A Major Project Report

On

Effective Face Recognition Using Graph Auto-Encoder

Submitted in partial fulfillment of
the requirements for the award of the degree of

Master of Computer Applications

Submitted by

Sanjog Kumar Pradhan
Roll No:19MCA016

Sonika Pradhan
Roll No:19MCA019

Under the guidance of
Prof. Swarup Roy

Department of Computer Applications
Sikkim University
Gangtok-737102, India

August, 2022

Acknowledgements

It is our privilege and honor to work under the supervision of Prof. Swarup Roy, Department of Computer Applications, Sikkim University. We would like to express our sincere gratitude to our supervisor for his guidance, care, patience, and for providing us with an excellent atmosphere for doing research.

We are grateful to Dr. Mohan Pratap Pradhan, Associate Professor and Head, Department of Computer Applications, Sikkim University, for his support and encouragement during the course of our research.

We would also like to thank all faculty members at the Department of Computer Applications, Sikkim University, for their valuable suggestions during our research work. And special thanks to Mr. Binon Teji Research Scholar at Department of Computer Applications, Sikkim University, for helping and guiding us through out this project.

Sanjog Kumar Pradhan
Roll No:19MCA016

Sonika Pradhan
Roll No:19MCA019

Declaration

We **Sanjog Kumar Pradhan[19MCA016], Sonika Pradhan[19MCA019]**, are registered candidates for Master of Computer Applications under Department of Computer Applications (MCA) of Sikkim University, Gangtok, India, declare that this is our own original work and does not contain material for which the copyright belongs to a third party and that it has not been presented and will not be presented to any other University/ Institute for a similar or any other Degree award.

We further confirm that for all third-party copyright material in our project report (including any electronic attachments), we have "blanked out" third party material from the copies of the dissertation fully referenced the deleted materials and where possible, provided links (url) to electronic sources of the material.

We hereby transfer exclusive copyright for this project report to Sikkim University. The following rights are reserved by the author:

a) The right to use, free of charge, all or part of this article in future work of their own, such as books and lectures, giving reference to the original place of publication and copyright holding.

b) The right to reproduce the article or dissertation for their own purpose provided the copies are not offered for sale.

Sanjog Kumar Pradhan
Roll No:19MCA016

Sonika Pradhan
Roll No:19MCA019

Date:

Certificate from the Supervisor

This is to certify that **Sanjog Kumar Pradhan [19MCA016]** , **Sonika Pradhan [19MCA019]** , are registered candidates for Master of Computer Applications Programme under the department of **Computer Applications** of Sikkim University, Gangtok, India.

The undersigned certifies that they have completed all other requirements for submission of the dissertation and hereby recommends for the acceptance of their dissertation entitled **'Effective Face Recognition Using Graph Auto-Encoder'** in the partial fulfillment of the requirements for the award of MCA Degree by Sikkim University, Gangtok, India.

Prof. Swarup Roy
Supervisor
Department of Computer Applications
Sikkim University
Gangtok-737102, India

Date:

Certificate from Head of Department

This is to certify that **Sanjog Kumar Pradhan [19MCA016]** , **Sonika Pradhan [19MCA019]** , are registered candidates for Master of Computer Applications Programme under the department of **Computer Applications** of Sikkim University, Gangtok, India.

The undersigned certifies that they have completed all other requirements for submission of the dissertation and hereby recommends for the acceptance of their dissertation entitled **'Effective Face Recognition Using Graph Auto-Encoder'** in the partial fulfillment of the requirements for the award of MCA Degree by Sikkim University, Gangtok, India.

Dr. Mohan Pratap Pradhan
Associate Professor/Head of Department
Department of Computer Applications
Sikkim University
Gangtok-737102, India

Date:

Abstract

Face recognition is an important problem in computer vision domain that seeks to authenticate human face from an image or a video frame. Majority of the algorithms extract key facial features using image pixels in the prediction pipeline. In this work, we retrieve a graph from the facial landmarks of an input image that are connected to each other with a corresponding weight value using a predefined mediapipe Google's algorithm. Further, we leverage the power of network embedding that represents the entire graph in the latent space for the recognition task. We demonstrate our work on Yale Face Data using Graph Auto-Encoder (GAE), which is a Graph Neural Network (GNN) based embedding method and we also compare the results with random-walk based node2vec algorithm. Experimental results reveal the efficiency of our proposed approach towards the face recognition task using deep graph embeddings.

Contents

Chapter 1

Introduction

The automatic identification of an individual based on his or her physiological and/or behavioural features is referred to as biometric recognition, or simply biometrics. Biometrics can be used to verify or establish a person's identity based on "who" they are rather than "what" they have (such as an ID card) or "what" they can recall (such as a password) [3].
Due to its potential use in a wide range of domains, including identity identification, information security, surveillance, human-computer interface, and other areas, face recognition has drawn a lot of attention. Face image alterations due to viewpoint, lighting, and facial expressions constitute a significant difficulty.

