

A REPORT
ON
**FACE VERIFICATION MODEL
USING 2DPCA FEATURE
EXTRACTION AND ESRGAN
UPSCALING.**

BY

Apoorva Gayatri K

2022A7PS0232H

AT



(Formerly, Loop Reality Pvt Ltd)

A Practice School-I Station of



Birla Institute of Technology And Science Pilani, Hyderabad Campus
JULY 2024

A REPORT
ON
**FACE VERIFICATION MODEL
USING 2DPCA FEATURE
EXTRACTION AND ESRGAN
UPSCALING.**

BY

Name of student:

Apoorva Gayatri K

ID:

2022A7PS0232H

Discipline:

Computer Science

Prepared in partial fulfillment of
the
Practice School-I
Course No. BITS C221

AT

PerspectAI

(Formerly, Loop Reality Pvt Ltd)

A Practice School-I Station of

Birla Institute of Technology And Science Pilani, Hyderabad Campus

JULY 2024

Acknowledgements

I am deeply grateful to the Practice School (PS) division of BITS Pilani for this invaluable opportunity, which has significantly expanded my skill set. My sincere thanks to Dr. Soumyo Mukherji, Director of BITS Pilani, Hyderabad Campus, for his support. I also extend my gratitude to Loop Reality Pvt Ltd (Perspect AI) for hiring me as a summer intern and fostering an environment conducive to learning both hard and soft skills.

I am especially thankful to my faculty-in-charge, Professor Shantanu Prasad Dutta, for his dedication in conducting evaluations, clarifying our doubts, and connecting us with mentors at Loop Reality. He was always understanding, ensured we never felt pressured, and guided us with patience and care. His consistent support has greatly enhanced my experience at the company.

I appreciate my fellow team members whose passion, collaborative spirit has been ever inspiring

During my internship at Loop Reality Pvt Ltd (Perspect AI), I was mentored by Parijat Chatterjee, who guided us throughout the project and was always available to answer our questions. I have learnt a lot from him during this internship and I am immensely thankful for his time and effort.

Additionally, I would like to thank Mr. Suraj Vanka, the CTO, for providing this opportunity.

Lastly, I extend my heartfelt gratitude to my family and friends for their understanding, encouragement, and patience throughout this project. My mother, in particular, has always encouraged me to think outside the box and was the one who introduced me to the world of machine learning and deep learning. Their unwavering support has been a constant source of motivation for me.

The contributions of each individual mentioned above have been instrumental in shaping this report and enriching my learning experience. I am deeply thankful for their involvement and support.

Birla Institute of Technology And Science Pilani, Hyderabad
Practice School Division

Station: Loop Reality Pvt Ltd

Centre Online

Duration 56 Days

Date of Start 28.05.2024

Date of Submission 21.07.2024

Title of the Project: Face verification model using 2dpca feature extraction and ESRGAN UPSCALING.

ID No. 2022A7PS0232H

Discipline of
the student Computer Science

Name and designation of the expert: Parijat Chatterjee, AI/ML engineer @PerspectAI

Name of the PS Faculty: Professor Shantanu Prasad Dutta

Key Words: Face Verification, ESRGAN, 2DPCA, MLP Classification, Binary Classification

Project Areas: AI/ML

Abstract:

This report presents a comprehensive study on the performance of a face verification system utilizing ESRGAN upscaling and 2DPCA feature extraction, evaluated using multiple classifiers. The system demonstrates exceptional capability in distinguishing between two classes, achieving an AUC-ROC of 0.93540 and an Equal Error Rate (EER) of 4.438%. Detailed performance metrics reveal that the MLP classifier achieved an overall accuracy of 0.94 across 8112 samples, with a precision of 0.95, recall of 0.92, and F1-score of 0.93 for class 0, and a precision of 0.92, recall of 0.96, and F1-score of 0.94 for class 1. The dataset was meticulously curated to ensure equal pairs of positive and negative samples across training, testing, and validation sets. Positive pairs were generated from individual folders, while negative pairs were created by selecting random images from different folders, ensuring balance through random sampling.

The model was tested using various classifiers, including MLP, Random Forest, Gradient Boosting Machine, and KNN, with hyperparameter tuning applied to optimize performance. Notably, the inclusion of ESRGAN upscaling resulted in a 1% increase in accuracy, particularly when images were resized to 128x128 pixels. This enhancement underscores the potential benefits of high-resolution image data.

Given the lightweight nature of the model, it shows promise for edge deployment on candidates' devices, reducing server load. The study also explores the impact of diverse training datasets, suggesting that principal component analysis can be tailored to accommodate variations in facial features across different populations. Recommendations for further improving accuracy include training region-specific models and using simple, solid-colored backgrounds to minimize interference with face detection algorithms.

This report concludes that with further refinement and more diverse training data, the proposed system can offer robust and accurate face verification, suitable for various practical applications.



Signature of Student.

