



Effective Machine Learning Schemes for Face Recognition in Higher-Dimensional Datasets

Onibeju Basit Adeyinka¹, Akinyemi Moruff Oyelakin^{2*}

^{1,2}Department of Computer Science, College of Information and Communication Technology,
Crescent University, Abeokuta, Nigeria

Email: ¹onibeju.basit.adeyinka@oilservltd-ng.com, ²moruff.oyelakin@cuab.edu.ng

Received Feb 14th 2026; Revised May 13th 2026; Accepted Jul 13th 2026; Available Online Jul 31th 2026

Corresponding Author: Akinyemi Moruff Oyelakin

Copyright © 2026 by Authors, Published by Institut Riset dan Publikasi Indonesia (IRPI)

Abstract

Face recognition systems play a vital role in personal identification, particularly in security applications. However, high-dimensional image data often reduces the effectiveness of machine learning (ML) models by increasing computational complexity and training difficulty. This study aims to improve face recognition performance by combining dimensionality reduction and deep feature extraction techniques. Two benchmark datasets, Filtered LFW and a custom VGGFace2 dataset, were used to evaluate two ML-based models. The first model integrates Linear Discriminant Analysis (LDA), CNN (ResNet18), and K-Nearest Neighbors (KNN), while the second combines CNN (ResNet18) with Naïve Bayes (NB). Both models were trained, tested, and evaluated using accuracy, precision, recall, F1-score, and execution time. On the LFW dataset, the CNN-LDA-KNN model achieved the best performance, with 97.82% accuracy, 0.9799 precision, 0.9782 recall, 0.9782 F1-score, and 68.85 seconds evaluation time, outperforming the CNN-NB model (96.71% accuracy). Similarly, on the VGGFace2 dataset, CNN-LDA-KNN obtained 93.46% accuracy compared with 78.87% for CNN-NB. These findings demonstrate that integrating LDA with CNN-based feature extraction and KNN classification significantly enhances face recognition performance on high-dimensional image datasets while maintaining competitive computational efficiency.

Keywords: Dimensionality Reduction, Face Recognition, Feature Extraction, Machine Learning

1. INTRODUCTION

Facial recognition systems have been widely used as biometrics in laptops, mobile phones, and related devices. These systems have also been found to be very useful in border control and attendance monitoring [1]. Owing to recent technological advancements, many organizations use face recognition systems for access security, surveillance, home security automation, and criminal identification. Face recognition software works by comparing the person's facial and geometric features. The traditional method uses a database to store individuals' images, which are then compared with captured images. In such a system, various features like edges, corners, texture descriptors, etc. were extracted and used as facial features. These features would then be used along with machine learning algorithms for the classification. Some of the challenges with traditional methods include variability in facial poses, aging, lighting conditions, and facial changes with expression.

Authors in [2] pointed out that in the field of digital image recognition, a common problem is the high dimensionality of input data (images). In ML-based studies, data is needed to train and test models in different domains. For these ML-based models to generalize well, large amounts of representative data from the problem domain are always required. Researchers in [3] noted that, in this context, the data is, mathematically, a subset $A \subseteq \mathbb{R}^d$, where the dimension d is typically very large. Thus, an important issue in a face recognition system is the execution speed of the algorithm when identifying the characteristics of face images. For this purpose, one can argue that the dimensions of the feature vectors in face image data should be sufficiently small, especially when the database is large. Since the face recognition system must operate on a wide range of databases, dimensionality reduction techniques are required to reduce processing time and increase accuracy. That is, different dimensionality reduction methods are generally used for this purpose as mentioned by [4]. A feature in ML-based studies is an input attribute that serves as representative information extracted from the raw dataset [5]. That is why techniques such as feature extraction, feature selection, and dimensionality reduction have proven very useful in Machine Learning problems across many domains.