1.1 Face Recognition

Face recognition is a way of identifying or confirming an individual's identity using their face. It can be used to identify a person in photos, videos, or real time. Unique identification of a person has become an important area of study as it finds a wide range of commercial and law enforcement applications.
The various use of face recognition system are as follows:
i. Finding missing people and identify perpetrators: Face recognition is being actively used in law enforcement agencies to identify criminals and find missing people by comparing faces on live camera feeds with those on the watch list.
They also use face recognition to find missing children by combining facial recognition with ageing software to predict how children will look after several years and find them even if they have been missing for several years.
ii. Automated Time Tracking System: It can automatically record the entry and exit time of the student for the attendance, it can also track time of the employee.
iii. Cost-Effective: A facial recognition attendance system can save business

resources by automatic employee time tracking.
iv. Touch-less Sign In System: Covid 19 can be better managed by minimizing physical contact in public places and work environments. Post pandemic there has been a significant increase in demand and adoption of contact-less technologies.
v. More Accurate and Better Worker Attendance: With a face recognition attendance system, the entire environment is automated. Face recognition system won't just take the attendance but also automatically record the entry-exit time of the employees. It also adds to the security of the workplace as the system can recognize who left the designated area and when accurately.
In general, face recognition is carried out in the following basic steps:

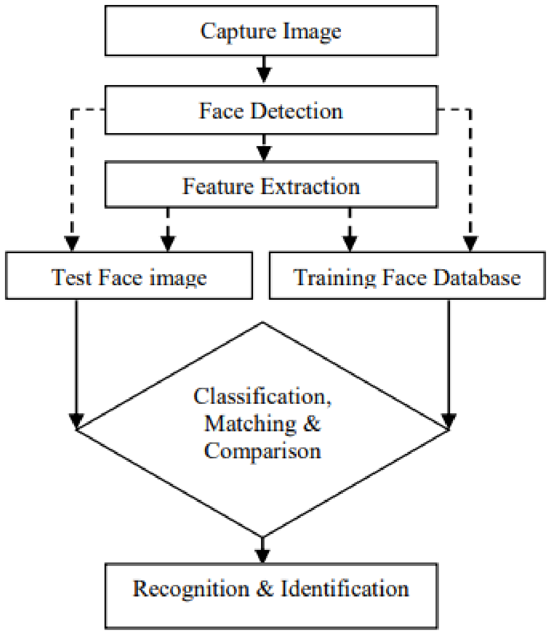

Figure 1.1: Architecture of face recognition

First, the image is captured, and to ascertain whether the face is actually present in the image or not, face detection is carried out on the captured image. If a face is detected, depending upon different methods, a feature of the face is extracted. Then the input image is compared with the picture in the database. Classification is done to identify picture to which the new observed picture is similar to.

1.2 Various face recognition methods

The various face recognition methods are [8]:
i. Holistic method: In the holistic method the entire face is given as an input to the face catching system. Eigenfaces, PCA are some well know example of holistic methods. For example in Eigenface approach, before creating the eigenface the given image is normalised and resized so that all the images eyes, mouths are lined up, then the eigenface is extracted from the image. This is then represented as a vector which is then used for comparison and recognition.
In this paper [7] a face recognition system using the Principal Component Analysis (PCA) algorithm was implemented. The algorithm is based on an eigenfaces approach which represents a PCA method in which a small set of significant features are used to describe the variation between face images.
ii. Feature Based: In this approach the local features of the face like eyes, nose, mouth are first extracted and their locations, geometry and appearance are given as a input to a structural classifier. This features can be extracted using different methods like feature-template-base methods or structural matching methods or generic methods based on edges lines and curves.
This paper [5] introduces a novel Gabor-Fisher (1936) classifier (GFC) for face recognition. The GFC method, which is robust to changes in illumination and facial expression, applies the enhanced Fisher linear discriminant model (EFM) to an augmented Gabor feature vector derived from the Gabor wavelet representation of face images.
iii. Hybrid Methods: This is a combination of both holistic methods and feature based methods generally 3D images are used to get the curves of the eye sockets, or the shapes of the chin or forehead.This is used to make a template which is then converted into a numerical representation of the face. This numerical representation is used to compare with the existing image.
This paper [2] considers hybrid classification architectures and shows their feasibility on large data bases consisting of facial images. Its architecture, consists of an ensemble of connectionist networks - radial basis functions (RBF) - and decision trees (DT).
iv. Neural Network Based Method:
Neural networks are used to recognize the face through learning correct classification of the coefficients calculated by the eigenface algorithm. The network is first trained on the pictures from the face database, and then it is used to identify the face pictures given to it.
In this paper [6], they have propose a novel framework for dynamic 3D face identification/recognition based on facial keypoints. Each dynamic sequence of facial expressions is represented as a spatio-temporal graph, which is constructed using 3D facial landmarks. Each graph node contains local shape and texture features

that are extracted from its neighborhood. For the classification/identification of faces, a Spatio-temporal Graph Convolutional Network (ST-GCN) is used.

Majority of face recognition system uses pixel values directly or indirectly while processing the image, which creates the need to handle pose, illumination, and expression variations problem.This results in the overhead of running various pre-processing methods on the image before working on it for face recognition. Graph using facial key points to capture the anthropomorphic measurements of a face can be the other alternative. A model was needed to learn the information from the graph in order to preserve it in the lower dimensional space, GAE a Graph representation model is one such model.