Date 21.07.2024

Signature of PS Faculty

Date

TABLE OF CONTENTS

Introduction	1
Preliminary Research	2
Related Work	3
Theorised Model	4
Machine Learning Pipeline	5
Feature Extraction:Visualised	6
Model Comparisons and Benchmark Scores	7
Metrics and Analytics	11
Conclusions and Promising areas for further development	16
References	17
Glossary	18

Introduction

Proctoring, the process of supervising or monitoring interviewees during their assessments, is essential to maintaining the integrity of the evaluation process. In a digital context, this involves capturing images from a webcam stream at periodic intervals to ensure no impersonation occurs. PerspectAI aims to fully automate this process, eliminating the need for human proctors and implementing a high-accuracy face recognition model powered by AI and machine learning algorithms.

Currently, PerspectAI relies on a third-party API for proctoring, achieving an accuracy rate between 75% and 85%. However, the primary goal of the proctoring team is to maximize the accuracy of this model, ensuring more reliable and effective monitoring.

This report presents a comprehensive study on enhancing the performance of the face verification system used in proctoring by leveraging ESRGAN upscaling and 2DPCA feature extraction. We explore various classifiers to identify the most effective one for this application. The proposed system demonstrates significant improvements in distinguishing between two classes, achieving an AUC-ROC of 0.93540 and an Equal Error Rate (EER) of 4.438%.

Detailed performance metrics for the MLP classifier indicate an overall accuracy of 0.94 across 8112 samples, with class 0 achieving a precision of 0.95, recall of 0.92, and F1-score of 0.93, and class 1 attaining a precision of 0.92, recall of 0.96, and F1-score of 0.94. The dataset was carefully curated to ensure balanced pairs of positive and negative samples across training, testing, and validation sets. Positive pairs were generated within individual folders, while negative pairs were created by selecting random images from different folders, balanced through random sampling.

The model was evaluated using multiple classifiers, including MLP, Random Forest, Gradient Boosting Machine, and KNN, with hyperparameter tuning applied to optimize performance. Notably, the incorporation of ESRGAN upscaling resulted in a 1% increase in accuracy, particularly when images were resized to 128x128 pixels. This underscores the potential benefits of high-resolution image data.

The lightweight nature of the model suggests it can be deployed on candidates' devices, reducing server load. Further, we discuss the impact of training on diverse datasets and suggest that principal component analysis can be tailored to accommodate variations in facial features across different populations. Recommendations for improving accuracy include developing region-specific models and using simple, solid-colored backgrounds to minimize interference with face detection algorithms.

In conclusion, this study aims to provide PerspectAI with a robust and accurate face verification system, significantly enhancing their proctoring capabilities and paving the way for fully automated, high-accuracy monitoring solutions.

Preliminary Research

Preliminary research for implementing this model involved garnering knowledge of AI/ML based proctoring system and Face Recognition models

- M. Ghizlane, B. Hicham and F. H. Reda, "A New Model of Automatic and Continuous Online Exam Monitoring," 2019 International Conference on Systems of Collaboration Big Data, Internet of Things & Security (SysCoBIoTS), Casablanca, Morocco, 2019, pp. 1-5, doi: 10.1109/SysCoBIoTS48768.2019.9028027. keywords: {Authentication;Monitoring;Face recognition;Servers;Smart cards;Face;Online exam;continuous authentication;machine learning;automatic monitoring},
- Slusky, Ludwig (2020) "Cybersecurity of Online Proctoring Systems," Journal of International Technology and Information Management: Vol. 29: Iss. 1, Article 3.DOI: [Goal 2](#)
- N. H. Barnouti, M. H. N. Al-Mayyahi and S. S. M. Al-Dabbagh, "Real-Time Face Tracking and Recognition System Using Kanade-Lucas-Tomasi and Two-Dimensional Principal Component Analysis," 2018 International Conference on Advanced Science and Engineering (ICOASE), Duhok, Iraq, 2018, pp. 24-29, doi: 10.1109/ICOASE.2018.8548818. keywords: {Face;Face recognition;Feature extraction;Webcams;Face detection;Classification algorithms;Training;Face Detection;Face Tracking;Face Recognition;Viola-Jones;KLT;2DPCA.},
- R. S. V. Raj, S. A. Narayanan and K. Bijlani, "Heuristic-Based Automatic Online Proctoring System," 2015 IEEE 15th International Conference on Advanced Learning Technologies, Hualien, Taiwan, 2015, pp. 458-459, doi: 10.1109/ICALT.2015.127. keywords: {Conferences;E-learning;Proctoring system;Face Detection;Inference system;On-line examination},
- Aurelia, S., Thanuja, R., Chowdhury, S. et al. AI-based online proctoring: a review of the state-of-the-art techniques and open challenges. *Multimed Tools Appl* 83, 31805–31827 (2024). <https://doi.org/10.1007/s11042-023-16714-x>
- C. Ledig et al., "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 105-114, doi: 10.1109/CVPR.2017.19. keywords: {Image resolution;Signal resolution;Gallium nitride;Image reconstruction;Manifolds;Training;Network architecture},
- Wang, X. et al. (2019). ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. In: Leal-Taixé, L., Roth, S. (eds) *Computer Vision – ECCV 2018 Workshops. ECCV 2018. Lecture Notes in Computer Science()*, vol 11133. Springer, Cham. https://doi.org/10.1007/978-3-030-11021-5_5

