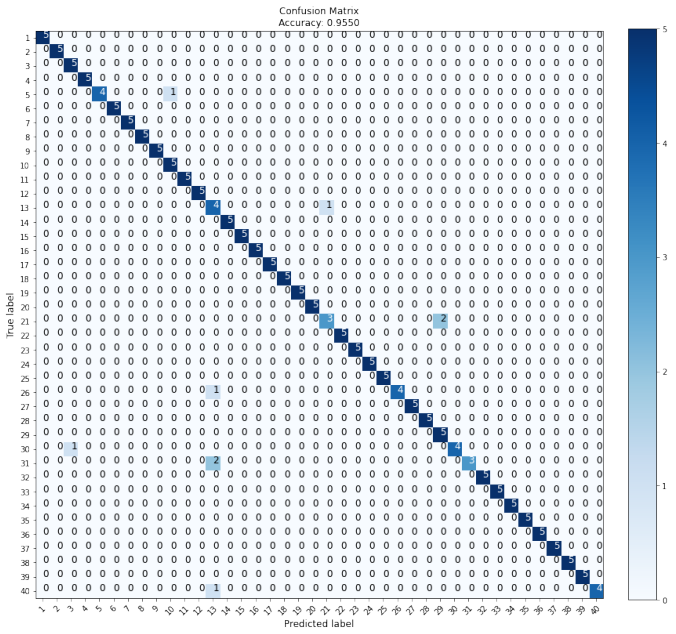

Fig. 5. Confusion matrix for SVM with set A as training set and B as testing set

The results indicate that the classifier performed well and correctly predicted most classes. However, some classes were where the classifier needed to have corrected predictions. Out of the 40 classes, only seven classes had incorrect predictions. However, the accuracy for two of these classes was only 60% (correctly recognizing three out of five images), while for the other five classes, the accuracy was 80% (only one image out of five was misclassified).

IV. Conclusion and Discussion

This study presented an approach for facial recognition, which utilized LBP feature extraction and compared the performance of two classifiers: SVM and KNN. Facial recognition has gained significant interest due to its potential security, surveillance, and human-computer interaction applications.

Hyperparameter tuning is an essential step in training a classifier model. In this study, multiple models were trained with cross-validation using various C and gamma regularization parameters for the SVM model and different numbers of neighbours for KNN.

However, the study also identified limitations and areas for improvement. The small sample size of the database, consisting of only 40 individuals with 10 images each, is a limitation. A good training set with diversity and representation of real-world scenarios is crucial for developing accurate face recognition systems. Without a diverse training set, the accuracy of the recognition system may be limited in real-world scenarios with many facial variations. Updating and

expanding the training set is necessary to incorporate new variations and improve system accuracy over time.

The confusion matrix (Fig. 5) provides a visual summary of the model's predicted outcomes in relation to the actual ground truth. From the confusion matrix, it is noted that specific individuals were assigned to class number 13 in three out of the seven cases. This suggests that the algorithm identified some similarities among them, which could cause confusion in future comparisons.

Figure 6 displays a visual comparison of class 13 with classes 40, 31, and 26, where the confusion was observed, and highlights the images that were inaccurately assigned to class 13 using red rectangles.

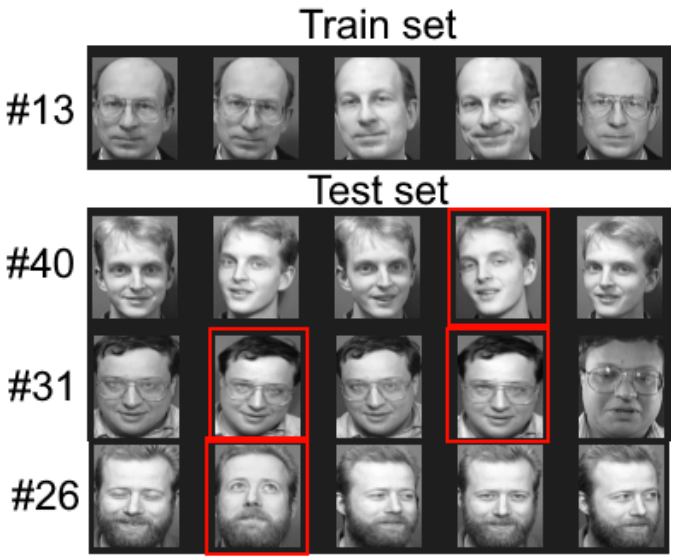

Fig. 6. NAME

During the visual inspection of the images, identifying the underlying reason that prompted the algorithm's classification of these images into the same category was challenging. It remains unclear what specific features the algorithm uses to infer and classify objects into the same category.

For this reason, future studies should consider the chance of false positive rates to avoid confusion and improve the accuracy of the classifier.

The study shows that LBP-based facial recognition combined with the SVM classifier can achieve high accuracy and low false positive and false negative rates. However, there is still room for improvement. Future work could explore other feature extraction techniques and machine learning algorithms to improve the performance of facial recognition systems further. It is also important to note that the algorithm's decision-making process and the features corresponding to the division of objects into different classes require further investigation.

ACKNOWLEDGMENT

REFERENCES

[1] L. Li, X. Mu, S. Li, and H. Peng, "A review of face recognition technology," *IEEE Access*, vol. 8, pp. 139 110–139 120, 2020.

[2] M. Owayjan, A. Dergham, G. Haber, N. Fakih, A. Hamoush, and E. Abdo, "Face Recognition Security System," in *New Trends in Networking, Computing, E-learning, Systems Sciences, and Engineering*, ser. Lecture Notes in Electrical Engineering, K. Elleithy and T. Sobh, Eds. Cham: Springer International Publishing, 2015, pp. 343–348.

[3] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[4] Y. Kortli, M. Jridi, A. A. Falou, and M. Atri, "A novel face detection approach using local binary pattern histogram and support vector machine," in *2018 International Conference on Advanced Systems and Electric Technologies (IC_ASET)*, Mar. 2018, pp. 28–33.

[5] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face Recognition with Local Binary Patterns," in *Computer Vision - ECCV 2004*, ser. Lecture Notes in Computer Science, T. Pajdla and J. Matas, Eds. Berlin, Heidelberg: Springer, 2004, pp. 469–481.

[6] "AT&T Database of Faces — Kaggle," https://www.kaggle.com/datasets/kasikrit/att-database-of-faces.

[7] D. Singh and B. Singh, "Investigating the impact of data normalization on classification performance," *Applied Soft Computing*, vol. 97, p. 105524, 2020.

[8] B. Yang and S. Chen, "A comparative study on local binary pattern (LBP) based face recognition: LBP histogram versus LBP image," *Neurocomputing*, vol. 120, pp. 365–379, Nov. 2013.

[9] "Texture Analysis with Local Binary Pattern — Handbook of Pattern Recognition and Computer Vision," https://www.worldscientific.com/doi/abs/10.1142/9789812775320_0011.

[10] Y. Luo, C.-m. Wu, and Y. Zhang, "Facial expression recognition based on fusion feature of PCA and LBP with SVM," *Optik - International Journal for Light and Electron Optics*, vol. 124, no. 17, pp. 2767–2770, Sep. 2013.

[11] "Sklearn.svm.SVC," https://scikit-learn/stable/modules/generated/sklearn-svm-SVC.html.

[12] "Sklearn.model_selection.GridSearchCV," https://scikit-learn/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

[13] A. Jain, R. Bolle, and S. Pankanti, *Introduction to biometrics*. Springer, 1996.