Objective of this project is to detect large number of facial feature points which will be further used to create graph to incorporate as much information as possible about the face. The GAE will be modified to better suit our requirement for utilizing this edge weight. The vector representation will then be utilized for the comparison and recognition of the face.

The structure of this report is organized as follows:

Section 1.3 will introduce representation learning and graph representation learning. Chapter 2 presents our methodology. The experiments, results, and comparisons are given in chapter 3. Finally, chapter 4 concludes the report and discusses possible future directions.

1.3 Representation Learning

Representation learning is an important aspect of machine learning. It automatically detects the feature pattern in data. When a machine is provided with data, it automatically learns the representation itself without any human intervention. The goal of representation learning is to train machine learning algorithms to learn useful representations, such as those that are interpretable, incorporate latent features, or can be used for transfer learning.

1.3.1 Graph Representation Learning

The idea behind the graph representation learning approach is to learn a mapping that embeds nodes, or entire graphs or sub graphs as points in a low-dimensional vector space and to optimize this mapping so that geometric relationships in this learned space reflect the structure of the original graph. After optimizing the embedding space, the learned embedding can be used for the down-stream tasks.

Graph Auto-Encoder

Graph auto-encoder [4] is a powerful graph embedding method which can map graph data into a low-dimensional space. It is a applied in graph analytic to reduce the computational cost. It consists of encoder which converts the input into its low dimensional space whereas decoder tries to reconstruct the original input adjacency matrix. Detail explanation will be in section 2.2.

Node2Vec

Node2Vec [1] is another unsupervised graph representation learning algorithm that transforms graphs (or networks) into their numerical representations. In our previous minor project, we have already used Node2Vec for the purpose of face recognition. Node2vec does the 1st order random walk and 2nd order random walk on the given input graph. The 1st order random walk depends only on the current state and is done by sampling nodes on the graph along the edges of the graph. The 2nd order random walk depends not only on the current state but also the previous state. Node2vec creates numerical representations of nodes in the graph via a 2nd order (biased) random walk. Each node in the network is used as a starting point for a series of random walks, which are then passed to word2vec to generate the final node embeddings.
The node2Vec model does a biased random walk on the graph to convert the given graph to it's latent representation. These random walk is done in two phase:
i. First order random walk: A random walk on a graph can be thought of as a "walker" traversing the graph along the edges of the graph. At each step, the walker needs to decide where to travel next and then move to the next state. This process is referred to as a1-hop transition.
Thefirst orderrandom walk performs 1-hop transition that depends only on the current state. Node2Vecperformssecond orderrandom walk, which is a slightly modified version that incorporates information from the previous step.
ii. Second order random walk: Instead of looking at direct neighbors of the current state, the second order transition applies a bias factor alpha to reweight the edge weights depending on the previous state.
The random walk generation process is repeated multiple times using each node in the graph as the initial node. Then, as a result, we have a large series of node sequences. This series can then be directly fed into the word2Vec algorithm to generate node embeddings.

Chapter 2

Methodology

To achieve our objective, we will first detect the face and estimate the landmark on it. Using those landmarks, we will construct the face graph. This graph is passed to the model to get its embedding, which will be used for the comparison and recognition.

2.1 Input graph Construction

In this project, we have used Google's Mediapipe package to detect the face in the image as well as to estimate the facial landmarks in that face. MediaPipe handles orientation, lighting, and pose variation in the image well, while detecting the face in the image. First, the face is detected in the image and, to construct the face graph, 468 facial key points are estimated in the detected face. Then these facial feature key points are connected using the list of connections from the mediapipe, which forms a mesh-like graph. For the experimental purpose, both binary and weighted graphs were constructed. This graph can be conveniently represented through an adjacency matrix . To represent this graph by an adjacency matrix, we order the nodes in the graph so that every node indexes a particular row and column in the adjacency matrix. We can then represent the presence of edges as entries in this matrix by the euclidean distance between them or by 1 in the case of a binary graph and 0 otherwise.

Figure 2.1: An illustrated example of an input graph for particular images of a Yale dataset using Mediapipe

2.2 Model

The above constructed graph is given to the GAE model for further processing of the input graph. The GAE takes an adjacency matrix as well as a feature matrix as input to learn the features of the graph. In our case, we are giving an identity matrix as a feature matrix, as we want our modal to learn these features from the input graph.
Graph auto-encoders (GAE) comprises of 1) a graph encoder model, which take as input a feature matrix and a graph adjacency matrix, and produces an node embedding matrix and 2) a decoder model, which takes the node embeddings and predicts respective entries in the adjacency matrix.
Architecture
It's architecture is as follows:-