Related Work

Proposed model draws inspiration from the work of the following researchers:

- C. Song, Z. He, Y. Yu and Z. Zhang, "Low Resolution Face Recognition System Based on ESRGAN," 2021 3rd International Conference on Applied Machine Learning (ICAML), Changsha, China, 2021, pp. 76-79, doi: 10.1109/ICAML54311.2021.00024. keywords: {Image recognition;Art;Databases;Face recognition;Superresolution;Machine learning;Generative adversarial networks;Face detection;Face Recognition;YOLO;Super-resolution reconstruction;ESRGAN},
- Yang, Jian & Zhang, David & Frangi, Alejandro & Yang, Jing-yu. (2004). Two-Dimensional PCA: A New Approach to Appearance-Based Face Representation and Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence. 26. 10.1109/TPAMI.2004.1261097.
- Koch, G.R. (2015). Siamese Neural Networks for One-Shot Image Recognition.
- V. D. M. Nhat and S. Lee, "Kernel-based 2DPCA for Face Recognition," 2007 IEEE International Symposium on Signal Processing and Information Technology, Giza, Egypt, 2007, pp. 35-39, doi: 10.1109/ISSPIT.2007.4458104. keywords: {Face recognition;Principal component analysis;Covariance matrix;Kernel;Independent component analysis;Feature extraction;Face detection;Lighting;Signal processing;Information technology;PCA;Kernel PCA;2DPCA;Face Recognition},
- Turk, M.A. and Pentland, A.P., 1991, January. Face recognition using eigenfaces. In Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition (pp. 586-587). IEEE Computer Society.

Theorised Model

- The theorised model aims to build the lightest weight face recognition model with an accuracy better than the present deployed model of PerspectAI that gives 75-85% accuracy.
- This could in future also cater to the needs of PerspectAI's goal to deploy the face verification model on the system of the candidate instead of server processing.
- The theorised model also aims to handle images taken by the candidate's webcam, typically of low resolution through upscaling.
- The images are first preprocessed by applying a face detection model of Viola-Jones. This gives us the face Region of Interest or ROI
- The ROI is the upscaled using ESRGAN and resized to a resolution of 64x64 px, also converted to grey scale.
- The model is then trained by taking an equal number of positive pairs and negative pairs of candidates. Generated 21,248 training pairs with 10,624 positives and 10,624 negatives.
- Model also generated 5160 validation pairs with 2580 positives and 2580 negatives; 8112 test pairs with 4056 positives and 4056 negatives
- All pairs generated with corresponding labels '1' if the pair is of images of the same subject and '0' if the pair is of images of different subjects.
- Then we apply 2DPCA to the stack of training images compute the covariance matrix and extract the desired number of top eigen vectors (k=30 diff values of k experimented with, more on page 8)
- Then we iterate through the training pairs and project the images onto the subspace defined by the extracted eigen vectors
- We concatenate the arrays of the projection of images and append it finally to an array named train features
- Then fit the model into a Multi Layer Perceptron Classifier to perform binary classification. '1' if the images are of the same person and '0' if the images are not of the same person. (Different Classifiers were experimented with, more on page 8)

MACHINE LEARNING PIPELINE

01

Data preprocessing.

- Viola Jones Algorithm.
- Upscaling through ESRGAN.
- Resize and convert to grey scale.
- Make equal number of positive and negative pairs from train, val and test data.
- Assign appropriate labels.

02

Feature extraction.

- Apply 2DPCA and obtain the top eigen vectors with the help of training images.
- Project each image onto the subspace defined by the top eigen vectors obtained.
- Concatenate the arrays of image projections into a single array and append to the final training features array.

03

Model Training.

- Fit the training features with their corresponding labels into an MLP classifier
- MLP classifier with hidden layer sizes= (256,128,64), max_iterations as 1000, validation_fraction=0.1 and learning rate=0.001
- XGB Classifier objective='binary:logistic', n_estimators=8000

04

Model Evaluation and Hyperparameter Tuning

- Calculating validation accuracy and validation precision
- Calculating test accuracy and test precision
- Hyper parameter tuning of the two promising models- XGB classifier and MLP classifier to find the best possible accuracy and precision

05

Exploring promising areas.

- Benefits of upscaling could be further optimised if the size rescaling of images after upscaling the ROI is made to 256x256 px or 128x128 px instead of 64x64 px as we are failing to capture the benefits made through upscaling.
- More training data could improve the model training, subsequently its accuracy on unknown data
- We could try other face detection models like YOLO,MTCNN

FEATURE EXTRACTION:VISUALIZED

These are the thirty eigen vectors that were extracted from the training images. When an image is processed in test data or validation data. The images is projected on to the subspace created by these eigen vectors and the dot product of each image with these following eigen vectors is taken.

We then get a coefficient for each 'PC' Principle Component signifying its similarity or contribution to constitute the image given. This array of coefficients of image 1 is concatenated with the array of images of image 2. This combined feature coefficient array is passed on to the classifier to verify if both images belong to the same person or not.