Although past studies have shown that integrating facial recognition with dimensionality reduction techniques such as PCA and LDA significantly enhances system efficiency and accuracy, challenges such as



execution speed and high data dimensionality still persist. Addressing these issues through advanced methodologies such as feature extraction and machine learning algorithms ensures better system performance. In the paper, two multi-stage ML models were built, and their performances were evaluated against each other. The results were also benchmarked with a recent similar study.

2. MATERIAL AND METHOD

This study used a multi-stage ML approach to build models that can effectively recognize face images. KNN and NB classifiers were chosen at the classification stage. The study is using the argument of the researchers in [2] about the dimensionality problem that is most common in input image data, it emphasizes the use of LDA to solve the curse of dimensionality in the first model. CNN was applied as a feature extractor in the two models built. The study further investigates how the two models behave by using the same feature extractor, with and without LA, before face classification. This is more important because KNN generally becomes slower when the data is large and the dimensionality issue is not addressed. Based on the rationale put forward by the authors in [2], reducing the dimensionality of image data before face recognition by KNN is very important in this study because it simplifies the data, removes redundancy, and enables faster, more accurate face detection. For the second model, an NB classifier is used immediately after the CNN is applied as a feature extractor.

The key steps in the two models built in this study are represented pictorially using Figures 1 and 2, respectively.

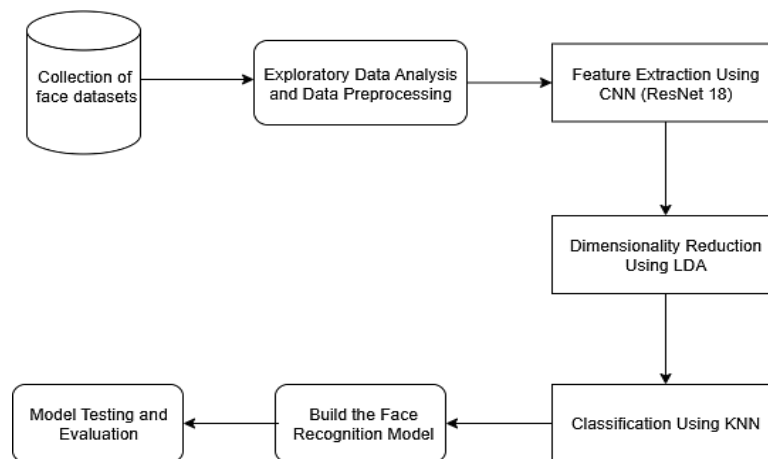


Figure 1. Methodological Workflow diagram for Model I

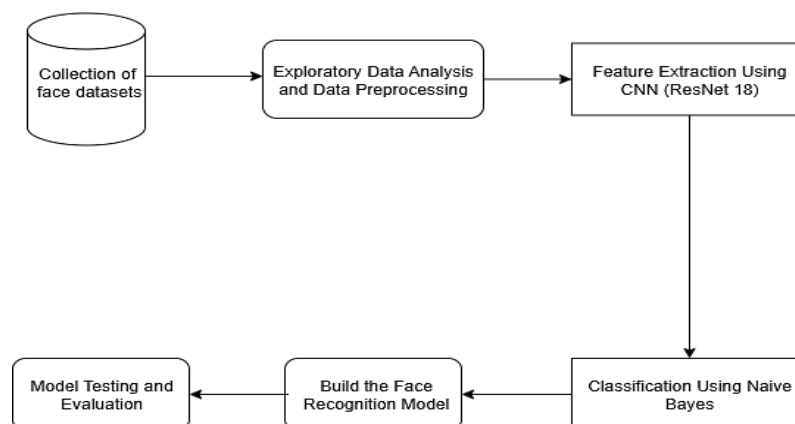


Figure 2. Methodological workflow diagram for Model II

2.1. Collection of Face Recognition Datasets

The datasets used in this study are the FW (Labeled Faces in the Wild) and VGGFace2 datasets. The LFW dataset was obtained from <https://www.kaggle.com/datasets/marvinluckianto/labelled-faces-in-the-wild-LFW> as released by the researchers in [13] [10]. The dataset was last updated in 2024, making it more recent, therefore less outdated. The dataset contains 13000 images of size 250 x 250. The second dataset was a training split from VGGFace2, sourced from <https://www.kaggle.com/datasets/hearfool/vggface2>, as provided by [14] [11].

