Department of Computer Science

Individual Project - CS3IP16

A Face Authentication System based on Artificial Neural Networks

Student name: Ilektra Pavlopoulou

Student number: 25019583

Supervisor: Dr Hong Wei

29th April 2019

ABSTRACT

The unique ability of face recognition and identification by humans has been for years the focus of scientific research. In the early years the research community in order to master the multitude of aspects of this perception problem and to be able to make computer systems perform this task as effectively as humans, treated face recognition as a common pattern recognition problem. As research progressed the focus moved to the aspect of features representation. In recent times, realising the uniqueness of the face recognition process, the methodology focused on combinations of both recognition and representation techniques, derived from computer learning, computer vision and pattern recognition. This project is an attempt to use predefined tools based on artificial neural network implementations in order to correctly register and identify a face from a camera video stream.

DEDICATION AND ACKNOWLEDGEMENTS

I would like to acknowledge Dr. Hong Wei at the University of Reading for her guidance and assistance throughout the duration of this project.

I would also like to thank Timothy Nicholls for his support and invaluable advice on the implementation of the system.

Finally, I would like to express my gratitude to my father for his dedication and willingness to explain important aspects of this project.

CONTENTS

GLOSSARY OF TERMS & ABBREVIATIONS

ANN - Artificial Neural Network

API - Application Programming Interface

ARJIS - Automated Regional Justice Information System

CNN - Convolutional Neural Network

CSV - Comma Separated Values

EBGM - Elastic Bunch Graph Matching

ERT - Ensemble of Regression Trees

FRS - Face Recognition System

GDPR - General Data Protection Regulation

HCI - Human-Computer Interfaces

HOG - Histogram of Oriented Gradients

ICA - Independent Component Analysis

ID - Identification Data

LDA - Linear Discriminant Analysis

MLP - Multi-Layer Perceptron

PCA - Principal Component Analysis

PID - Project Initiation Document

ResNet - Residual Network

SGD - Stochastic Gradient Descent

SQL - Structure Query Language

SVM - Support-Vector Machine

LIST OF TABLES, FIGURES & EQUATIONS

1. INTRODUCTION

In today's world there is a large number of commercial, security, and forensic applications requiring the use of face recognition. These applications include automated crowd surveillance, access control, mug shot identification (for example driver licenses issuing process), face reconstruction, design of human computer interfaces (HCI) and management of databases with content-based image elements. Face recognition can be used in order to identify people in crowd like in airport terminals or railway stations. A face authentication system could be used to grant access to authorised users in access controlled areas or restricted environments such as in a high security workplace. The demand for quick and reliable personal identification has resulted in an increased interest in biometrics in order to replace traditional identification methods.

In recent years, intense research in the fields of biometrics, pattern recognition and computer vision, the machine learning and computer graphics communities have made remarkable progress in developing systems solutions mimicking the remarkable human ability to recognise people. Face recognition has developed into a major research area. The use of biometrics has addressed key problems inherent to the traditional verification methods. Biometrics make use of human features, such as the characteristics of a face, which can be used to effectively identify and verify a person's identity.

By definition, a face recognition system (FRS) is a computer application capable of automatically identifying or verifying a person from a digital image or a video frame from a video source [1]. A face image or image sequence is given as input to the FRS, then the FRS applies selected methods to match the given input with face images stored in the database.The FRS implements a method of identification of individuals, on the basis of their unique facial characteristics. Facial recognition system can capture multiple images in a second, compare them and produce results.

In this project the aim is to build a face authentication system able to operate in real time. A face recognition process was implemented in order for this system will be able to determine a person's identity by comparing an unknown face with the images of known individuals contained in a provided data set through an Artificial Neural Network (ANN) based classifier.

1.1 Organisation of the Report

This report will discuss the steps that were taken in order to develop the system mentioned above. First, a detailed description of the problem being addressed will be provided including a problem statement which will establish the criteria by which the problem would be validated and accepted as being adequately solved. Following, the research that was conducted on the subject will be detailed through discussing the existing literature that is relevant to this project in order to further the understanding of the approaches that could be applied and the selected solution approach will be provided. Subsequently, an extensive description of the design and implementation of the system based on the solution approach will be detailed, followed by a section providing information on the tests that were performed upon the completion of the implementation. Next, discussion will take place which will go through the results that were obtained from testing, the limitations of this project and personal reflection. Furthermore, a section addressing the social, legal, health and safety and ethical issues concerning this project will be provided. The last section will briefly restate the project objectives and will make conclusions about the project work and results as well as suggest future improvements for the developed system.

2. PROBLEM ARTICULATION & TECHNICAL SPECIFICATION

The problem being addressed in this project is the development of a face authentication system capable of identifying enrolled users in real time through the implementation of a neural network based classifier. The face recognition process that the system should follow is a series of several related tasks which are sequenced as follows.

1. Examine an image and detect all the faces in it.
2. Focus on each face and be able to understand that even if a face is slightly at an angle or under uneven lighting conditions, it is still the same person.
3. Be able to pick out unique features of the face that can be used to differentiate it from other faces, such as the size of a person's eyes or nose.
4. Compare the unique features of that particular face to all the faces that are already known to determine the person's identity.

The human brain has a natural ability to perform these tasks automatically and instantly. A human can quickly and easily recognise a known face even in crowded or low lighting environments. Even though a great deal of progress has been made, today computers are not capable of this sort of high-level generalisation. Therefore it is needed to implement the steps described above as separate processes that work in a pipeline fashion. Facial recognition makes use of technology for the measurement of different facial characteristics and generates a unique reference in digital form, which can be used to establish identity of an individual and then verify it in the future. The set of hardware and software used for the purpose is called Facial Recognition System [1]. An FRS can identify people by processing their digital images if their facial recognition identity has been pre-established. The system takes advantage of digital images or still frames from a video source, which are taken through the facial recognition algorithm in order for the system to produce a match in the case that it has already been trained on the individual user. Conversely, if the network has not been trained on this user, a match will not be found. Face recognition is in principal a visual pattern recognition problem. However, face recognition differs from typical pattern-recognition problems such as character recognition. In classical pattern recognition, there is a relatively small number of classes and a large number of samples per class. With many samples per class, algorithms can easily classify samples not previously seen by interpolating among the training samples. On the contrary, when it comes to face recognition, there is a large number of classes which represent each individual and frequently a small number of samples per person which means that there are few images. Algorithms must be able to recognise faces by extrapolating from the training samples. In numerous applications there can be only one image used as a training sample of each person [2].

In detail, an FRS with the input of a face image should search in the connected dataset in order to find a match and output the individual's identification data. An FRS generally consists of four stages as depicted in Figure 1: face detection, face alignment, feature extraction and matching, where localisation and normalisation in the form of face detection and face alignment are processing steps before face recognition in the form of facial feature extraction and matching is performed [3]. The goal of the system is to acquire the ability to perform facial recognition in real time. The system should be able to accept a live video stream from which the user's face would be detected and used to perform recognition through a feature matching algorithm. The system must communicate with a provided data set of face

images so that the chosen algorithm would be able to train before attempting face recognition. In Figure 1 which was designed for the purposes of this section, the intended system process for face recognition can be visualised.

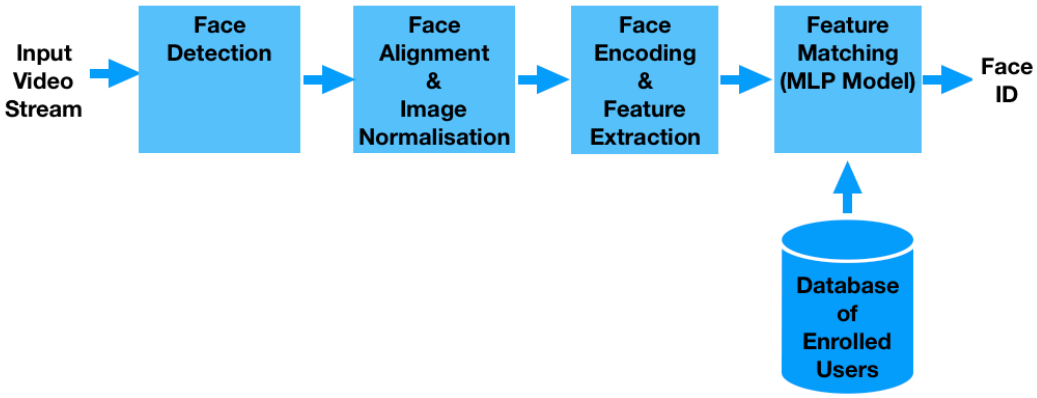

Figure 1 - System process for face recognition

The potential solutions to be assessed will be required to fulfil the following requirements:

- The system should be able to recognise a face from frames extracted from a live video stream
- The system should implement a face detection mechanism in order to spot the face in the frame acquired from the live video stream
- The system's core operation will be based on the ability to recognise a face and then measure the various features of the face
- The system should implement a feature extraction algorithm based on the fact that every face has numerous, distinguishable landmarks that make up facial features. Based on the fact that each human face has approximately 80 landmarks as nodal points, the algorithm should be able to use a number of them in order to extract useful measurements, such as distance between the eyes, width of the nose, the shape of the cheekbones and the length of the jaw line. The nodal points measured should create a set of numerical values, representing the face which will be stored in the dataset.
- For the system to give concise detection the image captured needs to be of a face that is looking more or less directly at the camera (using a frontal face detector), with little variance of light or facial expression from the image stored in the dataset.
- The features extraction subsystem should be able to effectively compare its output values to those kept in the data base containing the faces images archived values.
- An ANN must be developed and trained on the classification problem of face recognition.
- The classifier must achieve matching the given input with the images in the dataset and accomplish recognition of the face provided the face is enrolled in the system.
- The system should have a means of enrolling a user which should incorporate an image acquisition process.

To address the above specification, the desired system must perform in order to create a solution that would be based on the following acceptance criteria which would establish the validity of the problem solution:

- Computer software will be implemented as the interface to the system.The software should integrate libraries that provide face detecting and image manipulating functions.
- Accuracy: Correctly recognise faces at least 90% of the time.
- Communication/Feedback: System will produce output to inform the user of successful or unsuccessful recognition.
- Execution: System will return a result within an acceptable timeframe (<10 seconds).

In summary, a number of requirements were determined for the implementation of this project. The main concern for this system is that face recognition is a technology that is never 100% accurate, therefore the algorithm implemented should aim to reach as high accuracy as possible meaning that accuracy should not be less than 90%. Upon successful recognition the system output should be the face ID and the corresponding user name of the enrolled user as well as the probability statistics for the match. Upon unsuccessful recognition the system output should inform the user that there are no matching faces in the data set. The problem would be adequately solved when the system built is able to recognise a person's face in real time, by comparing the facial features of the person in the live video stream to the ones provided in the data set and successfully finding a match through utilising a neural network as a classifier.

3. LITERATURE REVIEW

Before making a decision on the final solution approach, it was necessary to conduct thorough research regarding the existing literature that was relevant to this project in order to come to a better understanding of the problem topic and the potential approaches that could be implemented for its solution. Initially, the development of face recognition approaches throughout history was considered following with the most relevant and effective face recognition techniques in the present day.

3.1 Progression of Face Recognition

Face recognition is a relevant subject in multiple computer science sectors such as pattern recognition, neural networks, computer graphics, and image processing [4].

As far back as in the 1960s engineering began to show interest in implementing technology capable of face recognition. Bledsoe designed and implemented a semi-automatic system as a first attempt to achieve face recognition [5]. Initially, a number of face coordinates were defined on a face photograph by a human operator, and subsequently computers used this information to perform recognition [6]. The system could classify photos of faces by hand using period available technology what's known as a RAND tablet, a device that people could use to input horizontal and vertical coordinates on a grid using a stylus that emitted electromagnetic pulses. The system could be used to manually record the coordinate locations of various facial features including the eyes, nose, hairline and mouth. These metrics could then be inserted into a database. When the system was given a new photograph of an individual, it was able to retrieve the image from the database that most closely resembled that individual [7]. As expected, system performance was unfortunately limited by the technology of the era and available computer processing power. Bledsoe described some of the key problems automated face recognition systems could face, which are still relevant such as variations in illumination, head rotation, facial expression, ageing [8, 9].

In the 1970s, some approaches tried to define a face as a set of geometric parameters and then perform pattern recognition based on those parameters. The first to develop a fully automated face recognition system was Kenade in 1973 [10]. He designed and implemented a face recognition program which ran on a computer system designed specifically for this purpose. The algorithm that was implemented extracted sixteen facial parameters automatically such as lip thickness and hair colour in order to identify faces automatically. This method performed a correct identification with a positive hit rate of 45-75%. As with Bledsoe's system, the actual biometrics had to still be manually computed.

In the 1980s, new approaches were introduced, some of them still being used in the present day. For example, researchers build face recognition algorithms through the use of artificial neural networks for the first time [11]. The first mention to eigenfaces in image processing, a technique that became the dominant approach in the following years, was made by L. Sirovich and M. Kirby in 1986 [12]. Sirovich and Kriby were able to show that feature analysis on a collection of facial images could form a set of basic features. They were also able to show that less than one hundred values were required in order to accurately code a normalised face image. The eigenface approach was based on Principal Component Analysis (PCA) which revolves around dimensionality reduction. The goal of this approach was to represent an image in a lower dimension without losing a significant amount of information, and then reconstruct the image [13].

In the 1990s, eigenfaces had become the state-of-the-art approach for face recognition. Mathew Turk and Alex Pentland published a paper in which eigenfaces were implemented for face recognition where the algorithm being developed was able to locate, track and classify a subject's head [14]. Their approach was constrained by technological and environmental factors, however it was a significant breakthrough in terms of proving the feasibility of automatic facial recognition [15]. In the US, the Defence Advanced Research Projects Agency (DARPA) and the National Institute of Standards and Technology rolled out the Face Recognition Technology (FERET) program, in order to encourage the commercial face recognition market. The project involved creating a database of facial images. The database was updated in 2003 to include high-resolution 24-bit colour versions of images. Included in the test set were 2,413 still facial images representing 856 people. The hope was that a large database of test images for facial recognition would be able to inspire innovation that might result in more powerful facial recognition technology.