Different values of k were tested-10,15,20,25,30,35.
The best accuracy was obtained at k=30

MODEL COMPARISONS AND BENCHMARK SCORES

The Model was fitted on various types of classifiers (to obtain Binary classification: 0 if the person in the second image doesn't match person in first and 1 if they match. The following classifiers appeared to look the most promising.

1. MLP Classifier
2. Random Forest Classifier
3. Gradient Boosting Machine Classifier
4. KNN Classifier

After Hyperparameter tuning parameters for each model proved to be effective and these are the precision and accuracy values respectively.

1. MLP CLASSIFIER

```
In [62]: from sklearn.neural_network import MLPClassifier
mlp_classifier = MLPClassifier(hidden_layer_sizes=(256,128, 64), activation='relu',
                               solver='adam', random_state=42, max_iter=1000, verbose=True,
                               learning_rate='adaptive', early_stopping=False, alpha=0.1,
                               learning_rate_init=0.001, n_iter_no_change=500, validation_fraction=0.1)
```

```
MLPClassifier(alpha=0.1, hidden_layer_sizes=(256, 128, 64),
               learning_rate='adaptive', max_iter=1000, n_iter_no_change=500,
               random_state=42, verbose=True)
```

```
In [50]: val_features = []
for img1, img2 in tqdm(val_pairs, desc='Validation pairs'):
    proj_img1 = project_2dpca(img1.flatten(), eig_vecs)
    proj_img2 = project_2dpca(img2.flatten(), eig_vecs)
    val_features.append(np.concatenate([proj_img1, proj_img2]))
val_features = np.array(val_features)
```

Validation pairs: 100%  5160/5160 [00:00<00:00, 9134.79it/s]

```
In [51]: val_pred = mlp_classifier.predict(val_features)
val_accuracy = accuracy_score(val_labels, val_pred)
val_precision = precision_score(val_labels, val_pred)

print(f"Validation Accuracy: {val_accuracy:.4f}")
print(f"Validation Precision: {val_precision:.9f}")
```

Validation Accuracy: 0.8703
Validation Precision: 0.912386707

```
In [52]: test_features = []
for img1, img2 in tqdm(test_pairs, desc='Test pairs'):
    proj_img1 = project_2dpca(img1.flatten(), eig_vecs)
    proj_img2 = project_2dpca(img2.flatten(), eig_vecs)
    test_features.append(np.concatenate([proj_img1, proj_img2]))
test_features = np.array(test_features)
```

Test pairs: 100%  8112/8112 [00:00<00:00, 10745.01it/s]

```
In [53]: test_pred = mlp_classifier.predict(test_features)
test_accuracy = accuracy_score(test_labels, test_pred)
test_precision = precision_score(test_labels, test_pred)

print(f"Test Accuracy: {test_accuracy}")
print(f"Test Precision: {test_precision:.9f}")
```

Test Accuracy: 0.9323224852071006
Test Precision: 0.913268914

2. Random Forest Classifier

```
In [55]: rf_classifier = RandomForestClassifier(n_estimators=100000, random_state=77)

rf_classifier.fit(train_features, train_labels)
```

```
In [56]: val_features = []
for img1, img2 in tqdm(val_pairs, desc='Validation pairs'):
    proj_img1 = project_2dpca(img1.flatten(), eig_vecs)
    proj_img2 = project_2dpca(img2.flatten(), eig_vecs)
    val_features.append(np.concatenate([proj_img1, proj_img2]))
val_features = np.array(val_features)


Validation pairs: 100%  5160/5160 [00:02<00:00, 2535.95it/s]
```

```
In [57]: val_pred = rf_classifier.predict(val_features)
val_accuracy = accuracy_score(val_labels, val_pred)
val_precision = precision_score(val_labels, val_pred)

print(f"Validation Accuracy: {val_accuracy:.4f}")
print(f"Validation Precision: {val_precision:.9f}")

Validation Accuracy: 0.8308
Validation Precision: 0.935681470
```

```
In [58]: test_features = []
for img1, img2 in tqdm(test_pairs, desc='Test pairs'):
    proj_img1 = project_2dpca(img1.flatten(), eig_vecs)
    proj_img2 = project_2dpca(img2.flatten(), eig_vecs)
    test_features.append(np.concatenate([proj_img1, proj_img2]))
test_features = np.array(test_features)

Test pairs: 100%  8112/8112 [00:02<00:00, 3460.08it/s]
```

```
In [59]: test_pred = rf_classifier.predict(test_features)
test_accuracy = accuracy_score(test_labels, test_pred)
test_precision = precision_score(test_labels, test_pred)

print(f"Test Accuracy: {test_accuracy}")
print(f"Validation Precision: {val_precision:.9f}")


Test Accuracy: 0.9309664694280079
Validation Precision: 0.935681470
```