2.2. About the LFW

The LFW dataset contains 13,000 images of over 5,000 people. The images were collected all over the internet. The faces are already centered by the Viola-Jones algorithm. The dataset is suitable for supervised learning because it is already labeled. Some of the images from the dataset are shown in Figure 3.

2.3. About the VGGFace2

The VGGFace 2 dataset is very large and has been shown to be effective for face recognition studies as well. The few selected faces from the dataset during the EDA are shown in Figure 4.



Figure 3. Sample images from the LFW dataset



Figure 4. Sample images from the VGGFace2 Dataset

2.4. Data Preprocessing Activities

The dataset was pre-processed based on issues identified during the exploratory analysis stage.

2.5 Training and Testing Set for the Face Recognition Task

The two datasets were split 80% for training and 20% for testing. The training set was used to train the model, while the testing set was used to evaluate the trained model. The 80% split ensures that models have sufficient data to learn and classify face images.

2.6 MODEL I: A Multi-Stage Hybrid Model of CNN with LDA and KNN

2.6.1 Convolutional Neural Network

Researchers in [5] have identified the importance of feature extraction in every Machine learning task in different domains. ResNet18 was selected for the feature extraction in this study due to its balance between representational power and computational efficiency, making it suitable for training on a CPU. The final classification layer of ResNet18 was replaced with a temporary fully connected (dummy) layer to enable supervised training. Both datasets were extracted using two different approaches: the LFW dataset was trained with the fourth convolutional block unfrozen, while the custom VGG dataset was trained with all four blocks unfrozen. After training for a small number of epochs, the dummy classifier was removed, and the extracted 512-dimensional feature vectors were passed to Linear Discriminant Analysis (LDA) for dimensionality reduction, followed by K-Nearest Neighbors (KNN) for classification.

2.6.2 Linear Discriminant Analysis (LDA)

Linear discriminant analysis is a statistical method that is used to separate a group of objects by finding linear combinations of features. LDA is used for dimensionality reduction in this study.

2.6.3 K-Nearest Neighbors (KNN)

KNN is a supervised machine-learning technique used for both regression and classification. It is a lazy learning algorithm that is known for its efficiency. The K in KNN refers to the number of neighbors to consider; choosing an optimal value of k is necessary to build an unbiased model. KNN uses the distance of the nearest neighbors to classify data. However, KNN suffers from the curse of dimensionality, in which meaningful patterns cannot be learned due to overfitting. Thus, LDA was specifically used in this model to tackle this issue. CNN will perform automatic feature selection and extraction for the data.

2.7 MODEL II: A Hybrid Model of CNN with Naive Bayes

In the second model, Convolutional Neural Network is used for the feature extraction as applied in the first one and then Naive Bayes is used for the face image recognition. Only the initial layers of the network were utilized, with the fully connected layer replaced with a temporary dummy classifier to allow supervised fine-tuning. Both datasets were trained using different approaches, with only layer 4 unfrozen for the LFW datasets, while for the custom VGG_112_50, all four blocks will be unfrozen.

Naive Bayes is a probabilistic classifier that assumes every feature is independent. This assumption allows it to avoid the curse of dimensionality. In this study, faces are classified by computing the posterior probability of each class given each face's features.

$$P(C|X) = \{P(X|C) P(C)\} / P(X) \quad (1)$$

$P(C)$ is the probability of each face occurring in the dataset. $P(X|C)$ is the likelihood of observing a feature vector X knowing that it belongs to a class C . Naive Bayes assumes feature independence, which is computed as the repeated product of each observed feature.

$$P(X|C) = P(X_1|C) * P(X_2|C) * P(X_3|C) * \dots * P(X_n|C) \quad (2)$$

$P(X)$ is the probability of observing the feature vector X across all classes. It is used as a normalization constant; it is the same for every class when classifying a new instance. $P(X)$ can be omitted when classifying since it does not affect the posterior probability.