In 2002, in order to improve security for the needs of the Super Bowl game in the USA, officials installed and operated a facial recognition system. This was a major test of the efficiency and effectiveness of available technology. Although the system was able to trace a number of people wanted by the police, the conclusion was that the system was a failure. This was due to the face that it alerted operators incorrectly, producing a large number of false positives that lead to unnecessary and time wasting cross-checks. The backlash from the system critics indicated that face recognition technology had not matured enough for reliable application within the field. One of the big issues at the time was that face recognition did not yet work well in large crowds, an essential prerequisite to using face recognition for event security.

In 2009, in the US the Pinellas County Sheriff's Office implemented a database that had access of the photo archives of the state's Department of Highway Safety and Motor Vehicles (DHSMV). By 2011, about 170 deputies had been outfitted with cameras that let them take pictures of suspects that could be cross-checked against the database. Evaluation of the usage of the system concluded that the system contributed in more arrests and criminal investigations than would have otherwise been possible.

In 2010, with the evolution and rise in social media platforms, Facebook began implementing facial recognition functionality as a back-end process in order to help identify people whose faces may be featured in the photos that Facebook users update daily. The activation of this feature started a controversy, with the news media and a myriad of privacy-related articles in the press. Facebook users did not seem to mind, perhaps not immediately realising the deeper implications. Statistics indicated that there was no apparent negative impact on the website's usage or popularity, where millions of photographs are uploaded and tagged using face recognition each day.

In 2011, the government of Panama, with the assistance of the US Government, authorised a pilot program of a facial recognition platform installation in order to monitor illicit activity in the country's main airport, known as a hub for drug smuggling and organised crime. Shortly after system activation, multiple Interpol suspects were identified and arrested. The Face Recognition system implemented at Panama's main airport is one of the largest biometrics installations at an airport to date.

The Automated Regional Justice Information System (ARJIS) the US, began in 2014 to supply partner agencies with a mobile platform supporting face recognition for law enforcement based on a FaceFirst's facial recognition platform. ARJIS promotes sharing of data and information among local, federal and

state law enforcement services, which caused a major problem that the instant identification of people who had no identification documents were unwilling to be identified. Some of the services that are using the complex criminal justice enterprise network to identify suspects in the field are POLICE, DOJ, FBI, DEA, CBP and U.S. Marshalls [16].

3.2 Face Recognition Techniques Review

An ANN was chosen to be the classifier for the system to be developed, however there is a number of different approaches that could serve as solution options when addressing the problem of face recognition. Since face recognition has been a topic of research for decades [17], multiple techniques have been developed throughout the years. These methods can be classified in two groups that include appearance-based methods and model based methods. Former methods use holistic texture features that are applied to either whole-face or specific regions in a face image whereas the latter methods employ the shape and texture of the face. The most well known methods of appearance-based face recognition are Principal component analysis (PCA), independent component analysis (ICA) and linear discriminant analysis (LDA) [18]. Researching and reviewing these methods could help to gain a better understanding of the problem being addressed and could provide ideas that would contribute to its solution. Some of the most popular and effective face recognition techniques are described below.

Principal Component Analysis (PCA)

PCA is used in the Eigenface algorithm for dimensionality reduction in order to find the vectors that best account for the distribution of face images within the entire image space [14]. These vectors define the subspace of face images which is called face space. All of the faces in the training set are projected onto the face space in order to find a set of weights that describes the contribution of each vector in the face space. To perform face recognition by identifying a test image, the projection of the test image onto the face space is required in order to obtain the corresponding set of weights. By comparing the weights of the test image with the set of weights of the faces in the training set, the face in the test image can be successfully identified.

The key procedure in PCA is based on Karhumen - Loeve transformation [13]. If the image elements are considered to be random variables, the image may be seen as a sample of a stochastic process. The Principal Component Analysis basis vectors are defined as the eigenvectors of the scatter matrix S_T, also known as covariance matrix shown in eq. (1).

$$S_T = \sum_{i=1}^{N} (x_i - \mu)(x_i - \mu)^T. \qquad \textbf{(1)}$$

The transformation matrix W_{PCA} is composed of the eigenvectors corresponding to the d largest eigenvalues.

Independent Component Analysis (ICA)

While PCA de-correlates the input data using second-order statistics such as the covariance/scatter matrix, which results into compressed data with minimum mean squared re-projection error, ICA minimises both the second-order and higher-order dependencies in the input. ICA has the goal to find a linear representation of non-gaussian data so that the components are statistically independent, or as

independent as possible [19]. The fundamental restriction in ICA is that the independent components must be non-gaussian for ICA to be possible [20].

Linear Discriminant Analysis (LDA)

LDA is also known as the Fisherface algorithm [21] derived from the Fisher Linear Discriminant (FLD), which uses class specific information. By defining different classes with different statistics, the images in the learning set are divided into the corresponding classes. Then, techniques similar to those used in the Eigenface algorithm are applied. The Fisherface algorithm results in a higher accuracy rate in recognising faces in comparison to the Eigenface algorithm.

The Linear Discriminant Analysis finds a transform W_{LDA}, given in eq. (2).

$$W_{LDA} = \arg\max_{W} \frac{W^T S_B W}{W^T S_W W}, \qquad \textbf{(2)}$$

In eq. (2) S_B represents the between-class scatter matrix and S_W is the within-class scatter matrix, given in eq. (3) and (4) respectively.

$$S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T, \qquad \textbf{(3)}$$

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in X_i} (x_k - \mu_i)(x_k - \mu_i)^T. \qquad \textbf{(4)}$$

In the above expressions, N_i is the number of training samples in class i, c is the number of distinct classes, μ_i is the mean vector of samples belonging to class i and X_i represents the set of samples belonging to class i.

Elastic Bunch Graph Matching (EBGM)

In EBGM, individual faces are represented by a rectangular graph, each node labeled with a set of complex Gabor wavelet coefficients, called a jet. The EBGM algorithm recognises faces by matching the probe set represented as the input face graphs to the gallery set that is represented as the model face graph. Fundamental to the EBGM is the concept of nodes. Essentially, each node of the input face graph is represented by a specific feature point of the face. For example, a node represents an eye, another node represents the nose and the concept continues on by representing all the remaining facial features. Therefore the nodes for the input face graph are interconnected to form a graph-like data structure which is fitted to the shape of the face. When recognising a face of a new image, each graph in the model gallery is matched to the image separately and the best match indicates the recognised person [22].

Support-Vector Machine (SVM)

Given a set of points belonging to two classes a Support Vector Machine (SVM) finds the hyperplane that separates the largest possible fraction of points of the same class on the same side, while also maximising the distance from either class to the hyperplane. According to Vapnik [23], this hyperplane is called Optimal Separating Hyperplane and it can minimise the risk of misclassifying not only the examples in the training set but also the unseen examples of the test set. A feature extraction algorithm,

which is usually PCA, has to be used first on the face images and then discrimination functions between each pair of images are learned by the SVM. Following that, the disjoint test set enters the system for recognition [24].

3.3 Artificial Neural Networks

Artificial neural networks (ANN) have been successfully applied for solving signal processing problems in 20 years [25]. Researchers proposed many different models of artificial neural networks. A challenge is to identify the most appropriate neural network model which can work reliably for solving a realistic problem.

The materialisation of an Artificial Neural Network (ANN) has its base on that they have been implemented as generalisations of mathematical models of biological nervous systems. McCulloch and Pitts in 1943 [26] introduced a simplified analysis of the neuron and the first attempts of implementing neural networks used distributed parallel processing in a connectionist model. In neural networks the basic processing elements that implement them are called artificial neurons or nodes or simply neurons. The effects of the synapses of the neurons are effectively simulated by a simplified mathematical model. A transfer function formula represents the nonlinear characteristic exhibited by neurons whilst the effect of the associated input signals is represented by connection weights that are modulating the effects of the associated input. Therefore the output of the neuron simulator is the computational result of the transformation performed by the transfer function of the weighted sums of the input signals. This result is characterised as the neuron impulse. By careful adjustment of the used weights in relation to the chosen learning algorithm is what achieves the desired learning capability of an artificial neuron. In order to configure a neural network so as to produce a desired useful output, it has to be configured (trained) so that for a given set of inputs the output will be a meaningful result. The easy way to achieve this will be to use parameterisation based on previous knowledge that has given the desired results and set the neurons weights accordingly. The previous is not always possible and in order to achieve the desired results we would have to train the neural network using as inputs “teaching patterns” and provide learning rules allowing the change of weights to lead to the desired results.

Neural Networks, considering their learning methodology can be classified into three distinct categories, those of the supervised learning, those of the unsupervised learning and those of the reinforced learning. The first category an input set applied together with a set of desired responses one per node at the output layer. Following what could be characterised as a forward pass action leads to identification of deviations and errors from the expected or desired response and this is so for each node of the output layer of the network. The deviations from the desired output are used as a guide to the required changes for the weights according to the learning rule. The term supervised network derives from the fact that the tuning of the output response on individual output nodes so as to obtain the desired output is a result of the use of an “external teacher” [27].

In a way similar to that intrinsic to human brain functioning, humans learn from their mistakes, neural networks also could also learn from their mistakes by providing feedback to the processing engine to adjust accordingly to input patterns such as to reduce errors in the output of the network. The complexity in the implementation of this kind of neural networks which are called auto associative neural networks involves local minima which are the main issue with back propagation algorithms necessary for the implementation of such networks. A simpler implementation is the Feed-Forward networks. As previously discussed in simplified description an artificial network can be visualised as a collection of interconnected processing units which communicate with each other via message passing

through weighted connections. The functioning of each unit or neuron or node, is a task of receiving input from its neighbours outputs or external sources and produce an output as a result of computation which will propagate to the other units. During the network's operation the state of activation of a unit which effectively characterises the output of the unit can be updated in a synchronous or asynchronous manner. In the first case of synchronous update all units will conform simultaneously, whilst in the second case only one of the units will update at a given time frame. There is also the need to apply a rule that will define the effects of the inputs to the activation of the unit, this rule can be a linear, semi-linear function or S-shaped (sigmoid) function. In order for a neural network to perform so that a set of inputs will result in a desired set of outputs it has to be trained. A way to achieve this would be to apply weight values derived by prior knowledge, the other is to feed training inputs and apply a learning rule as the Perceptron learning, in order to produce the correct weight figures [28].

In the field of Pattern Recognition and especially Face Recognition the processing could be done either by using conventional methods like arithmetic algorithms so us to detect whether the given pattern matches an existing one or using neural networks that can tolerate noise and, if trained properly, will produce correct results even for unknown patterns. This is because neural networks if constructed with the correct architecture and provided they are trained correctly with select data, they will produce impressive results compared to straightforward methods like traditional arithmetic algorithms capable of limited tolerance to factors like noisy patterns. Without need to understand the internal mechanisms, the neural networks ability to learn by example, makes them flexible and powerful as processing engines tackling complex tasks that otherwise we would have to devise and implement complex algorithms to the same effect.

Feed-forward networks are networks that data strictly flows in one direction, from input to output nodes, the interconnection of their nodes do not form loops i.e. no feedback connections exist to extend from outputs of nodes to inputs of nodes in the previous or same layers. These type of neural networks have already been used successfully for human face recognition. The structure of a feed-forward neural network consists of layers, which are many function compositions, followed by a loss function that measures how well the neural network models the data, in effect how accurately the neural network performs a process. The Multilayer Perceptron (MLP) is the most common implementation of feed-forward neural network architecture. The units of an MLP neural network are organised in layers and the nodes that compose the layers are fully connected to each node of the subsequent layers, therefore the layer of the inputs passes the inputs to subsequent layers. At a function level the input layer nodes have linear response to activation whilst the hidden unit nodes have nonlinear response function at the output again the layer nodes have linear response function. The three layer MLP is the simpler implementation of this type of neural network [29] and can be visualised in Figure 2.

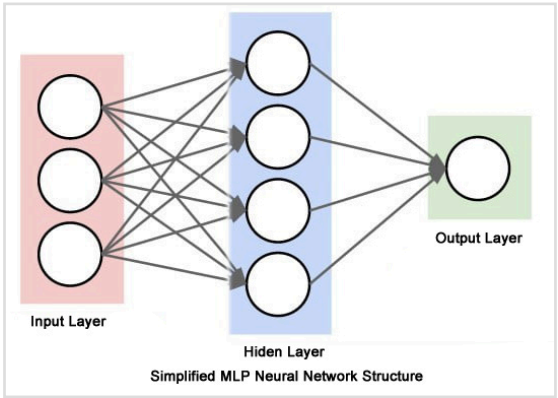

Figure 2 - MLP layer structure

Conclusively ANNs can solve classification problems such as face recognition very efficiently and offer a wide variety of options for the implementation of face recognition. The face images have to be pre-processed by a chosen feature extraction algorithm. The feature vectors are used to form a new data set which is split into training and a testing set. The ANN takes the features vector as input, and trains the network to learn a complex mapping for classification, which will avoid the need for simplifying the classifier. After the network has trained, it is able to perform face recognition. Initially, a face has to be presented to it for detection. Once a face is detected, the face region is cropped from the image to be used as "Probe" into the knowledge to check for possible matches. The face image is pre-processed for factors such as image size and illumination and to detect particular features [30]. The data from the image is then matched against the knowledge [31]. The matching algorithm will produce a similarity measure for the match of the probe face into the knowledge.

3.4 Face Detection

Face detection is a computer technology that is based on learning algorithms to allocate human faces in digital images [32]. The process takes images or video sequences as input and locates face areas within these. This is done by separating face areas from non-face background regions.
Face detection can be regarded as fundamental part of face recognition systems according to its ability to focus computational resources on the part of an image containing a face. The process of face detection in images is complex because of variability present across human faces such as pose, expression, position and orientation, skin colour, presence of glasses or facial hair, differences in camera gain, lighting conditions and image resolution [33].

In a face authentication system, after the chosen classification model has trained, a face detection algorithm has to be utilised in order to spot the face in the video stream and use the frame containing the face for feature extraction and matching. Also for certain face recognition techniques where it is necessary for the program to go through the image data set and extract facial features to use for matching, face detection has to be performed first on each image in order to determine if it indeed contains a face and then run the feature extraction algorithm.

Face recognition also depends on using the right face detector. It is possible to train an ANN for this purpose, although several simpler techniques are available which are not deep learning based. These include the OpenCV based face detectors and the one by DLIB. The former is based on Haar Cascades, a seminal work by Viola and Jones, while the latter is based on Histogram of Oriented Gradients combined with a linear SVM classifier.