3. XGB Classifier

```
In [79]: xgb_classifier = xgb.XGBClassifier(
    objective='binary:logistic', # For binary classification
    max_depth=3,                # Maximum depth of the tree
    learning_rate=0.1,           # Learning rate
    n_estimators=8000,           # Number of trees (boosting rounds)
    subsample=0.8,               # Subsample ratio of the training instance
    colsample_bytree=0.8,        # Subsample ratio of columns when constructing
    random_state=42,             # Random state for reproducibility
    eval_metric='error',
    grow_policy='lossguide'
)
```

```
XGBClassifier
XGBClassifier(base_score=None, booster=None, callbacks=None,
               colsample_bylevel=None, colsample_bynode=None,
               colsample_bytree=0.8, device=None, early_stopping_rounds=None,
               enable_categorical=False, eval_metric='error', feature_types=None,
               gamma=None, grow_policy='lossguide', importance_type=None,
               interaction_constraints=None, learning_rate=0.1, max_bin=None,
               max_cat_threshold=None, max_cat_to_onehot=None,
               max_delta_step=None, max_depth=3, max_leaves=None,
               min_child_weight=None, missing=nan, monotone_constraints=None,
               multi_strategy=None, n_estimators=8000, n_jobs=None,
```

```
In [65]: val_features = []
for img1, img2 in tqdm(val_pairs, desc='Validation pairs'):
    proj_img1 = project_2dpca(img1.flatten(), eig_vecs)
    proj_img2 = project_2dpca(img2.flatten(), eig_vecs)
    val_features.append(np.concatenate([proj_img1, proj_img2]))
val_features = np.array(val_features)


Validation pairs: 100%  5160/5160 [00:01<00:00, 4688.51it/s]
```

```
In [66]: val_pred = xgb_classifier.predict(val_features)
val_accuracy = accuracy_score(val_labels, val_pred)
val_precision = precision_score(val_labels, val_pred)

print(f"Validation Accuracy: {val_accuracy:.4f}")
print(f"Validation Precision: {val_precision:.9f}")

Validation Accuracy: 0.8864
Validation Precision: 0.920675105
```

```
In [67]: test_features = []
for img1, img2 in tqdm(test_pairs, desc='Test pairs'):
    proj_img1 = project_2dpca(img1.flatten(), eig_vecs)
    proj_img2 = project_2dpca(img2.flatten(), eig_vecs)
    test_features.append(np.concatenate([proj_img1, proj_img2]))
test_features = np.array(test_features)

Test pairs: 100%  8112/8112 [00:04<00:00, 3071.34it/s]
```

```
In [68]: test_pred = xgb_classifier.predict(test_features)
test_accuracy = accuracy_score(test_labels, test_pred)
test_precision = precision_score(test_labels, test_pred)

print(f"Test Accuracy: {test_accuracy}")
print(f"Test Precision: {test_precision:.9f}")

Test Accuracy: 0.9277613412228797
Test Precision: 0.906132959
```

4.KNN Classifier

(with k=20 i.e only using top 20 eigen vectors)

```
In [71]: from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=12)
knn.fit(train_features, train_labels)
```

```
Out[71]: KNeighborsClassifier
KNeighborsClassifier(n_neighbors=12)
```

```
In [72]: val_features = []
for img1, img2 in tqdm(val_pairs, desc='Validation pairs'):
    proj_img1 = project_2dpca(img1.flatten(), eig_vecs2)
    proj_img2 = project_2dpca(img2.flatten(), eig_vecs2)
    val_features.append(np.concatenate([proj_img1, proj_img2]))
val_features = np.array(val_features)
```

Validation pairs: 100%  5160/5160 [00:00<00:00, 13902.19it/s]

```
In [73]: val_pred = knn.predict(val_features)
val_accuracy = accuracy_score(val_labels, val_pred)
val_precision = precision_score(val_labels, val_pred)
print(f"Validation Accuracy: {val_accuracy:.4f}")
print(f"Validation Precision: {val_precision:.4f}")
```

Validation Accuracy: 0.8198
Validation Precision: 0.7708

```
In [75]: test_features = []
for img1, img2 in tqdm(test_pairs, desc='Test pairs'):
    proj_img1 = project_2dpca(img1.flatten(), eig_vecs2)
    proj_img2 = project_2dpca(img2.flatten(), eig_vecs2)
    test_features.append(np.concatenate([proj_img1, proj_img2]))
test_features = np.array(test_features)
```

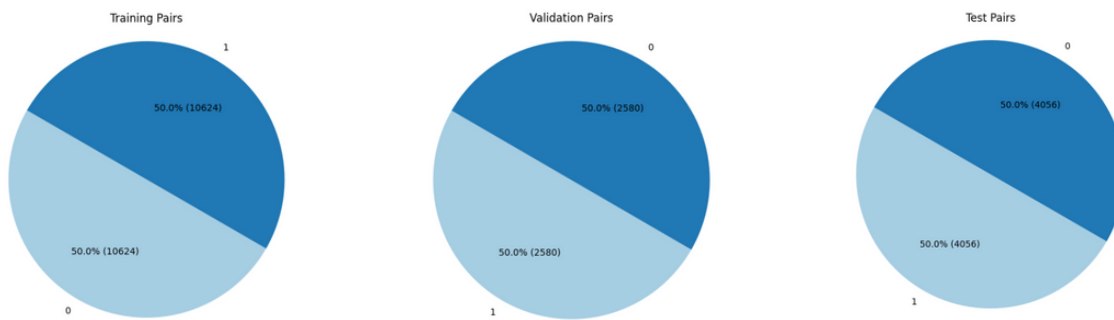
Test pairs: 100%  8112/8112 [00:00<00:00, 16801.24it/s]

```
In [76]: test_pred = knn.predict(test_features)
test_accuracy = accuracy_score(test_labels, test_pred)
test_precision = precision_score(test_labels, test_pred)
print(f"Test Accuracy: {test_accuracy}")
print(f"Validation Precision: {val_precision:.4f}")
```