2.7 Model Evaluation Metrics

2.7.1 Accuracy

The accuracy of the models is measured by their ability to recognize the correct faces. It is the ratio of true positives to the total of true positives, true negatives, false positives, and false negatives.

$$\text{Accuracy} = \frac{TN+TP}{TN+TP+FP+FN} \quad (3)$$

2.7.2 Precision

Precision is a measure of how accurately the model identifies a face belonging to a particular person.

$$\text{Precision} = \frac{TP}{TP+FP} \quad (4)$$

2.7.3 Recall

Recall is the measure of the correct number of faces the model has predicted successfully.

$$\text{Recall} = \frac{TP}{TP+FN} \quad (5)$$

2.7.4 F1- Score

This is the harmonic mean between precision and recall. It helps to provide a balanced measure between precision and recall.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (6)$$

2.7.5 Training Time

This evaluation metric will be the time required to train a single dataset. It will be measured for both models. The time library in Python will be used to perform this evaluation.

2.7.6 Evaluation Time

After training, the evaluation time was used to assess the datasets and make predictions on the test set.

3. RESULTS AND DISCUSSION

3.1. LFW Dataset

After performing an exploratory analysis of the LFW dataset, the following results were found: The total number of classes is 5729, indicating 5729 distinct identities. The average number of images per class is approximately 2. Classes in this dataset contain as many as 500 images and some as few as 1 image. Precisely 4069 classes contain only a single image, while only 158 classes contain 10 or more images.

The number of images in the dataset was 13233. The dataset had to be modified and tweaked for proper model training. All images were RGB, and a quick check of pixel intensity distribution confirmed no grayscale images were present. Further analysis indicated that there are no corrupted images in the dataset. The figure below displays the disparity in the number of images per class of the LFW dataset.

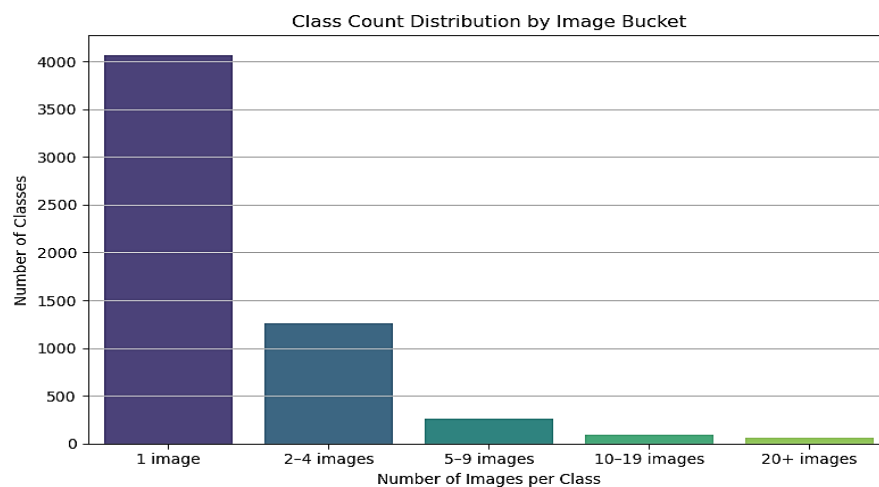


Figure 5. Class Count distribution by Image Bucket

3.2. Custom VGGFace2 Dataset

The results of exploratory analysis on the custom VGGFace2 datasets are as follows: The dataset contains 176398 images with 480 different faces. The average number of images per class is 367.5, and the most common image size is 133 x 152, with only 7 images having this size. Other images have varying sizes; this variation is not ideal, as CNNs prefer consistent images. Each class has a different number of images, with the highest containing 720 and the lowest 102.

3.3. Data Pre-Processing

For both datasets, extensive data preprocessing was performed using the following key activities: Filtering, Resizing, Splitting, Normalization, and Augmentation.