4. SOLUTION APPROACH

In this section, a number of steps that have to be taken towards implementation will be outlined in order to form a process that will successfully serve as a solution to the problem being addressed. These steps represent the most important stages of the face recognition process.

4.1 Process Stages

Image Acquisition

The process of image acquisition should be incorporated in the system as part of the enrolment procedure. Enrolment should allow users who are not stored in the system to enter their personal information and have their images taken and stored for future use. For each user in the dataset, a sufficient number of face images should be taken in slightly different poses and angles in order for the classifier to be trained on a wide variety of data and result in identification being more accurate. A class for image acquisition should be created for the case when a new user is introduced to the system through enrolment. This class should access the system's camera and provide a live video stream from which it would detect the new user's face and automatically take a number of pictures that it would then add to the existing image data set.

Face Detection

Face detection is an essential part of the face recognition process. In order to identify a face in a given image, it is necessary to first acknowledge the fact that that a face exists in the image. Therefore the developed system should include a face detection mechanism in order to fulfil the needs of this stage of the process. In more detail, the system should utilise face detection in order to find the face of the user in the frames provided by the live video stream. In the case of enrolment, face detection would locate the user's face in order for the required images to be taken and stored before feature extraction is performed. In the case of identification, face detection would spot the face in the frame and activate the identification process. Using libraries could assist with in the implementation of face detection. For example, the Dlib library provides a frontal face detector which could be suitable for this task as well as the OpenCV library which offers Haar Cascades. Face detection using Haar feature-based cascade classifiers is said to be an effective object detection method proposed by Paul Viola and Michael Jones in the paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images and is then used to detect faces in other images [34]. However, the Haar cascade classifiers have been reported to detect a high number of false positives, meaning that they often misclassify other objects as faces [35]. The Histogram of Oriented Gradients method suggested by Dalal and Triggs in their seminal 2005 paper, "Histogram of Oriented Gradients for Human Detection" demonstrated that the Histogram of Oriented Gradients (HOG) image descriptor and a linear Support Vector Machine (SVM) could be used to train highly accurate human face classifiers [36].

Face Alignment/Image Normalisation

Face alignment, or face landmark localisation, is the process of identifying the geometric structure of human faces in digital images [37]. This process is able to extrapolate a set of key points from a given face image. These key points are called facial landmarks and are used to describe the shape of the face attributes such as the following facial features: eyes, eyebrows, nose, mouth, and chin. Face alignment is

a step that can help to achieve image normalisation which is necessary for this system. The process to accomplish this would involve translation, scaling and rotation of each image. Through image normalisation, all the images in the system's dataset will be uniform which is crucial in order for the ANN to be able to train on them and later on make decisions based on comparing input images against the dataset. Therefore face alignment should be added to the system's process as part of data normalisation.

Feature extraction

Feature extraction is an integral part of the face recognition process [38] and similar to face detection, it has to be used in two different stages of the program. First, in order to extract the features form every face image in the data set so that they can be forwarded to the classifier for training and second in order to extract the features of the face presented to the system for identification after the classifier has trained and is able to perform feature matching. Although PCA could be implemented in order to perform feature extraction on the face images, it was decided that the focus of this project should be towards the development of the classifier based on ANN methodologies in order to achieve face recognition. Therefore a more straight forward feature extraction approach has to be used instead, such as the use of a pre-trained model that generates unique measurements for each face and then stores them in the system as a new dataset.

ANN classifier training for feature matching

The method that was chosen to implement in order to achieve facial recognition was deep learning through an artificial neural network. The reason behind this decision was that a neural network is closer to the way the human brain works which means that it is a comparative system and also is able to work efficiently on large data sets. Face recognition is a non-linear problem, therefore it was decided to be approached by a non-linear neural network since it can handle classification and pattern recognition very well. ANNs have the ability to learn and model non-linear and complex relationships, which is very important due to the fast that in real-life many of the relationships between inputs and outputs are non-linear as well as complex. After learning from the initial inputs and their relationships, it can infer unseen relationships on unseen data as well, thus making the model generalise and predict on unseen data. In principle, the back-propagation ANN [39] may be trained to recognise face images directly, however for even a moderate image size the network can be quite complex and therefore difficult to train. In order to reduce complexity, feature extraction will first be performed on the face images and the feature vectors will be used as the dataset to train the ANN, which will give the opportunity to the network to operate in a more common classification mode [40]. There are multiple ways to develop an ANN, such as Convolutional Neural Networks (CNN) or Multi Layer Perceptrons (MLP). For this project, the MLP was chosen as the most suitable model to implement. There are libraries that could help to create and customise a neural network such as Keras and TensorFlow. Keras is an open-source neural network library written in Python which contains numerous implementations of commonly used neural network building tools such as layers, activation functions and optimisers. It also supports other common utility layers like dropout and batch normalisation in order to help the developed model achieve optimum results.

Menu

A menu will be created for the system in order to provide the user with the option to either enrol if it is the first time they are presented to the system or proceed to identification in the case where their images

are already stored in the system's dataset. The user will be able to choose between the two options by providing input to the system through pressing a different key for each option. After one option is selected, the system will automatically run the corresponding process, directing the user to either image acquisition or face identification. In the option of enrolment, a Terms and Conditions form will be presented to the user requiring to be read and accepted in order for the system to ask for the user's information and redirect the user to the image acquisition process.

4.2 Solution definition

In summary, the chosen solution will implement an MLP model as the classifier to achieve real time face recognition for the face identification system. The first step for the software to be developed would be the acquisition of live video stream in order for the users to be able to witness the process happening in real time. The system should connect to a database in order to store the information that the user enters. Following, face detection would be applied on the frame attempting to locate the user's face. After a face has been successfully detected in the image, face alignment will be performed before the taken image is stored in the system in order to collect uniform data. The program will then proceed to compute a number of feature vectors that would represent the user's face as face descriptors and feature extraction would be performed by storing these feature vectors in a file that would be used as the new dataset. Subsequently, the feature dataset will be divided into a training and testing set before it is passed to the classifier. The training set will contain a larger amount of the data, while the testing set will be composed of the remaining amount. The feature vectors from the new datasets will be used in order to train the ANN classifier that would be developed. After training the system will provide live video stream again and will use the ANN classifier to determine if the face in the frame is known. Upon successful identification, the system will connect to the database in order to fetch the data that correspond to the identified user.

5. IMPLEMENTATION

In this section the implementation of each of the components of the system will be discussed and focus will be given on important sections of each while aiming to explain the logic behind their utilisation.

5.1 Enrolment and Image Acquisition

As mentioned in the solution approach, the system should provide a means that would allow new users to enrol to the database by entering their name and subsequently activate an image acquisition procedure where face images of the user would be taken as input and stored in the system. To implement this, two folders are created. The first folder is named faces and is intended to store a subfolder for each enrolled user containing all the acquired face images after they are processed. The second folder is named new_faces and is designed to temporarily store subfolders of the new users face images until they are processed by the system. In order to achieve this, a new folder path is initially defined for 'new_faces' to be created in the system. Following this, a face ID is allocated to the new user by the system that matches their folder name which is achieved by counting the length of the existing subfolders in 'faces' and automatically incrementing when creating subfolders in 'new_faces'. Subsequently, the user is prompted to enter their full name, then the system connects to the database in which all the user information is stored and automatically updates the 'faces' table by inserting a new row containing the user name and user ID. This is achieved through executing the query "INSERT INTO faces VALUES (‘‘‘+str(face_id)+’’’,’’’+name+’’’);" and committing to the database. Following this, the initial image number is automatically set to 1 and the system is set to take 20 images as input. This process is demonstrated in Figure 3 below.

```
@staticmethod
def __create_folder(folder_name):
    if not os.path.exists(folder_name):
        os.mkdir(folder_name)

def start(self):
    self.__create_folder(self.FACE_DIR)
    # get face id
    face_id = len(os.listdir("faces")) + len(os.listdir("new_faces"))
    face_id = int(face_id)
    face_folder = self.FACE_DIR + str(face_id) + "/"
    self.__create_folder(face_folder)
    name = input("Enter Name")
    connection = sqlite3.connect("face_db.db",)
    query = "INSERT INTO faces VALUES ('"+str(face_id)+"','"+name+"');"
    print(query)
    cursor = connection.execute(query)
    connection.commit()
    cursor.close()
    init_img_no = 1
    img_no = init_img_no
    total_imgs = 20
```

Figure 3 - Image acquisition implementation

After the name and user ID are successfully set and stored in the database, the system opens two windows initiating live video capture by activating the camera connected to it. In order for the system to be able to access the camera and provide live video stream, the OpenCV method cv2.VideoCapture() was utilised. This method creates a VideoCapture object that requires the device index to be passed as its argument. Device index is the number that specifies which camera is being used. In this case, one camera is connected to the system therefore, '0' is passed in order for the method to activate the only camera attached to the computer. Following that, two windows open automatically showing the video captured frame by frame. The methods cv2.resizeWindow() are also used to resize the windows in order for them to be visible on the screen simultaneously. The first window is set to perform live face detection.

5.2 Face Detection

Face detection is achieved through the Dlib method dlib.get_frontal_face_detector() which provides a pre-trained model based on HOG (Histogram of Oriented Gradients) and a linear SVM (Support Machine Vector). The method returns an object detector configured to detect human faces that are looking towards the camera. In broad terms, the way this method operates is as follows. The HOG algorithm iterates on every pixel of the given image and calculates the vector gradient of each pixel. This is achieved by examining all the pixels around it in order to assess how dark the current pixel is compared to the pixels directly surrounding it. It then draws an arrow that shows in which direction the image gets darker. After the process has been repeated on every pixel, the algorithm groups the pixels in cells and the cells in blocks, then calculates how many gradients point in each major direction. Each cell is then replaced with the arrow of the direction that is the strongest and the descriptor is assembled as a cell histogram list of all blocks. Then the SVM classifier uses the features extracted by the descriptor. The way in which SVM works is essentially through finding a hyperplane that fits in the middle of two classes. Mathematically, a training dataset can be represented by its feature vectors. Considering a linear separability problem, these vectors belong to only two classes. The objective of SVM is to find a hyperplane that classifies the training vectors correctly [41]. Through applying the HOG algorithm to a number of face images, the Linear SVM model is trained to detect faces in an image. The face detection approach can be visualised in Figure 4. A constraint is set for only one face to be detected at a time. This is done to prevent errors in the system during image acquisition.

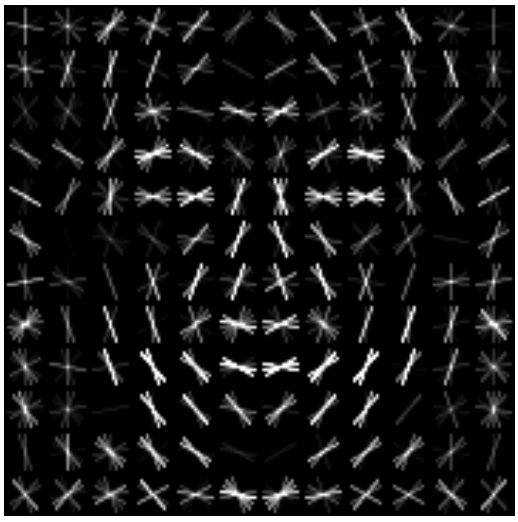

Figure 4 - Face detection through HOG & SVM

5.3 Face Alignment and Image Normalisation

When a face is detected the live video stream, a rectangle is drown around it in the video capture window. Subsequently, the second window is activated providing live face alignment before the images are captured and stored in the system. In order to accomplish face alignment, the Dlib method dlib.shape_predictor()was utilised. This method loads the shape predictor from the "shape_predictor_68_face_landmarks.dat" file which contains the output of the trained shape predictor routine. The face landmark detection algorithm offered by Dlib is an implementation of the paper "One Millisecond Face Alignment with an Ensemble of Regression Trees (ERT)" [42]. This technique utilises pixel intensity differences in order to directly estimate the face landmark positions in an image. These estimated positions are subsequently refined with an iterative process done by a cascade of regressors. Each regressor produces a new estimate from the previous one, aiming to reduce the alignment error of the estimated points on each iteration. The algorithm detects a set of 68 landmarks on a given face in order to estimate the face's pose and perform face alignment. The positions of the 68 facial landmarks are illustrated in Figure 5.