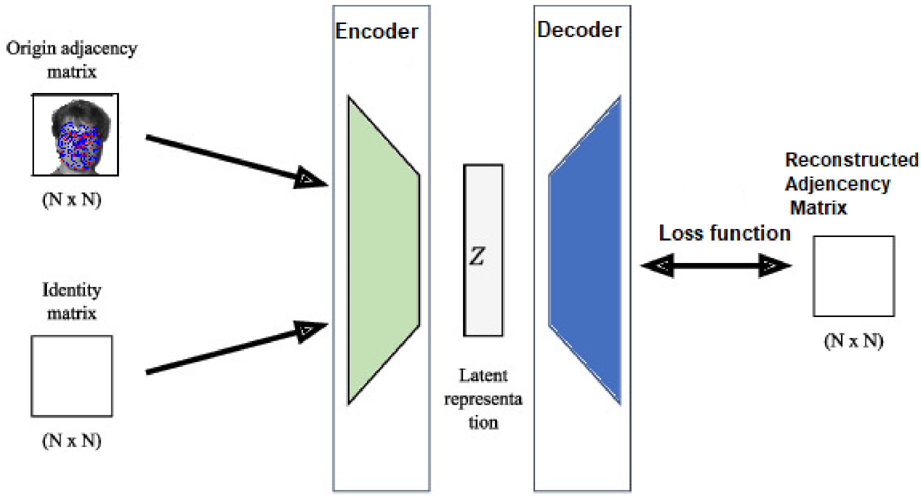

Figure 2.2: Architecture of Graph Auto-encoder

Encoder and Decoder
The encoder consists of two layers which produces a low dimensional embedding representation.It takes an adjacency matrix and a feature matrix as inputs and generates the latent variable as output. The embedding of the graph is the output of the encoder and is calculated using the formulae.

$$\mathbf{Z} = \mathbf{f}(\mathbf{A}, \mathbf{X}) = \sigma(\tilde{\mathbf{A}}\ \mathbf{ReLU}(\tilde{\mathbf{A}}\mathbf{X}\mathbf{W_0})\ \mathbf{W_1}), \tag{2.1}$$

$$with\ \tilde{\mathbf{A}} = \mathbf{D^{-1/2}}(\mathbf{A} + \mathbf{I})\mathbf{D^{-1/2}}$$

In the encoding phase, the GCN integrates both adjacency matrix (A) and the feature matrix (X) in the input module. The equation adds W_0 as the learning parameter and $ReLU$ is the activation function.The output of this layer along with $\tilde{A}$ and the updated weight W_1 gives the final embedding (Z) $\tilde{A}$ represents the normalized form of the input adjacency matrix. The decoding phase reconstructs the input adjacency matrix ($\hat{A}$) by doing the inner product of the embeddings (Z), which are the latent representations.

$$\hat{\mathbf{A}} = \sigma(\mathbf{Z^T}\mathbf{Z})$$

For the optimisation of weights we use Adam Optimiser and mean square error is used as the loss function. The loss function of an autoencoder measures the information lost during the reconstruction. This loss is used to optimize the embedding so that the loss in the next iteration decreases.

$$\mathbf{L}(\mathbf{X}, \hat{\mathbf{X}}) = ||\mathbf{X} - \hat{\mathbf{X}}||^{\mathbf{2}}, \tag{2.2}$$

2.3 Recognition and comparison

To check whether the test image and signature belong to the same subject, we have taken three distance measures. Pearson correlation, Euclidean distance, and Cosine similarity. We will measure the distance of the latent representation of test image with all the signatures and check with which signature the test image gives the best distance. If the images with the best distance belong to the same subject, we will say the system has successfully recognised else the system has failed to do so.