Table 1. Train-test Split of LFW Datasets

Dataset	Classes	Total Images	Train(80%)	Test(20%)	Augmented Train
LFW-10each	158	1580	1264	316	3792
LFW-30each	34	1020	816	204	2448
LFW-50each	12	600	480	120	1440

3.4. Feature Extraction with CNN (ResNet18)

ResNet18 is a convolutional neural network that has been pre-trained on the ImageNet dataset consisting of over 1 million images. ResNet18 learns the features of the datasets then feeds it to the LDA-KNN or NB pipeline. The ResNet18 feature extractor was truncated by removing the final classification layer, retaining the convolutional backbone and the global average pooling layer. The setup produced 512 dimensional feature vector for each image in the datasets.

3.5. Result for Feature Extraction for Custom VGG Dataset

Feature extraction is performed across all layers except the fully connected layer. This method trains the ResNet18 model from scratch, enabling it to learn features from a less-populated dataset. LFW-10each: The dataset was split in an 80:20 ratio corresponding to training and testing, with 1264 images for training and 316 images for testing. The training split was augmented to a total of 3792 images, with the test split remaining the same. A total of 3792 training feature vectors and 316 testing feature vectors were extracted, corresponding to augmented training images and real test images, respectively. These features were then used as input for the LDA-KNN classification pipeline. LFW-30each: For this dataset, 2448 training feature vectors were used along with 204 testing vectors to train the model. LFW-50each: The feature vectors for training are 1440 after augmentation, while for testing, 120 vectors were obtained from non-augmented images.

3.6. Results for Feature Extraction for Custom VGG Dataset

Feature Extraction was carried out by unfreezing four layers in the ResNet18 architecture. This allowed the network to fine-tune its higher-level features to better represent the structure of the custom VGG dataset, while preserving the lower-level features learned from ImageNet pre-training. The ResNet18 model was not trained from scratch; instead, transfer learning was employed to improve training speed and reduce computational overhead.

3.7. Model Building

The CNN-LDA-KNN model uses ResNet18, LDA, and KNN. The LDA served as a dimensionality-reduction algorithm, with KNN performing the classification. The ResNet18 network produces a 512-dimensional feature vector for each image, which is then reduced by LDA to (number of classes - 1) dimensions. The KNN classifier uses the cosine distance metric and predicts labels based on the 5 nearest neighbors in the reduced space.

The CNN-NB model retains the same feature extractor, followed by Gaussian Naive Bayes, for direct classification with no dimensionality reduction. The two datasets used the ResNet18 architecture for feature extraction, but some modifications were made to suit each dataset. The LFW datasets used only the fourth convolutional block to fine-tune the smaller image size, while the rest of the network remained frozen to retrain pre-trained weights. In contrast, for the larger VGGFace2 dataset, four layers were unfrozen, allowing the model to adapt more extensively to the dataset's more complex and varied features.

3.8. Model Testing

This section highlights the test accuracy obtained from the initial training on the dataset; the models are saved for easy reuse, and the final evaluation is performed afterward.

3.9. Face Recognition Results based on the LFW and VGG Datasets

After training each dataset using the specified pipelines (CNN-LDA-KNN and CNN-NB), respectively, the resulting models were saved within their respective dataset folders. This setup ensures easy reference, reproducibility, and further evaluation. Performance is reported as test accuracy, along with training time, giving insight into both the efficiency and effectiveness of each learning algorithm.

Table 2. Test accuracy and training of the Datasets

Datasets/ Learning Algorithm	CNN-LDA-KNN(Accuracy %)	CNN-LDA-KNN (Training Time s)	CNN-NB (Accuracy %)	CNN-NB (Training Time s)
LFW-10 each	73.33	2659.44	62.97	2746.68
LFW- 30each	94.12	1616.74	90.20	1502.16
LFW-50each	95.00	1034.56	97.50	903.08
VGG_112_50	82.21	4900	79.00	5905.5

3.10. Model Evaluation

This section presents the results of the two face recognition models, evaluated using the chosen metrics. The metrics are accuracy, precision, recall, and f1-score, with training time and evaluation time as additional metrics.