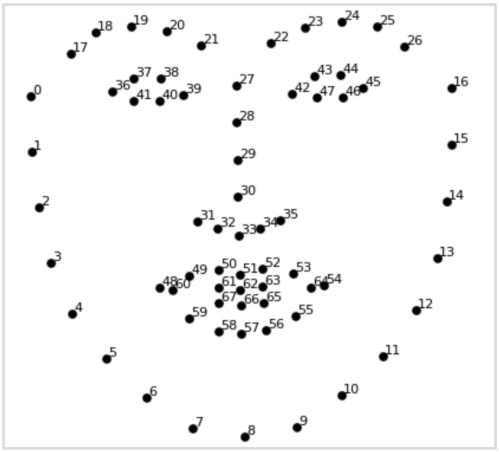

Figure 5 - Face landmarks positions

The Dlib method mentioned above is included as part of an imutils library function called face aligner. This function utilises this Dlib method in order to perform face alignment on the input image and then scales it by taking the ratio of the distance between the eyes in the current image to the ratio of the distance between the eyes in the desired image. The dimensions for the desired image are passed as arguments in the face aligner declaration. This function uses a rotation matrix for rotating and scaling the face in order for the input image to be normalised before it is stored in the system. After normalisation has been performed, each taken image is stored in the 'new_faces' folder. When the set number of images is successfully stored in the system, the video capture windows closes automatically. The routines described are presented in Figure 6.

```
# Open Video Capture
webcam = cv2.VideoCapture(0)
# Create Windows
video_capture = "Video Capture"
aligned = "Alignment"
cv2.namedWindow(winname=video_capture, flags=cv2.WINDOW_NORMAL)
cv2.namedWindow(winname=aligned, flags=cv2.WINDOW_NORMAL)
cv2.resizeWindow(video_capture, 1000, 800)
cv2.resizeWindow(aligned, 300, 300)
cv2.moveWindow(video_capture, 0, 0)
cv2.moveWindow(aligned, 1000, 200)

# Loop Save
while True:
    frame = webcam.read()[1]
    frame_with_instructions = frame
    cv2.imshow(video_capture, frame_with_instructions)
    frame_bw = cv2.cvtColor(src=frame, code=cv2.COLOR_BGR2GRAY)
    faces = self.detector(frame_bw)
    if len(faces) == 1:
        face = faces[0]
        (x, y, w, h) = face_utils.rect_to_bb(face)
        face_img = frame_bw[y:y + h, x:x + w]
        # align the face
        face_aligned = self.face_aligner.align(frame, frame_bw, face)

        # saving the face
        face_img = face_aligned
        img_path = face_folder + str(img_no) + ".jpg"
        cv2.imwrite(img_path, face_img)
        cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 255, 0), 3)
        cv2.imshow(aligned, face_img)
        img_no += 1
    cv2.imshow(video_capture, frame)
    cv2.waitKey(1)
    if img_no == init_img_no + total_imgs:
        break

webcam.release()
```

Figure 6 - Face alignment & image normalisation process

5.4 Feature Extraction

In order to perform feature extraction from a face image, the facial feature vectors should first be computed. The process for this is explained as follows. First, each of the aligned and normalised images stored in the user subfolder in ‘new_faces’ is loaded form the folder through the utilisation of the OpenCV method cv2.imread(). Face detection is applied to the images once again in order to find the face in each image before using the Dlib method dlib.compute_face_descriptor to generate an encoded face image of 128-dimensional feature vectors in numpy array format. This method loads a pre-trained model provided from the file “dlib_face_recognition_resnet_model_v1.dat” which converts the detected face into a 128-dimensional face descriptor. This model is a Residual Neural Network (ResNet) with multiple convolutional layers which is a version of the ResNet-34 network from the paper “Deep Residual Learning for Image Recognition” [43]. The library trained a CNN to generate 128 measurements for each face in order to represent them in the corresponding embedding. The term embedding is the collective name for mapping input features to vectors which in this case these inputs are images containing a subject’s face, mapped to a numerical vector representation. The idea behind this is to reduce complicated raw data, such as images, into a list of computer generated numbers. In Figure 7, the generated feature vectors for a test image are shown.

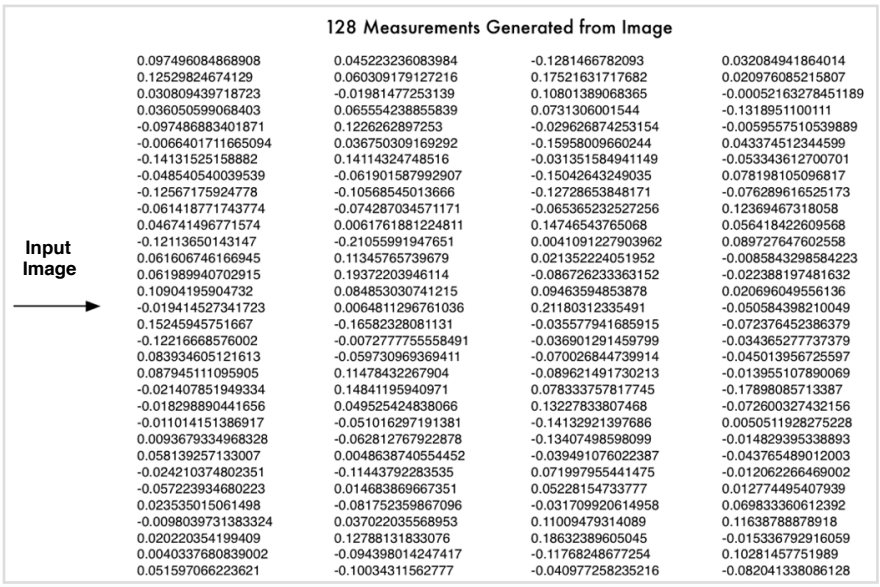

Figure 7 - 128 generated feature vectors for test image

The generated feature vectors for each face image are then stored into a CSV file in the system along with the corresponding ID numbers. When this process is complete, the subfolder containing the user images is deleted form ‘new_faces’ and added to ‘faces’ which stores all the subfolders representing the users in the data set. The corresponding code is pictured in Figure 8.

```
for folder_name in folder_names:
    create_folder(OLD_FACE_DIR + folder_name)
    full_folder_path = NEW_FACE_DIR + folder_name + "/"
    images = os.listdir(full_folder_path)
    for image in images:
        full_image_path = full_folder_path + image
        print(full_image_path)
        img = cv2.imread(full_image_path)
        cv2.imwrite(OLD_FACE_DIR + folder_name + "/" + image, img)
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        try:
            face = detector(img, 1)[0]
        except:
            print("Error")
            continue
        shape = shape_predictor(img, face)
        face_descriptor = face_rec.compute_face_descriptor(img, shape)
        face_descriptor = list(face_descriptor)
        face_descriptor.insert(0, int(folder_name))
        with open(CSV_FILE, "a", newline="") as csvfile:
            writer = csv.writer(csvfile)
            writer.writerow(face_descriptor)
    shutil.rmtree(full_folder_path)
```

Figure 8 - Feature vector generation & feature extraction

After all the face images are encoded, the dataset of feature vectors is loaded from the CSV file and the feature vectors are separated from the user IDs in order to be divided into a training and a testing set that would later be used by the MLP model. 90% of the data form the training set whilst the remaining 10% are grouped into the testing set. This was achieved by the code provided in Figure 9.

```
DATASET_CSV_FILE = "dataset.csv"
features = []
labels = []
with open(DATASET_CSV_FILE, newline="") as csvfile:
    reader = csv.reader(csvfile)
    for row in reader:
        labels.append(int(row[0]))
        features.append(np.array(row[1:], dtype=np.float32))

no_of_faces = len(features)
train_features = features[:int(0.9*no_of_faces)]
train_labels = labels[:int(0.9*no_of_faces)]
test_features = features[int(0.9*no_of_faces):]
test_labels = labels[int(0.9*no_of_faces):]
```

Figure 9 - Dataset division into training and testing sets

Following the dataset division, four new folders are created in the system to store the training and testing datasets of feature vectors (features) and user IDs (labels).

5.5 MLP Model Development

The next step is to train an ANN to compare the embedding vectors of the images stored in the file system with the embedding vector of the image captured from the webcam. The classifier should be trained to take in the feature vectors from a new test image and make a decision based on the question if any of the known users is a suitable match. The type of ANN that was decided to be implemented is a MLP model developed through the utilisation of the Keras API. Keras is a model-level library, which provides high-level building blocks for developing deep learning models. It does not handle low-level operations such as tensor products or convolutions itself. Instead, it relies on a specialised, well optimised tensor manipulation library to do so. This serves as the “back-end engine" of Keras which in this case was TensorFlow. TensorFlow is an open-source software library for differentiable programming across a range of tasks which is vastly used for machine learning applications such as neural networks.

The model type being implemented is ‘Sequential’ which is stated to be the most quick and efficient way to create a model in Keras since it allows to build the model layer by layer [44]. Each one of the network layers has weights that correspond to the layer that succeeds it. The model.add() function is used in order to add layers to the model. For this model, three layers were added; an input layer which was followed by one hidden layer along with an output layer. The layer type chosen for this model is ‘Dense’ which is a standard layer type that reportedly works well for most cases. In a dense layer, all nodes in the previous layer connect to the nodes in the current layer. The first layer added to the model which serves as the input layer requires an input shape. The input shape specifies the number of rows and columns of the input, which in this case is the folder containing the feature vectors and user IDs. The number of columns in the input is set to 128 due to the number of the feature vectors representing every face in the data set. There is no value passed after the comma following the input shape, which indicates that there can be any amount of rows since the dataset is not fixed but will keep becoming larger due to new users enrolling to the system.

Furthermore, a parameter called ‘activation’ is required to be passed to the layer which serves as the activation function for the layer. An activation function allows models to take into account nonlinear relationships which is necessary in this case, since face recognition is a non linear problem. The activation function being used is ‘relu’ which stands for Rectified Linear Activation and although it is composed from two linear pieces, it has been proven to work well in neural networks. The second layer of the model which represents the network’s hidden layer follows a similar structure to the first layer. The last layer which plays the role of the output layer uses the activation function ‘softmax'. Softmax enacts the output sum up to 1 so that it can be interpreted as probabilities and the model makes its prediction based on which option has a higher probability [45]. Between the layers, ‘Dropout’ is being used. Dropout is a technique that prevents overfitting and provides a way of approximately combining exponentially many different neural network architectures efficiently. The term “dropout” refers to dropping out units (hidden and visible) in a neural network. Dropping a unit out results into a temporary removal of said unit from the network, along with all its incoming and outgoing connections. The choice of which units to drop is random. In this case, each unit is retained with a fixed probability independent of other units, where the probability is set at 0.5, which, according to researchers, seems to be close to optimal for a wide range of networks and tasks [46].

Following this, the model.compile() function is used which configures the model for training. Compiling the model takes two parameters which are optimiser and loss. The optimiser controls the learning rate by providing the algorithm that changes the weights and biases during training. The learning rate determines how fast the optimal weights for the model are calculated. The optimiser used for this model is the “SGD” (Stochastic gradient descent) optimiser which includes support for momentum and learning rate decay as well as Nesterov momentum. SGD performs a parameter update for each training example and label. ‘Categorical cross entropy’ was used for loss since it is commonly used when performing multi-class classification. It operates by comparing the predicted label to the true label and calculating the loss. A lower score indicates that the model is performing better. Last, a metric function is required for compilation in order to judge the performance of the model. A metric function is similar to a loss function, except that the results from evaluating a metric are not used when training the model. In order to make the produced output easier to interpret, ‘accuracy’ was used as the evaluation metric to show the accuracy score on the validation set at the end of each epoch. The coding procedure of assembling the MLP model is presented in Figure 10 below.

```
def mlp_model():
    model = Sequential()
    model.add(Dense(128, input_shape=(128,), activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(256, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(len(os.listdir("faces/")), activation='softmax'))
    sgd = optimizers.SGD(lr=1e-2)
    model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
    filepath = "mlp_model_keras2.h5"
    checkpoint1 = ModelCheckpoint(filepath, monitor='acc', verbose=1, save_best_only=True, mode='max')
    callbacks_list = [checkpoint1]
```

Figure 10 - MLP model structure and compilation

The following step is forwarding the training and testing sets for the feature vectors and user IDs into the compiled MLP model. The network trained through the method model.fit() which takes four main arguments. The first argument passed is the training sets of feature vectors and user IDs while the second, called ‘validation data’ is set to the corresponding testing sets. The next argument is the function

setting the epochs. An 'epoch' is the number of iterations over the entire training data. The more epochs are run, the more the model improves, up to a certain point after which, the model will stop improving during each epoch. In addition, the more epochs, the longer the model will take to run. The number of epochs is set to 200 in order to allow the learning algorithm to run until the error from the model is sufficiently minimised. The last argument passed is 'batch size' which defines the number of samples that will be propagated through the network. Generally, a larger batch-size results in faster training, but doesn't always converge fast. A smaller batch-size is slower in training but it can converge faster therefore, the batch size for this model is set to 50. Finally, the scores for the network are calculated through the method model.evaluate() which takes as arguments the testing sets for feature vectors and user IDs as well as verbose which specifies how the training process for each epoch will be shown and is set to 3. The method developed in order to train the MLP model is shown in Figure 11.

```
def train():
    with open("train_features", "rb") as f:
        train_images = np.array(pickle.load(f))
    with open("train_labels", "rb") as f:
        train_labels = np.array(pickle.load(f), dtype=np.int32)

    with open("test_features", "rb") as f:
        test_images = np.array(pickle.load(f))
    with open("test_labels", "rb") as f:
        test_labels = np.array(pickle.load(f), dtype=np.int32)

    train_labels = np_utils.to_categorical(train_labels)
    test_labels = np_utils.to_categorical(test_labels)
    model, callbacks_list = mlp_model()
    model.fit(train_images, train_labels, validation_data=(test_images, test_labels), epochs=200, batch_size=50,
              callbacks=callbacks_list)
    scores = model.evaluate(test_images, test_labels, verbose=3)
    print(scores)
```

Figure 11 - MLP model training implementation

5.6 Recognition

The next stage is to utilise the developed MLP model as a probability based classifier capable of identifying a user in real time. This is achieved through the following steps. Initially, a similar procedure to the one in the stage of enrolment is followed where the OpenCV method cv2.VideoCapture() is used again in order to open a new window providing live video capture. The pre-mentioned Dlib methods are utilised to perform face detection and face alignment to the input image from the frame, again drawing a rectangle around the detected face. The feature vectors of the presented face are then computed through the pre-trained Dlib model and stored in a numpy array in order to be compared to the feature vectors representing the enrolled users in the system's dataset. The trained MLP model is then loaded and used to return a prediction for the given test image through the .predict() function provided by Keras. The probability threshold in order for the system to allow the identification is set to 0.9 so that false positives are avoided. If the model returns a probability greater than 0.9, then the system is set to output the user's name and the corresponding face ID above the detected face in the video capture window. In order to access the user data, its connects to the database and executes the query "SELECT * FROM faces". If the probability is lower than the threshold, the system outputs the specified message "no matching faces". Part of the code developed to achieve this is demonstrated in Figure 12.

```
def __extract_face_info(self, img, img_rgb, face_names):
    faces = self.detector(img_rgb)
    x, y, w, h = 0, 0, 0, 0
    face_descriptor = None
    if len(faces) > 0:
        for face in faces:
            shape = self.shape_predictor(img, face)
            (x, y, w, h) = face_utils.rect_to_bb(face)
            cv2.rectangle(img, (x, y), (x + w, y + h), (255, 255, 0), 2)
            face_descriptor = self.face_rec.compute_face_descriptor(img_rgb, shape)
            face_descriptor = np.array([face_descriptor, ])
            probability, face_id = self.__recognize_face(face_descriptor)
            if probability > 0.9:
                cv2.putText(img, "FaceID #" + str(face_id), (x, y - 70), cv2.FONT_HERSHEY_PLAIN, 1.5, (0, 255, 0),
                            2)
                cv2.putText(img, 'Name - ' + face_names[face_id], (x, y - 40), cv2.FONT_HERSHEY_PLAIN, 1.5,
                            (0, 255, 0),
                            2)
                cv2.putText(img, "%s %.2f%%" % ('Probability', probability * 100), (x, y - 10),
                            cv2.FONT_HERSHEY_PLAIN,
                            1.5, (0, 255, 0), 2)
            else:
                cv2.putText(img, 'No matching faces', (x, y - 20), cv2.FONT_HERSHEY_PLAIN, 1.5, (0, 0, 255), 2)
```

Figure 12 - Part of the face identification routine

5.7 Menu

A menu is implemented in order for the user to be able to navigate easier around the system. Upon initiation, the system presents two options to the user: “Press 1 for Enrolment, Press 2 for Identification” and then expects the user’s input. The user will have to press either of the suggested keys in order to select the corresponding option. The first option is directed towards new users that have not yet been enrolled. When it is chosen, it runs the complete routine of procedures that has been desired above in order to create a record of the new user in the system and update the database with the user’s information so that the system will be able to identify the user in the future. When choosing this option, a file containing the Terms and Conditions of using the system is output to the user asking for permission to collect, access and use their personal data. The user can confirm that they accept the terms and conditions in order to proceed to the image acquisition routine by pressing enter. The second option directly runs the recognition routine and should be chosen in the case where the user is already enrolled and their images are in the system’s dataset along with their information on the database.