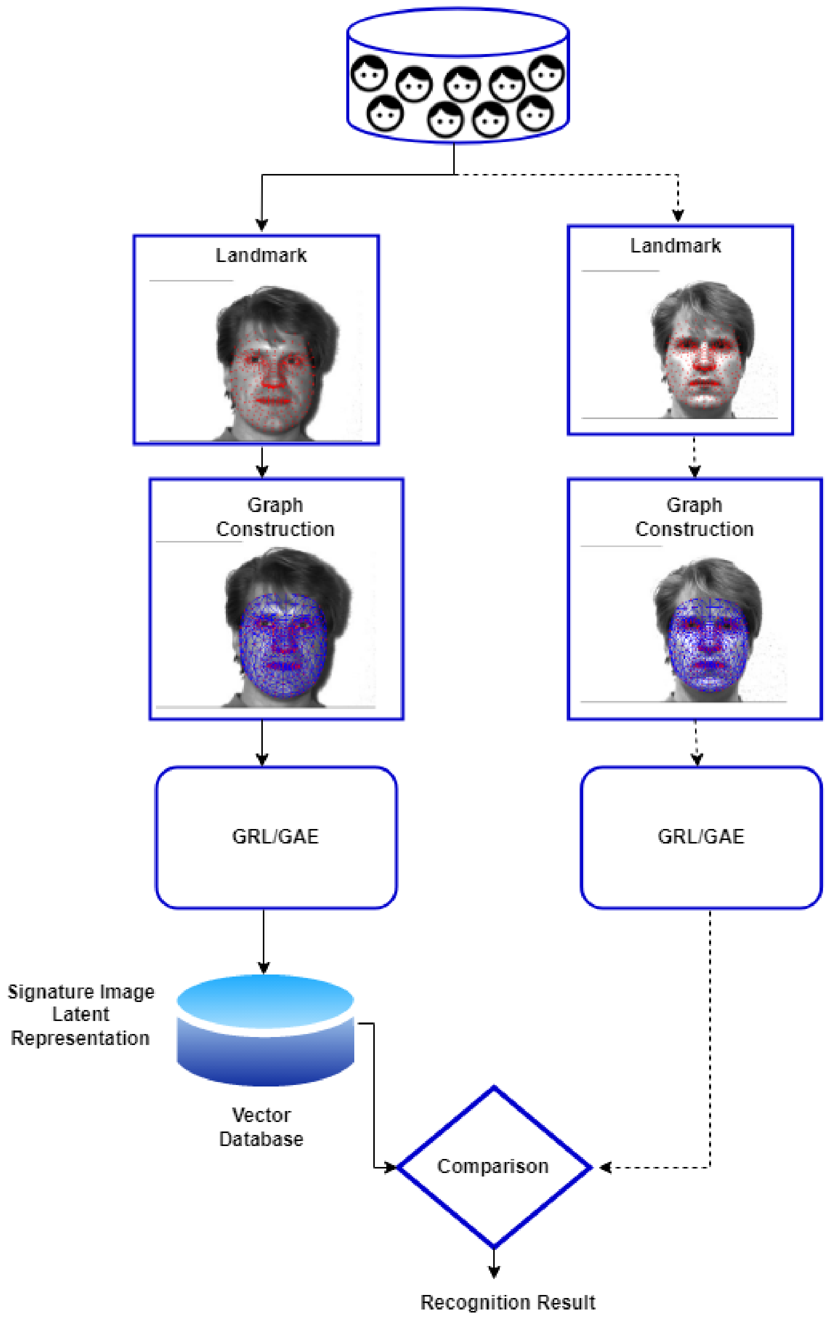

Figure 2.3: Workflow of our methodology

Chapter 3

Experiments and Results

Our system is tested on the yale dataset, which consists of grayscale images of 15 subjects. In this chapter, we will discuss experiments and results.

Dataset

The Yale Face data set contains 165 gray-scale images of 15 different subjects, that is, 11 different images of each subject. Each of those 11 images varies in expression and illumination configuration. For the expression variation, we have happy, sad, surprised, sleepy, and winking face images. For the illumination variation, we have three different conditions where light is either focused on the face, or to the left or right side of the subject. The data set also includes a few images of every subject with glasses on. This data set has a good number of variations in terms of expression and illumination.

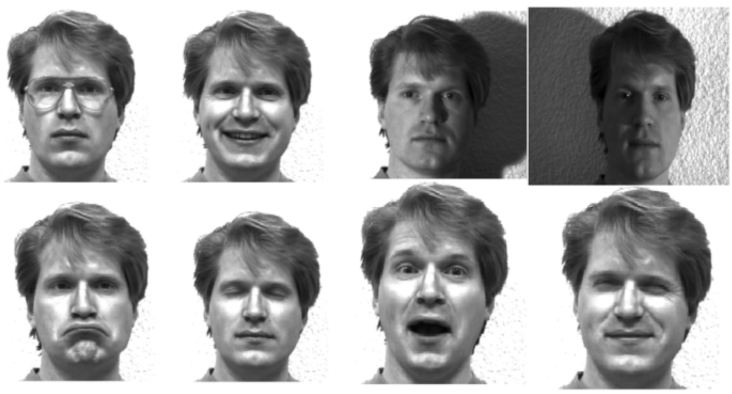

Figure 3.1: An illustrative example of facial images in Yale data set of a subject with its expression and illumination variations

For effective evaluation of the system, we have divided the Yale data set into two groups according to the following two types of images in the data set:
i. Neutral face image: The images containing faces with out expression.

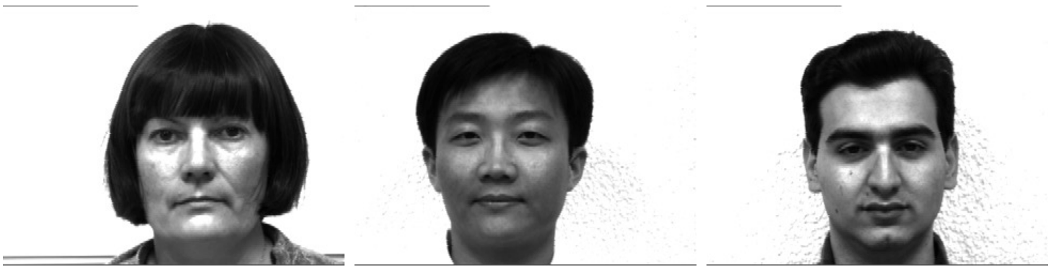

Figure 3.2: An illustrative example of neutral face images of a few subjects from the Yale data set

ii. Non-neutral face image: The images containing faces with different expression such as smiling,winking,sleepy etc or face with facial occlusion like glasses.