Test Accuracy: 0.8292652859960552
Validation Precision: 0.7708

MODEL	MLP Classifier	Random Forest Classifier	XGB Classifier	KNN Classifier (k=20)
VALIDATION ACCURACY	87.03%	83.08%	88.64%	81.98%
VALIDATION PRECISION	91.23%	93.56%	92.06%	77.08%
TEST ACCURACY	93.23%	93.09%	92.77%	82.92%
TEST PRECISION	91.32%	93.56%	90.61%	77.08%

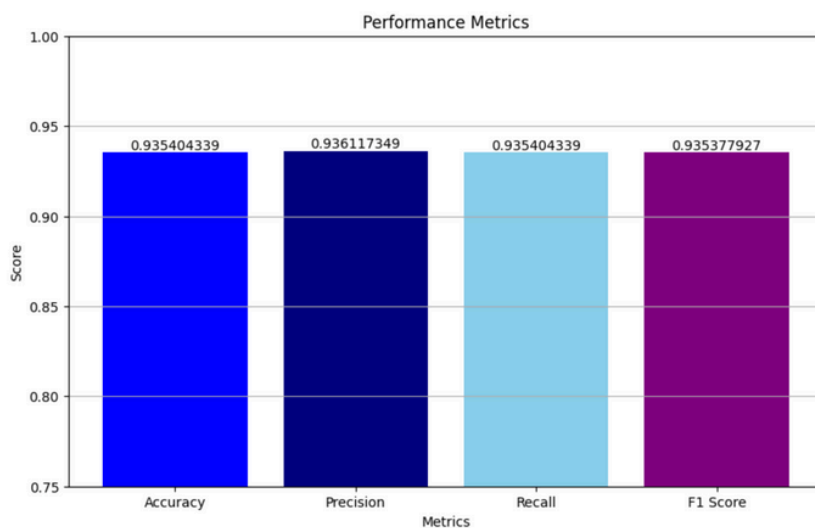
METRICS AND ANALYTICS



Equal Pairs of negatives and positives were made of all three types of data: train, test and validation. The process of generating pairs was as follows:

1. Iterate through all person folders.
2. Generate positive pairs from each folder (+ve pairs made)
3. Iterate through all person folders
4. Chose a random folder from other folders
5. Choose a random image and generate -ve pairs.
6. Ensuring equal number positive and negative pairs by using random.sample

Among the classifiers providing the highest accuracy, the metrics for the MLP classifier are as follows:

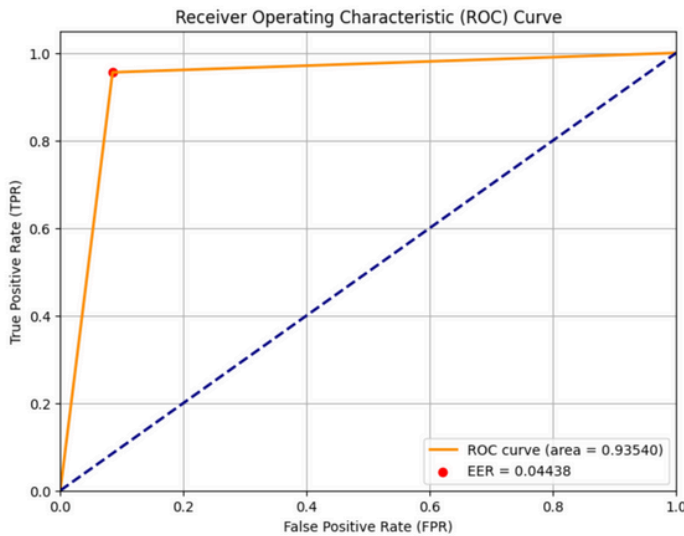


```
from sklearn.metrics import classification_report
# Print classification report for detailed metrics
print(classification_report(test_labels, test_pred))
```

	precision	recall	f1-score	support
0	0.95	0.92	0.93	4056
1	0.92	0.96	0.94	4056
accuracy			0.94	8112
macro avg	0.94	0.94	0.94	8112
weighted avg	0.94	0.94	0.94	8112

For class 0, it achieved a precision of 0.95, a recall of 0.92, and an F1-score of 0.93, based on 4056 positive test samples. For class 1, it attained a precision of 0.92, a recall of 0.96, and an F1-score of 0.94, also based on 4056 negative test samples. The overall accuracy of the MLP classifier was 0.94 across 8112 samples.