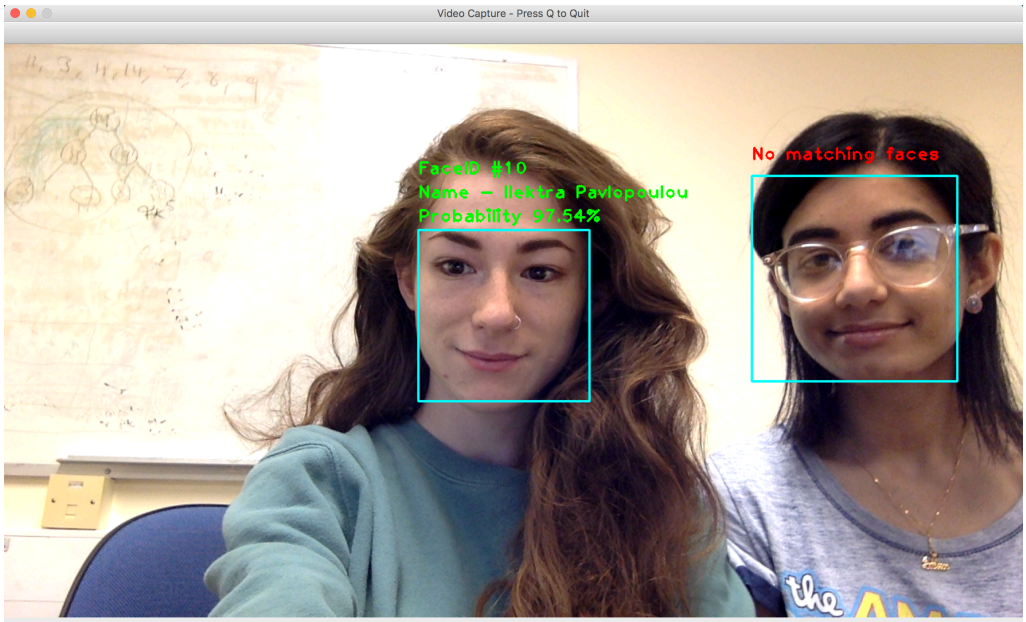

Figure 13 - System output demonstration

In Figure 13 above, a screenshot showing the results of the finished product is presented. As can be seen, the developed system could successfully connect to a camera in order to provide the live video stream through a window on the system's screen. Furthermore, the rectangles drawn around the faces in the frame prove that face detection is performed appropriately. The information that is output above the first rectangle demonstrates that face identification was successful for the enrolled user, whilst the user that was not enrolled was correctly classified as unknown through the output message "No matching faces". This, in combination with the matching probability that is provided as output for the identified user, verifies that the system's classifier operates appropriately. The system's connection to the database is also proven due to the fact that the face ID and the user's name are being correctly displayed on the screen. This means that the system managed to access the database and fetch the data stored in order to use them as output upon identification had taken place.

5.8 Summary

In conclusion, the implementation of the program effectively built upon the design created to deliver a product that utilises an ANN, in combination with a number of library functions in order to achieve face identification. This has helped to deliver a system that can manage to successfully communicate with database. Overall, the system provides an effective means of implementing a complete face identification system. The program implemented gathers information from the user during the enrolment process in order to be able to match this information and output it when the user has been identified, in combination with providing the means for the user to choose the option to either enrol or get identified through the menu.

6. TESTING: VERIFICATION & VALIDATION

In order to determine that the system built achieved to meet the desired requirements and criteria, testing was necessary to ensure that every component was working as expected.

6.1 Unit testing

Unit testing is a level of software testing in which individual components of the software are tested [47]. The purpose of this is to validate that each unit of software performs as designed. A unit is the smallest testable part of the software and usually has one or a few inputs and a single output. For the purposes of this project, unit testing comprises of testing individual functions in each of the program's classes in order to identify potential issues that could affect the behaviour of the system and impact its results. The functions were tested through the assistance of break points which were used to put traces in the code in order to validate the output of each individual function for every stage of the feature extraction, classifier training and recognition process. Furthermore, a number of print "test" methods were written in the python code for the methods that did not produce a visible output. This was conducted to confirm that each function was performing as intended. The unit tests implemented are detailed below in Table 1.

Table 1 - Unit Tests

Test	Description	Expected Outcome	Actual Outcome	Action Taken
User information	Pass the input user information to the system's data set	The name and ID the user passed to enrol should be saved in the system	The name and ID the user passed to enrol are successfully saved in the system	N/A
Database update	Automatically update the database upon user enrolment	The personal data of the user should be automatically added in the database	The database query did not add the user's data	The database query in the program was changed in order to allow the new data to be added to the database
Live video stream for image acquisition	Webcam connects to the system and provides live video stream	Window should open showing the live video stream	Window successfully opens and provides live video stream	N/A
Live face detection for image acquisition	When a face is detected in the live video stream, a rectangle is be drawn around it	Rectangle should appear in the live video stream window, surrounding the detected face	Rectangle successfully appears in the live video stream window, surrounding the detected face	N/A

Test	Description	Expected Outcome	Actual Outcome	Action Taken
Live face alignment for image acquisition	Each face image taken is aligned in live video stream	A second live video stream window should open, showing the aligned face	A second live video stream window successfully opens, showing face alignment in real time	N/A
Image storing	Each face image taken is stored in the data set	New face images should be stored in the data set	New face images are successfully stored in the data set	N/A
Exit window automatically	After a set number of images has been taken from the live video stream, automatically close the window	The window should automatically close after set number of face images has been taken	The window successfully closes after the required face images have been taken	N/A
Image loading	Load the new images from the data set	Each new image in the data set should be read	Each new image in the data set is correctly read	N/A
Data set face detection	Detect the face in each new image from the data set in order to normalise it	A face should be detected in each one of the new data set images	Successfully detects the face in each new data set image without presenting errors	N/A
Data set face normalisation	Compute 68 facial landmark on every image with a detected face to estimate the face's pose and normalise the face	The detected face in each image should be normalised	The detected face in each image is successfully normalised	N/A
Feature generation	Compute 128 measurements for every face	128 measurements should be generated for every face in the data set	128 measurements were successfully generated for each face in the data set	N/A
Feature extraction	Add the generated measurements to the specified CSV file	The measurements for each face should be stored in the CSV file	The measurements for each face were successfully stored in the CSV file	N/A
Load CSV file	Load the measurements from the CSV file in which they are stored	Should return the feature vectors stored in the CSV file along with the face IDs	Successfully loads the feature vectors from the CSV file along with the corresponding IDs	N/A
Label/feature separation	Separate the labels (face IDs) from the features (face measurements)	The labels and features should be separated from each other	The labels are successfully separated from the features	N/A

Test	Description	Expected Outcome	Actual Outcome	Action Taken
Data split features	Split the face measurements into training and testing data and store them in separate files	The face measurements should be split into two data sets for training and testing and stored into two corresponding files	The face measurements are correctly split into a training a testing set and stored in two corresponding files	N/A
Data split labels	Split the face measurements' corresponding labels into training and testing data and store them in separate files	The labels corresponding to each set of face measurements should be split into testing and training ones according to the measurements and be stored in two separate files	The labels are correctly split into training and testing sets corresponding to the face measurements' sets and are successfully stored in two separate files	N/A
MLP model creation	Set the layers for the MLP model and compile it	Take the layer information along with the loss, optimiser, metrics and compile the model	Model successfully compiled	N/A
Save MLP model	Save compiled model to the specified path	The compiled model should be saved in the system	The compiled model is successfully saved in the system	N/A
Load data into MLP model	Open the training and testing files for features and labels in the MLP model	Each of the training and testing sets of features and labels should be loaded in the model	Each of the training and testing sets of feature and labels loads successfully in the model	N/A
Train MLP model	Testing if the model created can train on the provided data	Model should train on the data without presenting errors	Model successfully trains on the data	N/A
Model output results	MLP model outputs evaluation	MLP model should output the scores and MLP error after training	The scores and MLP error of the model are outputted after the model trains	N/A
Live video stream for recognition	Webcam connects to the system and provides live video stream	Window opens showing the live video stream	Window successfully opens and provides live video stream	N/A

Test	Description	Expected Outcome	Actual Outcome	Action Taken
Live face detection for recognition	When a face is detected in the live video stream, a rectangle should be drawn around it	Rectangle should appear in the live video stream window, surrounding the detected face	Rectangle successfully appears in the live video stream window, surrounding the detected face	N/A
Input face alignment	Perform face alignment to the input image through the dlib model	The detected face in the input image should be aligned	The detected face in the input image is successfully aligned	N/A
Input feature generation	Compute 128 measurements for the input face image	128 measurements should be generated for the input face image	128 measurements were successfully generated for the input face image	N/A
MLP model prediction	Predict if the input face has a match in the data set using the MLP model as the classifier for feature matching	A prediction should be made based on the feature matching	A prediction is successfully made based on the feature matching	N/A
Data base connection	Connect to the database in order to fetch the usernames and user IDs	Database should connect to the system	Database did not connect to the system, Error: no table ‘faces’ found	Changed the path of the database file in the program to match the correct location of the folder on the computer
Recognition output (match)	When the input face image has a match, output the user’s information above the rectangle of the detected face	The matched face’s username and user ID should be outputted in the live video stream window	The matched face’s username and user ID are correctly outputted in the live video stream window	N/A
Recognition output (not a match)	When the input face does not have a match, output a message	“No matching faces” should be outputted above the rectangle of the detected face	“No matching faces” is correctly outputted above the rectangle of the detected face	N/A
Exit window	Pressing the key ‘q’ disconnects the camera providing live video stream	Window showing live video stream closes	Live video stream window closed successfully	N/A
Menu options	Running the program should output two options: “Proceed” and “Enrol”	User is presented with the two options	Options correctly outputted to the user	N/A

Test	Description	Expected Outcome	Actual Outcome	Action Taken
Option ‘Proceed’	Pressing the key ‘P’ redirects the user to the routine of face recognition	User should be transitioned to the first stage of the recognition routine	User is successfully transitioned to the first stage of the recognition routine	N/A
Option ‘Enrol’	Pressing the key ‘E’ redirects the user to a text file with the terms and conditions for using the system, asking the user to press enter if they accept	User should be transitioned to the text file	Text file successfully opens providing the correct information	N/A
Option ‘Accept’	Pressing the key enter runs the routine of image acquisition	User should be transitioned to the first stage of the image acquisition routine	User is successfully transitioned to the first step if the image acquisition routine	N/A

6.2 Integration Testing

Following the completion of all unit tests, integration tests were implemented in order to combine individual components of the system together and test these as a group [48]. This involved testing the interaction between integrated units, including the interaction between the program and the database including the user information. The most important part of the integration tests was to confirm that each of the main stages of the system was successfully producing the output and results that it was expected to achieve.

Table 2 - Integration Tests

Test	Description	Expected Outcome	Actual Outcome	Action Taken
Menu	Provides user with two options: proceeding to direct face recognition which skips stages and runs the last class or enrolling to the system which runs all the classes in order. In the case of the second option, loads the Terms and Conditions form before running the first class	Provide output: “Press 1 to Proceed or Press 2 to Enrol” and run appropriate class. In the case of the second option, display Terms and Conditions form and wait for user to press enter in order to run the first class	Output is correctly displayed and the appropriate classes are run depending on the option chosen Choosing the second option correctly outputs the Terms and Conditions form and waits for user to press enter before running the first class	N/A
Enrolment	User information is passed in the system and database is updated with the new information	User name and ID are passed in the system and added to the database	User data is correctly passed to the system and successfully added to the database	N/A
Image acquisition	Execute video capture, detect face in real time, align face in real time, take set number of images, save images in new folder, release video capture	Open live video stream window, draw rectangle around the face, open second window showing face alignment, capture set number of images, save images in the correct folder path, exit both live video stream windows	Window opens, rectangle is drawn, second window opens, windows close. Images of the new user are successfully captured, aligned and stored in the system	N/A

Test	Description	Expected Outcome	Actual Outcome	Action Taken
Storing features	Images are loaded from the dataset, normalisation is performed, face measurements are computed and stored in the system	128 measurements for each face in the dataset should be produced and stored in the provided CSV file	The measurements are successfully produced and stored in the correct file	N/A
Training/testing set formation	Face IDs are separated from the measurements, then the measurements and the IDs are split into corresponding training and testing sets and stored into four files	Four files should be created for training and testing measurements and the corresponding training and testing IDs	The files for each training ad testing set are successfully created in the system	N/A
MLP model training	Data is loaded to the network from the files, then model trains based on them and outputs scores	The training cycles should be output as the model trains, then MLP error and scores should be output after training	Training cycles, MLP error and scores were correctly output suggesting that the model trained successfully	N/A
Face recognition	Execute video capture, detect face in real time, align and normalise face, compute face measurements, perform feature matching, use MLP model for prediction, connect to database and fetch user's data	Open live video stream window, draw rectangle around the face, output MLP prediction results above detected face along with corresponding data from the database	Window successfully opens and rectangle is drawn around face. The model does not properly load resulting to the person detected not being identified due the probability being based on the classifier which in turn fails to activate the connection to the database which would fetch the user information	Changed the location and file path of the saved trained model in order to make it communicate with the recognition class and successfully work as the classifier, allowing the connection to the database to correctly fetch the user information in order for it to be outputted on the screen

6.3 System testing

After the integration tests verified that the main components of the system communicated successfully and produced the desired outputs, system testing was performed in order to find out how the system functioned as a whole and ensure that the produced final outputs matched the expected ones.