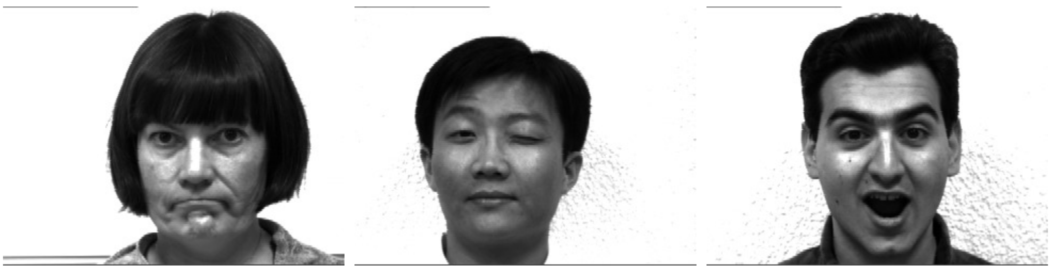

Figure 3.3: An illustrative example of non-neutral face images of a few subjects from the Yale data set

Implementation Environment

Before going to the experiment itself, all of the following tests are done on Google Colaboratory (colab), developed by Google Research. It is a free Jupyter notebook environment that runs entirely on the cloud. The colab can run both Python 2 and Python 3 run times and comes with essentially all of the most popular machine learning libraries like tensorflow, matplotlib,and keras pre-installed and ready to use. Users just have to import them wherever required. These notebooks can be shared and multiple users can work on the same notebook. Even though the space and processor allocation on the cloud is temporary and the notebook is deactivated after a period of time and all of the user's data and configuration is deleted, the notebook itself can be saved and stored in the Google drive. As for the data set and models, we can store them on the Google drive, which can be used seamlessly as an external drive in the colab.

MediaPipe is a framework for building machine learning pipelines for processing time-series data like video, audio, etc. This cross-platform framework works in Desktop/Server, Android, iOS, and embedded devices like the Raspberry Pi and Jetson Nano. In this project, we are using mediapipe to detect the face in the image as well as to estimate the facial landmarks in that face. MediaPipe handles orientation, lighting, and pose variation in the image well, while detecting the face in the image. MediaPipe is able to estimate 468 different facial feature points on the face. It also provides a connection between these points, using which a mesh-like graph is created.

Figure 3.4: Landmark detected using Mediapipe

Evaluation metrics

To summarise the performance of our system we have used multi-class confusion matrix. It is a table with four different combinations of predicted and actual values using which we will calculate various classification metrics.

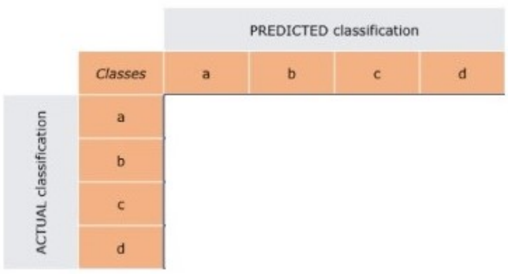

Figure 3.5: Multi-class confusion metrics

Following are the metrics which we will be using:

i. Accuracy: From all the classes (positive and negative), how many of them we have predicted correctly.It is calculated by a following formula.

Accuracy=(TP+TN)/(TP+TN+FP+FN)

ii. Precision: From all the classes we have predicted as positive, how many are actually positive.It is calculated by a following formula:

Precision=TP/(TP+FP)

iii. Specificity/True negative rate: Specificity is calculated as the number of correct negative predictions divided by the total number of negatives. It is also called true negative rate (TNR). The best specificity is 1.0, whereas the worst is 0.0. It is calculated by following formula:

Specificity=TN/(TN+FP)

iv. Sensitivity: Sensitivity identifies the rate at which observations from the positive class are correctly predicted.It is calculated by following formula:

Sensitivity=TP/TP+FP

v. False positive rate: False positive rate identifies the rate at which observations is predicted positive and is actually negative. It is calculated by following formula:

False positive rate(FPR)=FP/(FP+TN)

vi. False negative rate: False negative rate identifies the rate at which observations predicted value is negative, but the actual value is positive, It is calculated by following formula:

False negative rate(FNR)=FN/(TP+FN)

vii. Recognition rate: Recognition rate gives the overall how many faces was correctly recognised, It is calculated by following formula:

Recognition Rate= True Positive/Total Test Image

Here,

True Positive(TP): It was predicated positive and its true.

True negative(TN): It was predicated negative and its true.

False Positive(FP): It was predicated positive and its false.

False Negative(FN): It was predicated negative and its false.

While working on a unique signature for a subject, we converted the face graph into its latent representation. While doing so, the original GAE was not able to produce consistent latent representation even for the same image. Therefore, we fixed following variables, which were otherwise initialised randomly dropout_mask inside the dropout sparse function, which is used to avoid over-fitting of the curve while training. The weight variable was also initialised with a seeded random value.

3.1 Evaluating Recognition rate without Graph Preprocessing

In the first layer while doing the convolution the adjacency matrix was normalised before using it for convolution, We checked whether this normalisation was resulting in information loss or not. We noticed that giving graph without normalisation had a negative impact on our result, with only 40% recognition rate for 150 test images, which is well below the recognition rate with normalisation.We will say the system has successfully recognised when the distance between the latent representation of the test face and the signature belongs to the same subject. The different distance measures that we have calculated are Euclidean distance, Pearson correlation, and cosine similarity.