METRICS AND ANALYTICS

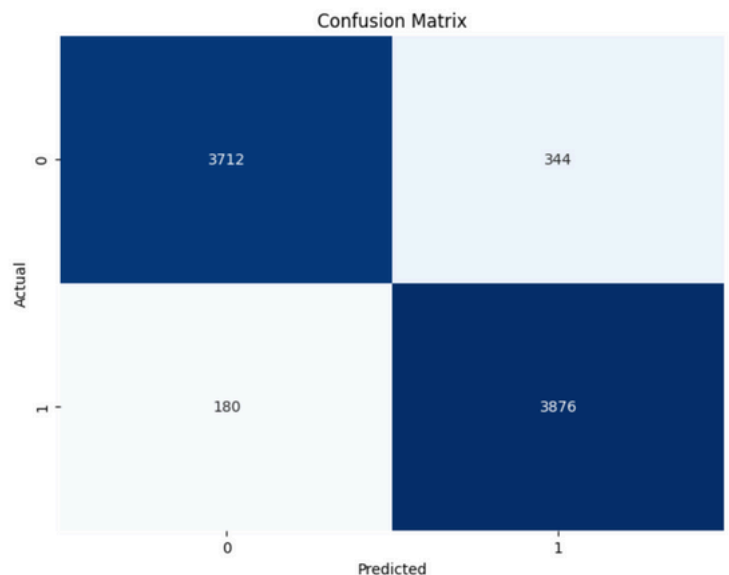


AUC-ROC of 0.93540 indicates that the model is performing very well in distinguishing between the two classes it is predicting.

The EER of 4.438% signifies that the system achieves an equal balance between incorrectly accepting legitimate users and rejecting authorized users.

The plotted confusion matrix indicates that out of 8112 test pairs:

- **True Positives** were 3876 in number
- **True Negatives** were 3712 in number.
- **False Positives** were 344 in number.
- **False negatives** were 180 in number



Computational Efficiency

Average Inference Time: 0.0385644 seconds
Peak Memory Usage: 23.77 MB
Average CPU Utilization: 19.79%

To evaluate the performance of the MLP classifier, a thorough assessment of its inference time, memory usage, and CPU utilization was conducted. First, the inference time was measured by running predictions on the test features 100 times, recording the duration for each run, and calculating the average. The results showed an average inference time of approximately 0.038 seconds.

Next, memory usage was assessed using the tracemalloc module. Memory allocation was traced before running predictions, and both the current and peak memory usage were recorded after the predictions.

The peak memory usage was found to be approximately 23.77 MB. Lastly, CPU utilization was measured by sampling the CPU usage during predictions over ten intervals, each lasting 0.1 seconds. The average CPU utilization was determined to be approximately 19.79%.

CONCLUSIONS AND PROMISING AREAS FOR FURTHER DEVELOPMENT

- Images could be resized to 128X128 px or 256x256 px and the model can be evaluated considering the loss of data when we resize the images to a standard of 64x64px for 2DPCA
- There is a increase in accuracy by 1% when you use ESRGAN upscaling. The benefits of this could be further improved if the images are of 128x128 size as we could lose less of the data provided by the ESRGAN upscaling when you resize it.
- Due to the lightweight nature of this model this shows potential of side deployment on the candidate's device rather than processing loads on the server
- With even more and diverse images to train on, the principle components extracted can accommodate a variation of faces thus improving the model accuracy.
- The principle components selected need to be looked into so they represent most of the population, making it suitable for all races
- Different models can be trained for specific geographic regions where there is a distinct variation in face features to increase the accuracy of recognising faces through principle components.
- The candidates can be made to sit in front of a simple, solid colour background, so the patterns wouldn't hinder the working of the face detection algorithm and detect a region other than the face.