The system was initially tested on four subjects in order to determine if the finished product worked as expected. The testing involved enrolling the subjects in the system by adding their names to the database and their images to the data set along with a face ID, then having each subject stand in front of the camera connected to the system and monitoring the results that were output on the window providing the live video stream on the system's screen. The system was tested repeatedly on each of the subjects in order to ensure that the predictions leading to face recognition were based on the system classifier's proper functionality and not on chance. This test confirmed that the system would operate successfully every time an enrolled user wished to utilise it for identification. In order for the system to pass the accuracy test and meet the acceptance criteria, it needed to achieve face recognition at least 90% of the times it was tested. The system managed to recognise all of the enrolled subjects successfully each time it was tested.

In order to carry out the second test, the assistance of three new subjects was provided. This was due to the fact that the system needed to be tested against unknown users in order to inspect if it would fail to identify them. The results obtained from the system testing demonstrated that although the system successfully identified enrolled uses, it misclassified new users by outputting enrolled users' information instead of presenting the specified message informing the user that matching faces were not found. This lead to re-examination of the trained MLP model since it was strongly believed that the problem of misclassification occurred due to a factor linked to the model's training. The action that was taken in order to address this was changing certain parameters in the part of the code that was allocated for the ANN's training. The model was then retrained and the system was tested once more against the new subjects. This time, the system successfully classified the new users as unknown by correctly outputting the message "No matching faces". Since the parameters of the network had changed, the system was tested against the enrolled users as well in order to ensure that the identification of known users was still successful. This test confirmed that the final product was capable of identifying known users while failing to recognise users that were unknown to the system.

6.4 Summary

In conclusion, the testing phase of the program proved to be extremely useful since it assisted in the detection of a number of different issues that were occurring within the system. There were a few cases found while conducting the unit tests that highlighted problems concerning the connection and correct communication with the database which resulted in the data not being stored in the way that was expected. Due to this reason, the SQL queries used were revised in order to ensure that the database functionality works as expected.

Integration testing was another important aspect of the testing process, which brought to attention that once the main components were connected, the database request to fetch the contents of the table including the users' information was presenting the intended output. Furthermore, it was discovered that this problem was occurring due to the fact that the saved MLP model was not being loaded in the code allocated in the face recognition class. This issue was successfully fixed by changing the location in

which the trained model was being saved, as well as changing the file path of the model in the program where it was being used.

System testing was also conducted in order to allow the system to operate on the two different types of users that it could come across when utilised. The system was tested against known and unknown users, the former ones being enrolled in the database while the latter ones not being enrolled. This test was conducted in order to make sure that the face identification was properly performed on the correct users. This test indicated that there was an extremely important issue that needed to be addressed immediately. The issue was problematic classification of the users the system was tested against. Fortunately, this issue was successfully tackled and the final system test results demonstrated that the face recognition was only performed on enrolled users while the users that had not been introduced to the system through enrolment were not being identified.

7. DISCUSSION: CONTRIBUTION & REFLECTIONS

Following the completion of testing the system, the progression of the project can be reviewed. By comparing the results of testing to the Project Initiation Document (PID), the extent to which the initial objects, outputs and features of the application that were achieved can be analysed.

7.1 Objective Outputs & Solution Success

After the first system test was conducted, there was only one issue with the overall product. The classification error through predicting false positives was alarmingly high. The cause of this was thought to be linked to the parameters that were set for the neural network before training. The steps that were taken in order to attempt to fix the classification accuracy in the model are described below. Initially the dropout for the network layers was set to 0.8 wish was thought to be relatively high and could be the cause of under-fitting. Therefore the first attempt to improve the network's accuracy involved decreasing the dropout to 0.5 helped reduce the MLP error and improve the recognition. When attempting to increase the network's accuracy by increasing the epochs to 400, the results were unexpected. Although the network demonstrated a higher accuracy than before and presented a 100% probability when recognising an enrolled user, it very often misclassified unregistered users as users in the data set instead of declaring that matching faces were not found. This suggested the idea that the network was probably overtraining. In order to tackle this, the epochs were reduced to 200, something that resulted in the network dramatically reducing the false positives during the classification for face recognition while still maintaining a high accuracy.

One of the initial project objectives was for the developed system to be based on an ANN. In other words, an ANN had to be implemented as the classifier of the system in order to address the task of face identification. The second main goal of this project was for the system to operate in real time which meant that when a user would be presented to the system, it would be able to instantly identify them and output the correct user information or acknowledge that the user is unknown and output that there are no matching faces found in the database. It can be said that, upon the completion of the system's implementation, both of these objectives were satisfied and the set criteria for the system were met. The ANN classifier was successfully implemented through the utilisation of the Keras API which allowed to create a customised MLP model able to handle the classification problem of face recognition. Furthermore, the system achieved to produce real time results for both successful and unsuccessful identification.

7.2 Limitations

Despite being mentioned in the PID, PCA and eigenfaces were not implemented in the project for the task of feature extraction. The PCA eigenfaces approach appears to be a fast, simple and practical method and has become one of the most widely used face recognition techniques. However, it does not provide invariance over changes in poses and scales. It is very strict on the required pose and rotation of the face, even slight changes or differences could seriously affect the recognition outcome. The simplest invariance could not be captured by the PCA unless the training data explicitly provides this information. Furthermore, the covariance matrix is difficult to be evaluated in an accurate manner and the time complexity for the process is high. Therefore, this method is computationally expensive and can become very complex with an increase in data size [49]. It was decided that a less complicated and time consuming method should be used for this task in order to reduce the risk of failing to complete the

implementation of the ANN classifier which was intended to be the main feature of the system. The methods that were used instead were provided by the DLIB library and achieved very good results for face alignment and feature extraction while not requiring a significant amount of time to implement. This allowed for more time to be dedicated on the design and the implementation of the MLP model and since the completion of the project, it was regarded as a good choice since the model that was developed produced very satisfactory identification results accomplishing high accuracy.

Another significant limitation in the developed system is the lack of a graphical user interface (GUI). The addition of a GUI to the developed system would improve its functionality by making the software appear easier to use and therefore, more user friendly. Giving the system a front end implementation would be extremely beneficial for the current and future users due to the fact that it will no longer be necessary for them to get involved with the system's backend which at times can be confusing and time consuming. The GUI to be developed would simplify the process of running the program, therefore providing the system with the potential to be more vastly used. One last limitation of this project would be data encryption. Although the developed system guarantees the protection of the user data, encryption would be a useful feature for the system. The purpose of data encryption is to protect digital data confidentiality as the data is stored on computer systems and transmitted through the internet or other computer networks [50]. Encryption could be applied on the users' face images in the system's dataset as well as on the names stored in the database connected to the system. Implementing a data encryption technique would therefore ensure that all the data is stored securely in the system and make the system appear more trustworthy to the user.

In reflection, the process of developing this project has helped me to significantly strengthen my ability to create large and complex systems. Through the implementation of this system, my understanding of the face recognition process was majorly improved as well as my knowledge on the existing face recognition techniques. Whilst the system developed is by no means perfect, I consider that I have been able to create an effective yet simplified version of the face authentication systems that are commercially available in the market and used for serious purposes. My understanding of Python has vastly improved throughout the duration of this project, along with my ability to understand and work with new tools using documentation. Furthermore, my knowledge on the subject of face recognition has expanded, through the study of existing approaches and techniques as well as through the implementation of all the different stages that are included in the process of achieving face identification. Working on this project and having to meet deadlines enhanced my time management skills as I had to allocate sufficient amounts of time every week in order to be able to deliver the finished product on time. Moreover, throughout the duration of the project's development, I kept constant track of the code being written for the program and payed attention on how changes and details affected it. This helped me to developed a better understanding of how code is being used in a number of different scenarios which in turn assisted me greatly in using it for the implementation of my system.

8. SOCIAL, LEGAL, HEALTH & SAFETY AND ETHICAL ISSUES

A person's face is a fundamental and highly exposed element that defines someones identity. In general people recognise one another when they see a known face or a photo of that face. In recent years, there has been an increase in the registration, storage and dissemination of peoples' face images in both private and public sectors. Social networks applications, corporate databases, smart-city deployments, digital media, government applications, even restaurants and coffee shops are collecting face images of individuals. Coupling this with the advancements in Artificial Intelligence (AI), face recognition algorithms tend to become more accurate than humans in recognising ones' face. Today it is common understanding that our personal data is collected, stored and processed by governments and organisations. In the case of face images being misused, the information could result in potential social damage to the person concerned and without them been able to take immediate corrective actions due to the complexity of the organisations access layers involved. Since face images are fundamental to define an individual's identity, the database numbers storing face image data grow in numbers and sizes at the same time the risks of identity theft, unauthorised tracking and other misuse can be potentially damaging to an individual. Without being able to eliminate the risks of data hacking concerns on the violation of privacy rights and its social impact, there is an increasing threat to ones basic right of public anonymity as we know it. Due to this and many other reasons face images and their related biometric data that could lead in a person's identification are now considered as sensitive personal information.

The system developed collects a number of personal data such as a user's faces pictures and stores them in the data set images as well as their full names, which are stored in the database. Therefore to make sure that it follows the legal and ethical guidelines for data protection. Every time a new user wishes to enrol to the system a form of the terms and conditions is presented to them which the user is asked to accept (consent) in order for the system to collect their data. In order to ensure that the data is protected GDPR guidelines were followed. The General Data Protection Regulation (GDPR) is a regulation in EU becoming a law on data protection and privacy applicable to all government, legal entities and individuals within the European Union (EU) and the European Economic Area (EEA) [51]. As of May 2016 the data protection reform package entered into force and applicable regulation became effective since May of 2018. It is an enforceable Data Protection Directive for the police and criminal justice, generally known as General Data Protection Regulation. The aim of this regulation was to bring up to date provisions that guarantee privacy rights so that advances in technology, since the previous 1995 Data Protection Directive, do not use any loopholes to exploit use of private data directly or indirectly. The regulatory framework binds all involved parties EU governments, national data protection authorities, companies and citizens with the aim to introduce more transparency and allow people to gain more control over their personal data and the ways to access it. The focus of the regulatory framework is in providing means to reinforce individual's rights, fortify the EU internal market, control how transfers of EU citizens personal data are effected to third countries and create global data protection standards. One of the fundamental issues of the GDPR is addressing the issue of when an individual no longer wishes their data to be stored and/or processed. Provided there are no legitimate reasons for their data to be retained their data should be deleted. Assume that an individual has given consent to his personal data to be processed for a specific purpose and in the course of time they do not wish to use this service for which his personal data had to be stored and processed. There is no reason for their data to be further kept in the system supporting the service and thus would have to be deleted

upon notice that they do not require use of the service, or if the purpose of the service itself has ceased to exist.

As the system developed for the purposes of this project is a processor of personal data, it clearly and appropriately discloses any such data collection, declares the lawful basis and purpose for data processing and states the time period for which the data is to be retained and if it is being shared with any third parties or outside of the EU/EEA. Furthermore in order for the system to fortify its internal data security to comply to the GDPR, its internal mechanism of individuals pictorial data storage cares not to assign names but rather index numbers, used as id's, to the images that are retained in the dataset. Similarly the entries to the database that keeps the biometric data as a result of the training of the system are numbers corresponding to a name that are organised in a manner that their use is specific to the application thus their use could be restricted for this projects purposes only. Last the individuals that agreed to provide their personal data for the purpose of this projects' development and test were informed that upon project completion their personal data will be permanently deleted after a period of two months and before they are enrolled to the system a clear message informs them on the use of their personal data requiring their consent before they proceed. The permission request message is a shown below in Figure 14.

Terms & Conditions

By enrolling to this system you activate a mechanism that acquires a number of photographs that are stored and processed in order to extract biometric data which would be used to train the system to automatically recognise you in the future. You will also be asked for your name, which will be linked to this biometric data in order for the system to identify you when it sees you. This data is considered sensitive personal information and its acquisition and processing falls under the GDPR rules framework for data protection. These data belongs to you. It does not give us any right to process or share these data except for the purposes of proving the functionality of this system. Two months after system demonstration your image data and its derivatives including any links to your name will be permanently deleted.

We need your permission to do things like acquiring pictures of you, processing your biometric information and name for the purpose of this system to automatically recognise you and indicate your identity. You must give us your permission to do these things and this permission extends to that your data will be used only locally solely for the purposes of this project and not be shared in any way. By choosing **I Agree** *you enter the system or else you opt out.*

Figure 14 - Terms and Conditions form

The data collected for this project were ethically used for the purposes of training and testing the face authentication system. In the PID it was stated that in order to ensure that their privacy is accounted for, all photos used in this project will be deleted after a period of two months from project report submission. The persons of whom pictures will be acquired and processed by the system will be required to accept a terms and conditions document, detailing how their data will be used and expressing their consent for this use. The wording of the message will be clear and concise so that the person concerned will be able to fully understand to what they concerned and how their images and personal data is going to be used for the purposes of this project. Apart from this there is no further ethical issues as with regard to this project since personal data are not publicised or shared with any third parties nor any cloud based technology has been used for supporting data storage needs and all data remains locally stored until the purposes of this project are fulfilled.

No health and safety issues were raised in the PID and after the implementation of the project, this remains the case.

9. CONCLUSION & FUTURE IMPROVEMENTS

9.1 Conclusions

The main objective of this project was to develop a face identification system which achieves face recognition based on an artificial neural network. Due to the testing results, it has been proven that the system developed achieved this objective and met the acceptance criteria. The system provides high accuracy, by successfully identifying each of the enrolled users and is able to connect to a database containing the personal data for each enrolled user. It also achieved to automatically update the database every time a new user enrols to the system. Furthermore, the system managed to provide output upon classification in order to inform the user if the identification was successful or not. It also was extremely efficient, by providing the classification results in an execution time of milliseconds. Moreover, the ANN which was developed produced results that helped greatly to optimise the network before saving the final model for use. Finally, the ANN was successfully trained on the classification problem of face recognition and fulfilled its role as the system's classifier.