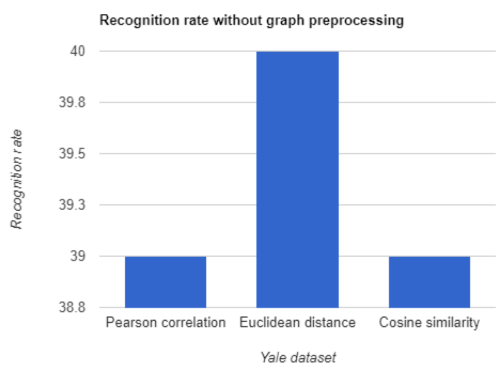

Figure 3.6: Bar-Graph for Recognition rate without Graph Preprocessing

3.2 Evaluating Recognition rate with different type of images

To test our face recognition system we have divide Yale face dataset into neutral face(63), non neutral face(102) i.e faces with expression variations like happy, sad, surprised, sleepy, and winking and all the images present in dataset. The neutral faces gave the best recognition rate of 85% followed by non neutral faces with 78%. Over all for the entire dataset we got 79%. For all three distance measure we got the same recognition rate. For this experiment the number of signature image remains same for both type of images in the dataset, moreover all the signature images are from neutral face.
The bar graph for the comparison with all the images of the Yale dataset are shown below.

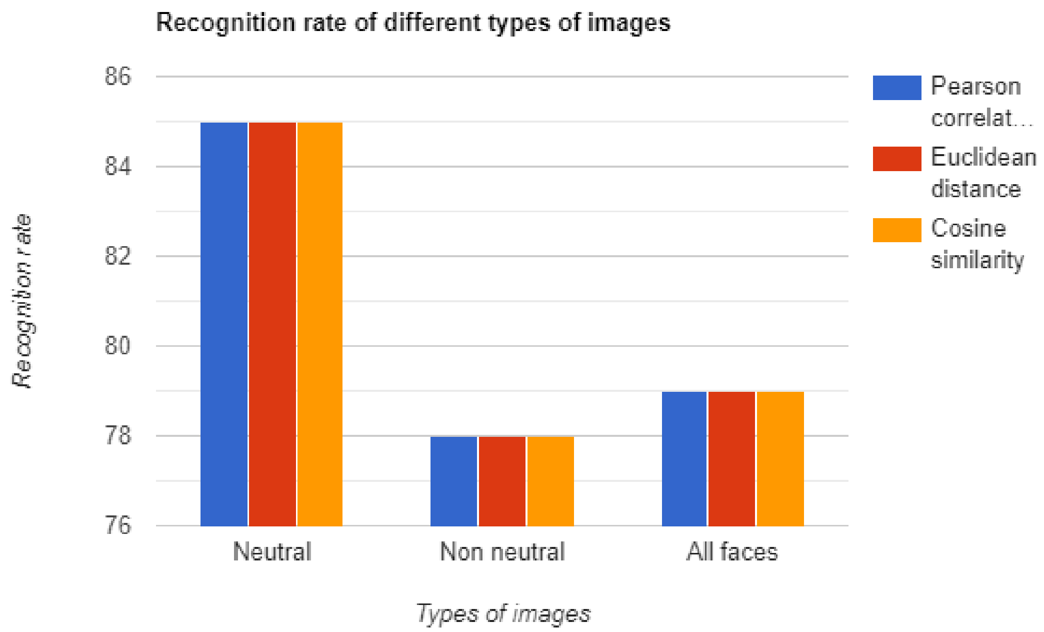

Figure 3.7: Bar-Graph for recognition rate with different types of images

We tested our system with yale face dataset, we used 1 image from each of the 15 subjects, stored them in the database, and tested them against 150 faces with varied expressions, faces with glass on and we got the following confusion matrix, here the highlighted boxes are true positive.

	Predicted0	Predicted1	Predicted2	Predicted3	Predicted4	Predicted5	Predicted6	Predicted7	Predicted8	Predicted9	Predicted10	Predicted11	Predicted12	Predicted13	Predicted14
Actual 0	6	2	0	0	1	0	0	0	0	0	0	1	0	0	0
Actual 1	0	9	0	1	0	0	0	0	0	0	0	0	0	0	0
Actual 2	0	0	7	1	0	0	1	0	1	0	0	0	0	0	0
Actual 3	0	0	0	9	0	0	0	0	0	0	0	0	0	0	1
Actual 4	0	0	0	0	9	0	0	0	0	0	0	1	0	0	0
Actual 5	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0
Actual 6	0	0	0	0	0	0	6	0	1	1	0	0	0	0	2
Actual 7	0	0	0	1	0	0	0	7	1	0	0	0	0	0	1
Actual 8	0	0	0	0	0	0	1	0	8	1	0	0	0	0	0
Actual 9	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0
Actual 10	0	0	0	0	0	0	0	0	0	0	8	1	0	1	0
Actual 11	0	0	0	0	1	1	0	0	0	0	0	8	0	0	0
Actual 12	0	0	0	0	0	0	1	0	0	1	0	0	8	0	0
Actual 13	0	0	0	0	0	1	0	0	0	0	0	1	0	8	0
Actual 14	0	0	0	0	0	1	1	2	0	0	0	0	0	0	6

Figure 3.8: Confusion Matrix for Yale Face Dataset

For Yale dataset our model had the accuracy of 97% with 81% as precision the False positive rate and False negative rate is also low. The recognition rate is at 79% which was affected by non-neural faces, the results are also tabulated below.