REFERENCES

- C. Song, Z. He, Y. Yu and Z. Zhang, "Low Resolution Face Recognition System Based on ESRGAN," 2021 3rd International Conference on Applied Machine Learning (ICAML), Changsha, China, 2021, pp. 76-79, doi: 10.1109/ICAML54311.2021.00024. keywords: {Image recognition;Art;Databases;Face recognition;Superresolution;Machine learning;Generative adversarial networks;Face detection;Face Recognition;YOLO;Super-resolution reconstruction;ESRGAN},
- Yang, Jian & Zhang, David & Frangi, Alejandro & Yang, Jing-yu. (2004). Two-Dimensional PCA: A New Approach to Appearance-Based Face Representation and Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence. 26. 10.1109/TPAMI.2004.1261097.
- Koch, G.R. (2015). Siamese Neural Networks for One-Shot Image Recognition.
- V. D. M. Nhat and S. Lee, "Kernel-based 2DPCA for Face Recognition," 2007 IEEE International Symposium on Signal Processing and Information Technology, Giza, Egypt, 2007, pp. 35-39, doi: 10.1109/ISSPIT.2007.4458104. keywords: {Face recognition;Principal component analysis;Covariance matrix;Kernel;Independent component analysis;Feature extraction;Face detection;Lighting;Signal processing;Information technology;PCA;Kernel PCA;2DPCA;Face Recognition},
- Turk, M.A. and Pentland, A.P., 1991, January. Face recognition using eigenfaces. In Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition (pp. 586-587). IEEE Computer Society.
- M. Ghizlane, B. Hicham and F. H. Reda, "A New Model of Automatic and Continuous Online Exam Monitoring," 2019 International Conference on Systems of Collaboration Big Data, Internet of Things & Security (SysCoBioTS), Casablanca, Morocco, 2019, pp. 1-5, doi: 10.1109/SysCoBioTS48768.2019.9028027. keywords: {Authentication;Monitoring;Face recognition;Servers;Smart cards;Face;Online exam;continuous authentication;machine learning;automatic monitoring},
- Slusky, Ludwig (2020) "Cybersecurity of Online Proctoring Systems," Journal of International Technology and Information Management: Vol. 29: Iss. 1, Article 3.DOI: [Goal 2](#)
- N. H. Barnouti, M. H. N. Al-Mayyahi and S. S. M. Al-Dabbagh, "Real-Time Face Tracking and Recognition System Using Kanade-Lucas-Tomasi and Two-Dimensional Principal Component Analysis," 2018 International Conference on Advanced Science and Engineering (ICOASE), Duhok, Iraq, 2018, pp. 24-29, doi: 10.1109/ICOASE.2018.8548818. keywords: {Face;Face recognition;Feature extraction;Webcams;Face detection;Classification algorithms;Training;Face Detection;Face Tracking;Face Recognition;Viola-Jones;KLT;2DPCA.},
- R. S. V. Raj, S. A. Narayanan and K. Bijlani, "Heuristic-Based Automatic Online Proctoring System," 2015 IEEE 15th International Conference on Advanced Learning Technologies, Hualien, Taiwan, 2015, pp. 458-459, doi: 10.1109/ICALT.2015.127. keywords: {Conferences;E-learning;Proctoring system;Face Detection;Inference system;On-line examination},
- Aurelia, S., Thanuja, R., Chowdhury, S. et al. AI-based online proctoring: a review of the state-of-the-art techniques and open challenges. Multimed Tools Appl 83, 31805–31827 (2024). <https://doi.org/10.1007/s11042-023-16714-x>
- C. Ledig et al., "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 105-114, doi: 10.1109/CVPR.2017.19. keywords: {Image resolution;Signal resolution;Gallium nitride;Image reconstruction;Manifolds;Training;Network architecture},
- Wang, X. et al. (2019). ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. In: Leal-Taixé, L., Roth, S. (eds) Computer Vision – ECCV 2018 Workshops. ECCV 2018. Lecture Notes in Computer Science(), vol 11133. Springer, Cham. https://doi.org/10.1007/978-3-030-11021-5_5

GLOSSARY

Proctoring: The process of supervising or monitoring interviewees during their assessments to ensure the integrity and fairness of the evaluation.

Face Verification System: A biometric system that compares two images to determine if they are of the same person.

ESRGAN (Enhanced Super-Resolution Generative Adversarial Network): An advanced neural network used to enhance the resolution and quality of images, making low-resolution images clearer and more detailed.

2DPCA (Two-Dimensional Principal Component Analysis): A statistical technique for feature extraction from images, reducing dimensionality while preserving essential information.

AUC-ROC (Area Under the Receiver Operating Characteristic Curve): A performance metric for classification models, indicating the model's ability to distinguish between classes. Higher values indicate better performance.

EER (Equal Error Rate): The point at which the false acceptance rate (FAR) equals the false rejection rate (FRR), used as a measure of the system's accuracy in biometric authentication.

Precision: The ratio of true positive predictions to the total number of positive predictions, indicating the accuracy of the positive predictions.

Recall: The ratio of true positive predictions to the total number of actual positive instances, indicating the ability of the model to identify positive instances.

F1-Score: The harmonic mean of precision and recall, providing a single metric to evaluate the model's performance.

MLP (Multilayer Perceptron) Classifier: A type of neural network used for classification tasks, consisting of multiple layers of interconnected nodes.

Random Forest Classifier: An ensemble learning method that constructs multiple decision trees for classification and outputs the majority vote.

Gradient Boosting Machine Classifier: An ensemble learning method that builds models sequentially, each correcting errors made by the previous ones.

KNN (K-Nearest Neighbours) Classifier: A simple, non-parametric classification algorithm that assigns a class to a sample based on the majority class of its nearest neighbors.

Positive Pair: In face verification, a pair of images that belong to the same person.

Negative Pair: In face verification, a pair of images that belong to different people.

Hyperparameter Tuning: The process of optimizing the parameters that govern the learning process of a machine learning model to improve its performance.

Principal Components: The directions in which the data varies the most, used in dimensionality reduction techniques like PCA and 2DPCA.

Edge Deployment: Running AI models on local devices (e.g., smartphones, laptops) rather than on centralized servers, reducing latency and server load.

Solid-Colored Background: A plain, single-color backdrop used in images to minimize distractions and improve the accuracy of face detection algorithms.

The true and false positives and negatives cannot be shown in the report bound to the non-disclosure agreement signed that prevents me from sharing the datasets to a third party