Table 3. Evaluation of The LFW Datasets on Model I(CNN-LDA-KNN)

Datasets/ Performance Metrics	Accuracy (%)	Precision	Recall	F1-Score	Evaluation Time (s)
LFW-10each	94.75	0.9522	0.9475	0.9474	102.02
LFW-30each	99.22	0.9923	0.9922	0.9922	67.44
LFW-50each	99.5	0.9951	0.995	0.995	37.1

This table presents the comprehensive evaluation metrics obtained from testing the already-trained Model I (CNN-LDA-KNN) on the filtered LFW datasets. This includes additional metrics such as precision, recall, f1-score, and evaluation time.

Table 4. Evaluation of the LFW Datasets on Model II(CNN-NB)

Datasets/ Performance Metrics	Accuracy(%)	Precision	Recall	F1-Score	Evaluation Time(s)
LFW-10each	92.59	0.9416	0.9259	0.9289	104.50
LFW-30each	98.04	0.9824	0.9804	0.9807	45.91
LFW-50each	99.50	0.9951	0.9950	0.9950	49.95

Table 4 presents the comprehensive evaluation metrics obtained after testing the already-trained Model II (CNN-NB). Evaluation time (in seconds) reflects the total duration required to generate the identification and compute metrics on the test set. The results obtained for each metric used to evaluate the models are used to establish the effectiveness of the models built in this study.

Table 5. Evaluation of VGG_112_50

Model/ Performance Metrics	Accuracy (%)	Precision	Recall	F1-Score	Evaluation Time(s)
CNN-LDA-KNN	93.46	0.9379	0.9346	0.9339	447.29
CNN-NB	78.87	0.8534	0.7884	0.7992	537.00

3.11. Discussion of Findings

With reference to Table 2, the larger the dataset, the longer it takes to train. VGG_112_50 is significantly larger than the other three; therefore, its training time is much higher than theirs. In summary, the two models achieved accuracies of 94.75% and 92.59%, respectively, on the LFW-10each, while CNN-LDA-KNN performed consistently better across both datasets, underscoring the importance of dimensionality reduction in face recognition studies.

3.12. Impact of Dimensionality Reduction (LDA)

From tables 3-5, it can be seen that the CNN-LDA-KNN pipeline consistently outperforms CNN-NB in terms of overall classification accuracy on the LFW datasets, particularly as the number of images increases. For instance, on the LFW-30each dataset, CNN-LDA-KNN achieved 99.22% accuracy compared to CNN-NB's 98.04%. This suggests that dimensionality reduction via LDA is beneficial in high-dimensional spaces by improving class separability before classification with KNN. Experimental results show that accuracy converges as the number of images increases, suggesting that LDA-KNN performs better on larger datasets than NB.

3.13. Robustness of CNN-LDA-KNN

In Table 5, CNN-LDA-KNN achieved a greater performance of 93.46 compared to CNN-NB's performance of 78.87. This huge performance gap shows the robustness of the CNN-LDA-KNN architecture when applied to more complex datasets. In line with the authors' arguments in [15] and [12] regarding the promise of feature extraction in ML-based studies, using a CNN as a feature extractor in this study yielded better results. The VGG_112_50 dataset exhibits increases in intra-class and inter-class similarity, which simple classification methods, such as Gaussian Naive Bayes, that assume feature independence, may struggle to handle. Thus, it can be argued that integrating LDA into the CNN-LDA-KNN model demonstrates its effectiveness in projecting high-dimensional feature vectors into a lower-dimensional space while preserving class separability. As a result, KNN can make more accurate predictions even in complex scenarios. This demonstrates that the CNN-LDA-KNN pipeline generalizes better and is suitable for the challenges posed by real-world, large-scale face recognition tasks.