Metric	Result
Accuracy	97%
Precision	81%
Specificity/TNR	98.50%
FPR	0.02%
FNR	0.20%
Recognition Rate	79%

Figure 3.9: Result drawn from the above confusion matrix

We have also compared our model with other face recognition technique such as ANFIS-ABC, ANFIS, FFBN,and ANN, for specificity and accuracy, where GAE got 98% specificity and 97% accuracy followed by ANFIS-ABC which got 97% and 95% respectively for the same dataset.

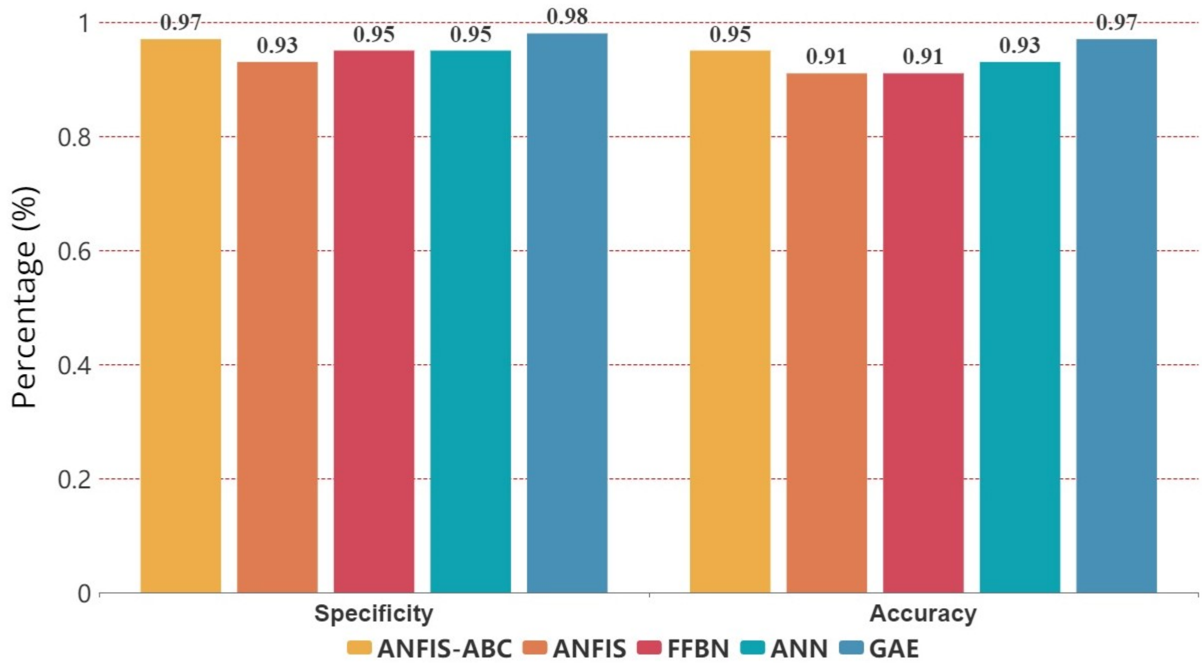

Figure 3.10: Comparison with other method

Chapter 4

Conclusion

In this work, we leveraged graph auto-encoder to get the latent representation of the graph. Google's mediapipe package was utilised to get 468 facial feature key points using which a mesh like face graph was constructed.Mediapipe also displayed better facial detection and more accurate facial feature estimation compared to dlib, which was used in the previous approach, which failed to recognise images with rotated faces This system was tested using the Yale Face dataset, for which we achieved an accuracy of around 97% with a recognition rate of 85% for neutral faces and 79% for the overall dataset. The expression variation and face occlusion seemed to affect the overall accuracy of the system. Compared to our previous approach using node2vec, this system performed better for the same dataset and the overall recognition rate increased from 55% to 79%, with the highest being 85% for neutral faces. Our model has better Specificity and Accuracy compared to four other model for the same dataset.

In the future, the use of edge embedding methods could be explored to improve the recognition rate. Also, an end-to-end face recognition system can be created for better applications. The scope for mask face recognition using this model can also be investigated.

Bibliography

[1] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[2] Srinivas Gutta and Harry Wechsler. Face recognition using hybrid classifiers. *Pattern Recognition*, 30(4):539–553, 1997.

[3] Anil K Jain, Arun Ross, and Salil Prabhakar. An introduction to biometric recognition. *IEEE Transactions on circuits and systems for video technology*, 14(1):4–20, 2004.

[4] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

[5] Chengjun Liu and Harry Wechsler. Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition. *IEEE Transactions on Image processing*, 11(4):467–476, 2002.

[6] Konstantinos Papadopoulos, Anis Kacem, Abdelrahman Shabayek, and Djamila Aouada. Face-gcn: A graph convolutional network for 3d dynamic face identification/recognition. *arXiv preprint arXiv:2104.09145*, 2021.

[7] Marijeta Slavković and Dubravka Jevtić. Face recognition using eigenface approach*. *Serbian Journal of electrical engineering*, 9:121–130, 02 2012.

[8] Renjie Wu, Sei-ichiro Kamata, and Toby Breckon. Face recognition via deep sparse graph neural networks. In *BMVCW*, 2017.