Throughout the development of this project, my horizons were broadened on and my knowledge was expanded on the subject of face recognition and the uses of ANNs as well as their creation and training. A better understanding was also developed for every method that was implemented in order to represent each stage of the process, such as face detection, face alignment, feature prediction and extraction.

9.1 Future Work

Future work for this project mainly involves the implementation of a graphical user interface (GUI). The addition of a GUI to the developed system would improve its functionality by making the software appear more user friendly. Giving the system a front end implementation would be extremely beneficial for the current and future users due to the fact that it will no longer be necessary for them to get involved with the system's backend which at times can be confusing and time consuming. The GUI to be developed would simplify the process of running the program, therefore providing the system with the potential to be more vastly used. Another improvement reserved for future implementation could be the addition of more face recognition techniques that could work in cooperation with the ANN in order to attempt to increase the system's accuracy. A hybrid neural network could be created which could be able to address more challenging face recognition tasks. An interesting technique that could be implemented is 3D face identification in order to protect the system against spoofing.

Furthermore, the process of running each class of the system manually could be automated in the future. For the purposes of demonstrating this project, only two processes would automatically run after the user chooses one of the two menu options. In the case of selecting "Identification", the system would automatically run the recognition class whilst, when choosing "Enrolment" the image acquisition class would run. After the images of the new user were stored in the system, each class of the program was run manually in order to clearly demonstrate the way in which each stage of the process operates. Following the demonstration, each of the processes that succeed image acquisition could be automated in order for the system to be more quick and efficient as it will require less handling from the user.

The system developed still requires a significant amount of further work in order to be truly competitive in the current market place or research field, however the system accomplishes to provide the main functionality of what a face identification system should achieve.

10. REFERENCES

[1] M. Hassaballah, S. Aly, Face recognition: challenges, achievements and future directions, IET Computer Vision, 2014 [Online]. Available at: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7172641 [Last Accessed April 1, 2019]

[2] P. J. Phillips, Support Vector Machines Applied to Face Recognition, 1999 [Online]. Available at: http://papers.nips.cc/paper/1609-support-vector-machines-applied-to-face-recognition.pdf [Last Accessed April 2, 2019]

[3] S. Z. Li and A. K. Jain, Handbook of Face Recognition, Springer, New York, NY, USA, 2004.

[4] W. Zhao, R. Chellappa, A. Rosenfeld, and P. Phillips, Face recognition: A literature survey. ACM Computing Surveys, pages 399–458, 2003.

[5] W. W. Bledsoe, The model method in facial recognition, Technical report pri 15, Panoramic Research, Inc., Palo Alto, California, 1964.

[6] W. W. Bledsoe, Man-machine facial recognition: Report on a large-scale experiment, Technical report pri 22, Panoramic Research, Inc., Palo Alto, California, 1966.

[7] W. W. Bledsoe, Some results on multicategory patten recognition, Journal of the Association for Computing Machinery, 13(2):304–316, 1966.

[8] W. W. Bledsoe, Semiautomatic facial recognition, Technical report sri project 6693, Stanford Research Institute, Menlo Park, California, 1968.

[9] W. W. Bledsoe and H. Chan, A man-machine facial recognition system-some preliminary results, Technical report pri 19a, Panoramic Research, Inc., Palo Alto, California, 1965.

[10] T. Kenade, Picture processing system by computer complex and recognition of human faces, PhD thesis, Kyoto University, November 1973.

[11] T. J. Stonham, Practical face recognition and verification with wizard. In H. D. Ellis, editor, Aspects of face processing. Kluwer Academic Publishers, 1986.

[12] L. Sirovich and M. Kirby, Low-dimensional procedure for the characterization of human faces. Journal of the Optical Society of America A - Optics, Image Science and Vision, 4(3):519–524, March 1987.

[13] M. Kirby and L. Sirovich, Application of the Karhunen - Loeve procedure for the characterization of human faces, IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(1):103–108, 1990.

[14] M. Turk and A. Pentland, Eigenfaces for recognition, Journal of Cognitive Neuroscience, 3(1):71–86, 1991.

[15] M. Turk, A random walk through eigenspace, IEICE Transactions on Information and Systems, E84-D(12):1586–1595, 200.

[16] J. D. West, A Brief History of Face Recognition, 2017 [Online]. Available at: https://www.facefirst.com/blog/brief-history-of-face-recognition-software/ [Last Accessed April 25, 2019]

[17] Kelly, M. D. Visual identification of people by computer. Tech. rep. AI-130, Stanford AI Project, Stanford, CA. 1970.

[18] A. K. Agrawal, Y. N. Singh, Evaluation of Face Recognition Methods in Unconstrained Environments [Online]. Available at: https://www.sciencedirect.com/science/article/pii/S1877050915006560 [Last Accessed April 13, 2019]

[19] S. Narmatha and K. Mahesh, Independent Component Analysis (ICA) Based Face Recognition System, International journal for advance research in engineering and technology, 3(7), 2015 [Online]. Available at: https://pdfs.semanticscholar.orgf228/2a846f12fb8ee1f297a1b39c659974142231.pdf [Last Accessed April 2, 2019]

[20] S. Elhabian and A. Farag, 2D Face Recognition Using PCA, ICA and LDA, University of Louisville [Online] Available at: http://www.sci.utah.edu/~shireen/pdfs/tutorials/Elhabian_2dFaceRecognition09.pdf [Last Accessed April 4, 2019]

[21] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, “Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection,” IEEE Trans. Pattern Analysis and Machine Intelligence, 19(7): 711–720, July 1997.

[22] L. Wiskott, J. M. Fellous, N. Krueuger and C. von der Malsburg, Face recognition by elastic bunch graph matching, IEEE Trans. on Pattern Analysis and Machine Intelligence, 19(7):775–779, 1997.

[23] V. N. Vapnik, Statistical learning theory, John Wiley & Sons, New York, 1998

[24] G. Guo, S.Z. Li, K. Chan, Face Recognition by Support Vector Machines, Proc. of the IEEE International Conference on Automatic Face and Gesture Recognition, pp. 196-201, 26-30 March 2000 [Online]. Available at: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=840634 [Last Accessed at April 6, 2019]

[25] C. Bishop, Pattern Recognition and Machine Learning, Springer, New York, NY, USA, 2006.

[26] W. S. McCulloch,W. Pitts, A logical calculus of the ideas immanent in nervous activity, 5(4): 115–133, 1943

[27] A. Abraham, “Artificial Neural Network”, Oklahoma Tate University, Stillwater, OK, USA

[28] B. Krose and P. van der Smagt, An introduction to Neural Networks, The University of Amsterdam, 1996

[29] J. M. Nazzal, I.M. El-Emary, S. A. Najim, Multilayer Perceptron Neural Network (MLPs) World Applied Sciences Journal 5 (5): 546-552, 2008

[30] K. Sobotta and I. Pitas, Face Localization and Facial Feature Extraction Based On Shape and Color Information, IEEE Int. Conf. on Image Processing, 3, 483-486, Switzerland, September 1996.

[31] R. Féraud, O. J.Bernier, J. E. Viallet, and M. Collobert, A Fast and Accurate Face Detector Based on Neural Networks, IEEE Trans. On Pattern Analysis and Machine Intelligence, 23(1): 42-53, 2001

[32] E. Arnaud, B. Fauvet, E. Memin, P. Bouthemy, A Robust And Automatic Face Tracker Dedicated To Broadcast Videos, IEEE International Conference On Image Processing, 2005 [Online]. Available at: https://ieeexplore.ieee.org/document/1530420?arnumber=1530420 [Last Accessed April 5, 2019]

[33] P. Brimblecombe, Face Detection using Neural Networks, University of Surrey, 2002.

[34] P. Viola and M. Jones, Rapid object detection using a boosted cascade of simple features, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001 [Online]. Available at: https://ieeexplore.ieee.org/document/990517 [Last Accessed April 4, 2019]

[35] https://stackoverflow.com/questions/37215036/dlib-vs-opencv-which-one-to-use-when [Online] [Last Accessed April 5, 2019]

[36] N. Dalal and B. Triggs, Histograms of oriented gradients for human detection, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005 [Online]. Available at: https://ieeexplore.ieee.org/document/1467360 [Last Accessed April 4, 2019]

[37] B.Shi, X. Bai, W. Liu, J. Wang, Face Alignment With Deep Regression, IEEE Transactions on Neural Networks and Learning Systems, 29(1), 2018 [Online]. Available at: https://ieeexplore.ieee.org/document/7728148 [Last Accessed April 6, 2019]

[38] H. Wang, J. Hu, W. Deng, Face Feature Extraction: A Complete Review, IEEE Access, Vol. 6, pp 6001 - 6039, 2017 [Online]. Available at: https://ieeexplore.ieee.org/abstract/document/8225635 [Last Accessed April 6, 2019]

[39] D. E. Rumelhart and J. L. McClellan, Parallel and Distributed Processing, vol. 1. Cambridge, MA: MIT Press, 1986.

[40] J. Zhang, Y. Yan and M. Lades, Face Recognition: Eigenface, Elastic Matching, and Neural Nets, Proceedings of the IEEE, 85(9): 1423 - 1435, 1997 [Online]. Available at: https://ieeexplore.ieee.org/abstract/document/628712 [Last Accessed April 7, 2019]

[41] J. E. C. Cruza1, E. H. Shiguemorib and L. N. F. Guimaraes, A comparison of Haar-like, LBP and HOG approaches to concrete and asphalt runway detection in high resolution imagery, Pan-American Association of Computational Interdisciplinary Sciences, 2015 [Online]. Available at: http://epacis.net/jcis/PDF_JCIS/JCIS11-art.0101.pdf [Last Accessed April 9, 2019]

[42] V. Kazemi, J. Sullivan, One millisecond face alignment with an ensemble of regression trees, IEEE Conference on Computer Vision and Pattern Recognition, 2014 [Online]. Available at: https://ieeexplore.ieee.org/document/6909637 [Last Accessed April 12, 2019]

[43] K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image Recognition, IEEE Conference on Computer Vision and Pattern Recognition, 2016 [Online]. Available at: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7780459 [Last Accessed April 14, 2019]

[44] https://keras.io/getting-started/sequential-model-guide/ [Online]. [Last Accessed April 15, 2019]

[45] E. Allibhai, Building A Deep Learning Model using Keras, 2018 [Online]. Available at: https://towardsdatascience.com/building-a-deep-learning-model-using-keras-1548ca149d37 [Last Accessed April 15, 2019]

[46] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Journal of Machine Learning Research 15, 2014

[47] A. Kolawa and D. Huizinga, Automated Defect Prevention: Best Practices in Software Management, Wiley-IEEE Computer Society Press. p. 75, 2007

[48] Software Testing Fundamentals., Integration Testing, 2019 [Online]. Available at: http://softwaretestingfundamentals.com/integration-testing/ [Last Accessed April 19, 2019]

[49] W. Chen, M. Joo Er, S. Wu, PCA and LDA in DCT domain, Pattern Recognition Letters 26(15): 2474-2482, November 2005 [Online]. Available at: https://www.sciencedirect.com/science/article/pii/S0167865505001522 [Last Accessed April 20, 2019]

[50] N. Lord, What Is Data Encryption? Definition, Best Practices & More, 2019 [Online]. Available at: https://digitalguardian.com/blog/what-data-encryption [Lsat Accessed April 21, 2019]

[51] Guide to the General Data Protection Regulation (GDPR) [Online]. Available at: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/711097/guide-to-the-general-data-protection-regulation-gdpr-1-0.pdf [Last Accessed April 22, 2019]

11. APPENDICIES

APPENDIX 1: PROJECT INITIATION DOCUMENT (PID)

Individual Project (CS3IP16)

Department of Computer Science

University of Reading

Project Initiation Document

PID Sign-Off

Student No.	**25019583**
Student Name	**Ilektra Pavlopoulou**
Email	**i.pavlopoulou@student.reading.ac.uk**
Degree programme (BSc CS/BSc IT)	**BSc CS**
Supervisor Name	**Hong Wei**
Supervisor Signature	
Date	**01/10/2018**

SECTION 1 – General Information

Project Identification

1.1	**Project ID** (as in handbook)
	120
1.2	**Project Title**
	Computational Vision
1.3	**Briefly describe the main purpose of the project in no more than 25 words**
	A Face Authentication System capable of identifying a person from a digital image

Student Identification

1.4	**Student Name(s), Course, Email address(s)** e.g. Anne Other, BSc CS, a.other@student.reading.ac.uk
	Ilektra Pavlopoulou, BSc CS, i.pavlopoulou@student.reading.ac.uk

Supervisor Identification

1.5	**Primary Supervisor Name, Email address** e.g. Prof Anne Other, a.other@reading.ac.uk
	Hong Wei, h.wei@reading.ac.uk
1.6	**Secondary Supervisor Name, Email address** Only fill in this section if a secondary supervisor has been assigned to your project

Company Partner (only complete if there is a company involved)

1.7	**Company Name**
1.8	**Company Address**
1.9	**Name, email and phone number of Company Supervisor or Primary Contact**

SECTION 2 – Project Description

2.1	**Summarise the background research for the project in about 400 words. You must include references in this section but don't count them in the word count.**
	In order to create a face authentication system it is necessary to choose a face recognition algorithm that is able to identify the facial features from an image and search for images with matching features. The face recognition algorithm that I decided to implement is Principal Component Analysis using Eigenfaces and the research I did for that consisted of the mathematics behind eigenvalues and eigenvectors and matrices as well as the statistics in order to figure out how eigen faces work. This algorithm will have to train on a data set of images in order for the system to learn and become able to identify features and make matches. This led to the realisation that an artificial neural network(ANN) will be required for this project as "learning" can be achieved through the way it operates. ANNs consist of input and output layers, as well as (in most cases) a hidden layer consisting of units that transform the input into something that the output layer can use which makes them excellent tools for finding patterns that are far too complex or numerous. I already had some knowledge on this field due to the neural networks module I took last year, but further research was necessary in order to get some ideas on how to create a software more suitable for this project. Once the system has trained on the data set that was provided, it will need to take input from a camera that is connected to it so that it can attempt to perform the recognition process. Therefore, research on how to create software that uses the input of a live camera feed was needed as well as how to link the image from the camera against the authentication process. Furthermore, an algorithm for face detection was necessary after the camera takes a picture of the person in order for the program to be able to proceed and perform the face recognition. References: Matthew Turk, Alex Pentland - Eigenfaces for Recognition http://www.face-rec.org/algorithms/pca/jcn.pdf A Multi-Algorithmic Face Recognition System https://ieeexplore.ieee.org/document/4289908 Satya Mallick - Principal Component Analysis & Eigenfaces https://www.learnopencv.com/eigenface-using-opencv-c-python/ Luke Dormehl - Artificial Neural Networks https://www.digitaltrends.com/cool-tech/what-is-an-artificial-neural-network/
2.2	**Summarise the project objectives and outputs in about 400 words.** These objectives and outputs should appear as tasks, milestones and deliverables in your project plan. In general, an objective is something you can do and an output is something you produce – one leads to the other.