3.14. Scalability and Performance

From Table 2, it can be seen that Naive Bayes is faster than LDA-KNN as the dataset size decreases, while LDA-KNN maintains higher classification accuracy. But as the dataset grows, LDA-KNN begins to catch up in speed while remaining more accurate than Naive Bayes. For evaluation time in Table 5, CNN-LDA-KNN is evaluated in less time. Overall, the CNN-LDA-KNN pipeline scales well across datasets of increasing size and complexity, consistently delivering high accuracy and robust metrics. The combination of learned features, dimensionality reduction, and instance-based classification makes it highly adaptable.

3.15. Benchmark Against Other Works

To examine the performance of the models, a benchmark comparison is conducted against a similar study by [6], which used a traditional machine learning pipeline with Principal Component Analysis (PCA) for dimensionality reduction and K-Nearest Neighbors (KNN) for classification. The authors used the LFW dataset and achieved 88% accuracy with $k=5$.

Table 6. Benchmark Against Singh et al. (2024)

Study Model	Feature Type	Dimensionality Reduction	Classifier	Dataset Used	Accuracy (%)	Remarks
This Work (Model I)	Deep features (ResNet18)	LDA	KNN ($k=5$, cosine)	LFW (10 each)	94.75%	Learns semantic, high-level representations Fast but reduced accuracy due to independence assumptions
This Work (Model II)	Deep features (ResNet18)	None	Gaussian Naive Bayes	LFW(10 each)	92.59%	Traditional pipeline, limited generalization
Singh et al. (2024)	Handcrafted (pixel-level)	PCA	KNN ($k=5$)	LFW	88.00	

The CNN-LDA-KNN model developed in this study achieved 94.75% accuracy on the same LFW dataset. The performance gap of over 6 percentage points can be attributed to several key differences:

1. **Feature Representation:** [6] made use of handcrafted, pixel-level features, which lack robustness to variations like pose, lighting, and expression. On the other hand, this model uses features extracted from a pretrained ResNet18 model, which are more semantically rich and invariant to such transformations.
2. **Dimensionality Reduction:** While both studies used dimensionality reduction, this work used Linear Discriminant Analysis (LDA), which is supervised and maximizes class separability, compared to PCA, which is unsupervised and does not account for class labels during projection.
3. **Classification:** Both pipelines used KNN, but Singh et al. made use of Euclidean distance, while this model used cosine distance, which is more suitable for higher-dimensional deep features.
4. **Preprocessing Strategies:** This study filtered and balanced the dataset (LFW-10each, LFW-30each, and LFW-50each) to ensure uniformity and fairness during evaluation, while authors [6] did not report balancing or augmentation strategies. Images in the LFW datasets were not resized or converted to grayscale, unlike what the authors did in [6].

4. CONCLUSION

The study introduced the concept of face recognition and explained how feature extraction and dimensionality reduction can be used to build models that efficiently identify faces. The study further demonstrated the advantages of combining deep convolutional neural networks with two traditional classifiers for the face recognition task in high-dimensional spaces. Two different models were built from the chosen face image datasets. While both models used the same feature extractor (ResNet18), the CNN-LDA-KNN pipeline consistently outperformed in accuracy, generalization, and robustness across multiple filtered LFW datasets and a subset of VGGFace2. The experimental results further showed the effectiveness of dimensionality reduction when meaningfully applied using LDA, particularly when paired with distance-based classifiers (KNN). Furthermore, the results showed that features extracted using the deep learning method achieved better performance than handcrafted features, as evidenced by benchmarking against the authors' work in [6]. This study included the evaluation period to better understand how the model can be made usable in real-world applications. It was found that the time taken by the CNN-LDA-KNN for face identification is shorter than that of the CNN-NB-based model. It is concluded that the ML-based approaches used in this research work are generally effective for the face recognition task when evaluated on unseen image data at test time.