	The main objective of this project is to develop a facial recognition system capable of identifying a person from a digital image. In order for this to be achieved a number of different tasks must be completed. Initially a dataset needs to be acquired in order to be used for training the ANN (Artificial Neural Network) for the project. This will be the first task of the project. The output of this will be acquiring a dataset and pre-processing the data (this includes aligning, resizing the images into a uniform fashion). Following this the second task will be to create and calculate EigenFaces. This will involve creating a data matrix, calculating principal components and then reshaping Eigenvector in order to obtain the EigenFaces. The output of this task will be having created algorithm for recognising faces. At this stage test cases will be created in order to test the validity of the data and perform some analysis as to how effective the recognition system is and how this can be improved. After this has been developed, the third task will be to create some software which utilises a camera as input. This software will be produced by creating the following sub-tasks: - Develop face recognition algirithm - Developing camera input system - Developing face alignment - Recognising face alignment in the screen - Taking a photo - Outputting this image Once this task has been completed the output of this task will have been developing a system which individually can take a photo and align this photo correctly on the screen. Test cases will be produced in conjunction with this in order to verify the quality of the software created. The fifth task of the project will be to develop a working GUI in order for the user to be able to interact with the camera, and undergo a facial recognition process. Use cases, User storyboard and wireframe models will be created during this task in order to plan the interface effectively. Implementing the interface will also include building a request handler in order to ensure that the previously created algorithm and camera input system are able to communicate. The output of this task will be having created a proto-type of the final system. The final task of this project will be completing the project report, which will be developed alongside all of the previous tasks and finished after the fifth task. This will have the output of producing a final report, a project demonstration and a poster.
2.3	**Initial project specification - list key features and functions of your finished project.** Remember that a specification should not usually propose the solution. For example, your project may require open source datasets so add that to the specification but don't state how that data-link will be achieved – that comes later.

	The key features and functions of the finished project are as follows:
	
	- Camera input system
	  - Face alignment
	  - Recognising face in centre of screen (face detection)
	  - Taking a photo
	  - Output this image for comparison
	- GUI for the user to interact with
	- Face recognition algorithm developed using EigenFaces and an ANN trained on a dataset

2.4	**Describe the social, legal and ethical issues that apply to your project. Does your project require ethical approval? (If your project requires a questionnaire/interview for conducting research and/or collecting data, you will need to apply for an ethical approval)**
	There are social, legal, and ethical issues that apply to this project. For instance, this includes dealing with rules and regulations introduced in 2018 with the GDPR Act. In order to test the system at the end of the project, I will need to take pictures of people's faces and test the validity of the facial recognition system. In order to ensure that their privacy is account for, all photos used in this project will be deleted after a period of 2 months and the user will be required to sign a terms and conditions document, detailing how their data will be used and expressing their consent for images to be used for the purposes of this project.
2.5	**Identify and lists the items you expect to need to purchase for your project. Specify the cost (include VAT and shipping if known) of each item as well as the supplier.** e.g. item 1 name, supplier, cost

2.6	**State whether you need access to specific resources within the department or the University e.g. special devices and workshop**

SECTION 3 – Project Plan

3.1	**Project Plan** Split your project work into sections/categories/phases and add tasks for each of these sections. It is likely that the high-level objectives you identified in section 2.2 become sections here. The outputs from section 2.2 should appear in the Outputs column here. Remember to include tasks for your project presentation, project demos, producing your poster, and writing up your report.

Task No.	Task description	Effort (weeks)	Outputs
1	**Background Research**	3	
1.1	Dataset Research	1	Sources
1.2	Algorithm Research	1	Sources
1.3	GUI & Component Integration Research	1	Sources
2	**Analysis and design**	7	
2.1	System design	4	Use cases, User Storyboard
2.2	System analysis	3	Analysis report
3	**Develop prototype**	12	
3.1	Algorithm prototype	4	Prototype
3.2	GUI prototype	4	Prototype
3.3	Camera system prototype	4	Prototype
4	**Testing, evaluation/validation**	3	
4.1	Unit testing	1	Test cases
4.2	Integration testing	1	Test cases
4.3	Alpha testing	1	Test cases
5	**Assessments**	4	
5.1	Project report	2	Project Report
5.2	Poster	1	Poster
5.3	Demo	1	Demonstration
TOTAL	**Sum of total effort in weeks**	**29**	

SECTION 4 - Time Plan for the proposed Project work

For each task identified in 3.1, please *shade* the weeks when you'll be working on that task. You should also mark target milestones, outputs and key decision points. To shade a cell in MS Word, move the mouse to the top left of cell until the curser becomes an arrow pointing up, left click to select the cell and then right click and select 'borders and shading'. Under the shading tab pick an appropriate grey colour and click ok.

	START DATE: 01/10/2018 **Project Weeks**												
Project stage	0-3	3-6	6-9	9-12	12-15	15-18	18-21	21-24	24-27	27-30	30-33	33-36	36-39
1 Background Research													
Dataset Research													
Algorithm Research													
GUI & Component Integration Research													
2 Analysis/Design													
System design													
System analysis													
3 Develop prototype.													
Algorithm prototype													
GUI prototype													
Camera system prototype													
4 Testing, evaluation/ validation													
Unit testing													
Integration testing													
Alpha Testing													
5 Assessments													
Project report													
Poster													
Demo													

RISK ASSESSMENT FORM

Assessment Reference No.		**Area or activity assessed:**	**Development**
Assessment date	**01/10/2018**		
Persons who may be affected by the activity (i.e. are at risk)	**Ilektra Pavlopoulou**		

SECTION 1: Identify Hazards - *Consider the activity or work area and identify if any of the hazards listed below are significant (tick the boxes that apply).*

	Fall of person (from work at height)			Lighting levels			Use of portable tools / equipment			Vehicles / driving at work			Hazardous fumes, chemicals, dust			Occupational stress	✓
	Fall of objects			Heating & ventilation			Fixed machinery or lifting equipment			Outdoor work / extreme weather			Hazardous biological agent			Violence to staff / verbal assault	
	Slips, Trips & Housekeeping			Layout , storage, space, obstructions			Pressure vessels			Fieldtrips / field work			Confined space / asphyxiation risk			Work with animals	
	Manual handling operations			Welfare facilities			Noise or Vibration			Radiation sources			Condition of Buildings & glazing			Lone working / work out of hours	✓
	Display screen equipment	✓		Electrical Equipment	✓		Fire hazards & flammable material			Work with lasers			Food preparation			Other(s) - specify	

SECTION 2: Risk Controls - *For each hazard identified in Section 1, complete Section 2.*

Hazard No.	Hazard Description	Existing controls to reduce risk	**Risk Level** (tick one)			Further action needed to reduce risks *(provide timescales and initials of person responsible)*
			High	Med	Low	
5	Display screen equipment	Limit time spent looking at the screen			✓	
10	Electrical equipment	Use safe equipment and plugs			✓	
26	Occupational stress	Taking breaks to relax	✓			
29	Lone working / work out of hours	Ensuring permission is obtained for lone working/long hours from supervisor. Informing supervisor of health risks or concerns		✓		
Name of Assessor(s)			**SIGNED**			
Review date						

Health and Safety Risk Assessments – continuation sheet

Assessment Reference No	
Continuation sheet number:	

SECTION 2 continued: Risk Controls

Hazard No.	Hazard Description	Existing controls to reduce risk	**Risk Level** (tick one)			Further action needed to reduce risks *(provide timescales and initials of person responsible for action)*
			High	Med	Low	
Name of Assessor(s)			**SIGNED**			
Review date						

APPENDIX 2: LOGBOOK

CS3IP16

Individual Project

Logbook

Date	Notes	References	Actions
1st - 5th October 2018	Began background research on existing face recognition algorithms	M. Turk and A. Pentland, Eigenfaces for recognition, Journal of Cognitive Neurosicence, 3(1):71–86, 1991. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5432754	- Compared the advantages and disadvantages of the PCA and eigenfaces method
8th - 12th October 2018	Research on Artificial Neural Networks and how they can be utilised for face recognition	https://ieeexplore.ieee.org/abstract/document/501715 https://ieeexplore.ieee.org/abstract/document/661121	- Compared the advantages and disadvantages of implementing an ANN without eigenfaces based on existing systems
15th - 19th October 2018	Programming language evaluation and selection	https://www.learnopencv.com/opencv-c-vs-python-vs-matlab-for-computer-vision/	- Adopted Python as the programming language in which the project would be developed

Date	Notes	References	Actions
22nd - 26th October 2018	Research and evaluation of real time face detection methods	https://www.bogotobogo.com/python/OpenCV_Python/python_opencv3_Image_Object_Detection_Face_Detection_Haar_Cascade_Classifiers.php https://stackoverflow.com/questions/4440283/how-to-choose-the-cascade-file-for-face-detection https://stackoverflow.com/questions/41341409/where-is-cv-haar-scale-image-in-opencv-3-1-0-with-python-3-5/41343094	- Connected camera to the system that provided live video stream - Implemented live face detection for the system with rectangle being drawn around the face
29th October - 2nd November 2018	Research on available database applications to be connected to the system and store user data	https://sqlitebrowser.org https://www.mysql.com/why-mysql/ https://www.postgresql.org/about/	- Installed the DB Browser for SQLite in order to visualise the data to be stored in the system
5th - 9th November 2018	Research into user permissions for data collection. Consideration of social legal and ethical concerns of storing and accessing personal data	https://www.gov.uk/government/publications/guide-to-the-general-data-protection-regulation	- Composed of a Terms and Conditions form.
12th - 16th November 2018	Implementation of image acquisition process	https://stackoverflow.com/questions/34588464/python-how-to-capture-image-from-webcam-on-click-using-opencv	- Created code that extracted the face image form the live video stream frame and stored it in the system
19th - 23rd November 2018	Research into face alignment and image normalisation	https://ieeexplore.ieee.org/document/6909637 http://dlib.net/face_alignment.py.html	- Began to implement face alignment with Dlib's pre-trained model for face landmark localisation

Date	Notes	References	Actions
26th - 30th November 2018	Incorporation of live face alignment in the system as part of image acquisition	https://www.pyimagesearch.com/2017/05/22/face-alignment-with-opencv-and-python/	- Created second live video stream window showing face alignment being performed on the input provided by face detection - Adjusted system to store the aligned image
2nd - 7th December 2018	Dataset Improvement		- Sorted file loading and file writing in order for the input data to be stored properly in the system - Set system to take a specific number of images fro the user in order to create a consistent dataset
10th - 14th December 2018	Research on feature extraction algorithms	https://scikit-learn.org/stable/modules/feature_extraction.html	- Adopted Dlib's pertained model for feature vector generation
14th - 18th January 2019	Feature extraction and formation of new dataset	http://jphandson.blogspot.com/2018/12/face-recognition-using-python-dlib-and.html http://dlib.net/face_recognition.py.html	- Created a CSV file in order to store the extracted feature vectors from the Dlib model - Formed a training and testing set of data by dividing the contents of the CSV file in order to use them as input for the ANN to be developed
21st - 25th January 2019	Re-evaluation of the selected face detection method	http://dlib.net/face_detector.py.html https://towardsdatascience.com/cnn-based-face-detector-from-dlib-c3696195e01c https://stackoverflow.com/questions/37215036/dlib-vs-opencv-which-one-to-use-when	- Re-evaluated the Haar Cascade method provided by open CV, due to a high number of false positives - Replaced the previous method with a Dlib model for face detection that provided higher accuracy
28th January - 1st February 2019	Research into implementation of ANNs	https://www.tensorflow.org/guide/ https://keras.io	- Chose TensorFlow for the development of the ANN classifier sue to the wide range of tools they provided

Date	Notes	References	Actions
4th - 8th February 2019	TensorFlow development complications		- Abandoned the implementation of the ANN in TensorFlow due to difficulty to build the network given the library being low-level
11th - 15th February 2019	Redesign of the ANN classifier	https://keras.io/getting-started/sequential-model-guide/ https://towardsdatascience.com/building-a-deep-learning-model-using-keras-1548ca149d37 https://stackoverflow.com/questions/44747343/keras-input-explanation-input-shape-units-batch-size-dim-etc	- Started implementing a new ANN using Keras on TensorFlow backend since it provided more high-level tools
18th - 22nd February 2019	Completion of ANN implementation		- Completed the implementation of the MLP model in Keras - Successfully attempted to run the class on the dataset (network seemed to train)
25th February - 1st March 2019	Face recognition class creation		- Connected webcam to the system for live video stream - Implemented the methods used in the image acquisition class for face detection, face alignment and feature extraction
4th - 8th March 2019	Database connection	http://www.sqlitetutorial.net/sqlite-insert/ http://www.sqlitetutorial.net/sqlite-update/ http://www.sqlitetutorial.net/sqlite-select/	- Connected the system to SQLite for image acquisition and face recognition - Wrote SQL queries for database automatic update after user enters name before image acquisition and for data selection in order to output the correct name after face recognition

Date	Notes	References	Actions
11th - 15th March 2019	System testing, evaluation and adjustment	http://jmlr.org/papers/volume15/srivastava14a.old/srivastava14a.pdf https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9	- Tested the developed ANN classifier on a number of subjects in order to evaluate its accuracy - Adjusted certain parameters of the ANN in order to address misclassification of unknown users
18th - 22nd March 2019	Menu addition		- Created a menu to be shown when the program is run that offers the option of Enrolment of Identification where the user can select through providing input - Added Terms & Conditions form for the case of Enrolment - Automated each process to run after it has been selected
25th - 29th March 2019	Report write up		- Tested the application thoroughly - Practice demonstration