REFERENCES

- [1] C. R. Kumar, N. Saranya, M. Priyadharshini, D. E. Gilchrist, and K. M. Rahman, "Face recognition using CNN and Siamese network," *Measurement: Sensors*, vol. 27, p. 100800, 2023.
- [2] K. Georgy and F. Pawel(2021). "Data Dimensionality Reduction for Face Recognition", retrieved from https://pforczmanski.zut.edu.pl/pub/2004_mgv.pdf on 10th August 2025
- [3] S. A. Wegner (2021). "Lecture Notes on High-Dimensional Data", arXiv:2101.05841v7 [math.FA] 20 Sep 2024
- [4] A. Mousavi, H. Arefanjazi, M. Sadeghi, A. M. Ghahfarokhi, F. Beheshtinejad, and M. M. Masouleh, "Comparison of feature extraction with PCA and LTP methods and investigating the effect of dimensionality reduction in the bat algorithm for face recognition," *International Journal of Robotics and Control Systems*, vol. 3, no. 3, pp. 500–509, 2023. doi: [10.31763/ijrcs.v3i3.1057](https://doi.org/10.31763/ijrcs.v3i3.1057).

- [5] A. M. Oyelakin & Jimoh R.G. (2021), “A Survey of Feature Extraction and Feature Selection Techniques Used in Machine Learning-Based Botnet Detection Schemes”, VAWKUM Transactions on Computer Sciences, 9 (2021),1-7, available at <https://vfast.org/journals/index.php/VTCS/article/view/604/658>
- [6] A. Singh, A. P. Singh, and M. R. Wadyalkar, “Optimizing face recognition with PCA and KNN: A machine learning approach,” *International Journal of Research and Review in Applied Science, Humanities, and Technology*, vol. 1, no. 2, pp. 86–96, 2024. Available: <https://ijrasht.com/>.
- [7] K. Muhamada, D. R. I. M. Setiadi, U. Sudibyo, B. Widjajanto, and A. A. Ojugo, “Exploring machine learning and deep learning techniques for occluded face recognition: A comprehensive survey and comparative analysis,” *Faith: Future Technology Science Journal*, vol. 30, pp. 160–173, 2024. doi: 10.62411/faith.2024-30.
- [8] A. Makandar, S. Kaman, and S. B. Javeriya, “Enhancing face recognition through dimensionality reduction techniques and diverse classifiers,” *LC Int. J. STEM*, vol. 5, no. 1, pp. 36–45, 2024.
- [9] O. N. Kadhim, “Face recognition approach via deep and machine learning,” *Wasit J. Pure Sci.*, vol. 2, no. 3, pp. 52–164, 2023, doi: 10.31185/wjps.194.
- [10] L. Antonio, B. S. Sandi, L. Ivan & T. Nikola (2026). “Face Recognition System Using CLIP and FAISS for Scalable and Real-Time Identification”, *Mathematical and Computational Applications*, 2026, 31(2), 36;<https://doi.org/10.3390/mca31020036ns>.
- [11] S. Asane, S. S. Salve, A. Potdar Athrav, A. Wagh & M. Nilwarn. “Security System Using Face Recognition: Machine Learning Based Approach”, *Journal of Information and Communications Technology: Algorithms, Systems and Applications*, 2025, 1(1),1-14,https://gr-journals.com/Ictasa/pdf/ICT_25304.pdf
- [12] R. Manik Rakhra, S. Tiyas Sarkar & A. A. Arsalan Ahmed Alyas. “Machine learning in human face recognition and age estimation: An in-depth analysis and prospective outlook”, *AIP Conf. Proc.* 3185, 020079,(2025), <https://doi.org/10.1063/5.0240417>
- [13] M. Luckianto(2023), “Labelled Faces in the Wild (LFW) Dataset”, retrieved from <https://www.kaggle.com/datasets/marvinluckianto/labelled-faces-in-the-wild-LFW-dataset> on 27th April 2025.
- [14] H. Fool and L. Fool (2022). VGGFace2, retrieved from <https://www.kaggle.com/datasets/hearfool/vggface2> on 11 May, 2025
- [15] K. Samina, K. Tehmina and N. Shaomila.” A Survey of Feature Selection and Feature Extraction Techniques in Machine Learning”, 2014 Science and Information Conference,” *Sci. Inf. Conf.*, 2014.