# Literature Review

## Facial recognition in the software industry

In this chapter, the Author describes how several well-known software companies addressed the problem of facial recognition software and how it functions.

**Face recognition on Facebook is handled by DeepFace.**

A Facebook research team created the deep learning facial recognition algorithm known as DeepFace. In digital photographs, it can identify human faces. The program employs a nine-layer neural network trained using four million photos uploaded on Facebook and has around 120 million connection weights (DeepFace, 2022). With 97% accuracy, DeepFace algorithms can recognise faces nearly as well as a person in an identical situation (DeepFace, 2022). The significant difference between human and machine face recognition is that a person can only maintain a few hundred or thousand faces in their memory. Therefore they can only recall a limited amount of faces. DeppFace, on the other hand, identifies all 1.4 billion users, and this number is continually expanding. The DeepFace system comprises four parts: a 2D alignment module, a 3D alignment module, a frontalization module, and a neural network module. When a picture of a face is sent through them in order, a 4096-dimensional feature vector is made that represents the Face. After that, the feature vector can be used for a wide range of tasks. For instance, a face may be recognised by comparing its feature vector to a list of existing faces and determining which Face has the highest resemblance to the other faces' feature vectors (DeepFace, 2022). The fact that the DeepFace facial recognition software learns from each new photo it recognises and adds what it discovers to the database in order to improve a person's facial profile is one of the program's most advantageous features.

**How does Apple Face ID work, and what is it?**

Before, a simple number password protected Apple products, but starting with the iPhone 5S in 2013, they were getting read off in favour of fingerprint readers in new phones. After having fun with fingerprint readers, Apple decided that if they were going to take unlocking to a new level, they needed to make it safer and faster. In 2017, they released the iPhone X with a unique face recognition feature called Face ID. Face ID creates a comprehensive 3D map of the user's Face using a "TrueDepth camera system," comprising an array of sensors, cameras, and a dot projector housed in the notch at the top of the phone display (Pocket-lint, 2018). The technology does a safe authentication check when you peek at your phone, and if it recognises you, you may unlock your phone or quickly and easily authorise a payment (Pocket-lint, 2018). The TrueDepth camera system, neural networks, and

Bionic chips are just a few of the hardware elements that Face ID depends on. Face ID can adapt to changes in a person's appearance, such as the application of cosmetic makeup or the development of facial hair. Suppose there is a more significant change in present appearances, such as shaving one's beard. In that case, Face ID will first verify a user's identification using the user password before updating the user's face data (Pocket-lint, 2018).

**How does the Google Vision API operate?**

One of Google's most intriguing and least-used capabilities is image-based search. The typical method for finding photographs on the Internet is to enter a phrase (keyword) of interest into the Google search bar, then choose the "Images" tab to see the results. In an image-based search, the image itself is the "keyword" or starting point, and the search results are images. Since this is how it works, this function is often called "reverse" search. Google's system for recognising images looks at pictures on the Internet from many angles. Google Vision API can determine a picture based on its shapes, colours, backgrounds, and locations. It can tell if the image is of a person, an animal, an object, or a product.

With the help of pre-trained models on big-picture datasets, the Google Cloud Vision API employs machine learning to recognise photos. It divides the images into hundreds of categories to identify items, locations, and faces. The findings are then produced with a confidence value (Vision?, n.d.).

## On-Site Software solutions in the Industry

Of course, in addition to cloud-based methods, on-premises solutions provide facial recognition by default. However, they have much more modest capabilities than their cloud counterparts and are usually not controlled by artificial intelligence. They are mostly not standalone software; they are frameworks implemented in different languages, so their use requires a certain programming skill. However, there are examples of such a framework containing an autonomous package. Among them is commercial software that requires a one-time payment, but open-source solutions that can be used for free are becoming increasingly popular. In this section, the two most common ways will be presented.

**OpenCV**

OpenCV is a BSD-licensed open-source, cross-platform package that is primarily a real transaction

for real-time image processing and is today the most popular solution in the field of image analysis, especially since it also has native Nvidia CUDA support. Originally It was written in the C language by Intel engineers. Still, it is also available in C++ and Python, and it has been extended with more language support using shell classes. Different wrap (Goyal, et al., 2017). In the field of face recognition, for detecting landmarks (Facemark API) and is suitable for solving related problems (e.g. finding faces, drawing landmarks), but it also has many other functions ( e.g. use own Face and Fischer face). The solution uses a Haar-Cascade classifier for face recognition, a machine learning-based method that uses cascading layers (Viola & M.Jones, 2003).

**Dlib**

Dlib is a low-level cross-platform open-source software written primarily in C++ but also supported by Python, part of the general artificial intelligence-related libraries that also have a set of tools for face recognition (King, 2009). Among other things, it provides the ability to search for face highlights and is also suitable for different implementations (such as face clustering). Despite the free Boost software license, many documentation and sample libraries provide step-by-step guidance from translating to performing development tasks. The face recognition module in the package contains a solution that only supports face-to-face recognition based on the so-called HOG (histogram of oriented gradients) method, which uses a distribution of image elements to identify different classes. More recently, methods based on neural networks are also part of the package.

We may conclude from the above example that facial recognition systems have enormous potential commercially and security-wise and how these systems have grown to be incorporated into our daily lives, whether shopping, online banking, or finding someone in the picture. We can use cloud base systems, on-site open-source, or commercial frameworks to develop Face recognition systems.

# Face Recognition techniques

This subject has a broad reach and extensive literature, and an entire series of expert publications deal with its particular understanding. Therefore, it is not surprising that although employing the same set of tools, the problem hides a variety of specialities: Face recognition is the study of figuring out the formal and shape requirements for recognising a person's Face and separating it from the background (Khan, 2018). Face-based identification, a subset of this, employs the former's advances to determine how similar or unlike two faces are and if they can both belong to the same person. Although the separation of these two notions is not strictly scientific in nature, the literature utilises them interchangeably to some degree, and they will be made separate throughout the remainder of the thesis.

## Face detection systems may be categorised into four classes based on their work**.**

* Knowledge-based methods
* Feature-invariant method
* Template matching method
* Appearance-based method

**Knowledge-based methods**

These approaches are based on shared human perceptions of a face. For example, in a photograph of a face, the minor intensities will be around the eyes, while the biggest would be near the nose. They attempt to abstract our understanding of faces into a set of rules. Some basic laws are straightforward to deduce. A face, for instance, typically features two symmetrical eyes, and the region around the eyes is darker than the cheeks. The space between the eyes or the contrast in colour intensity between the lower zone and the eye area is a facial characteristic (Solanki, 2016).

**Feature-invariant method**

Such techniques involve first identifying the distinguishing features of the Face (eyes, mouth, and nose), after which the Face is recognised by grouping these features. Characteristics of a face should remain the same regardless of the angle or location, according to feature-invariant techniques. The mouth, nose, eyes, cheekbones, chin, lips, forehead, ears, upper boundaries of the eye sockets, areas around the jawline, the sides of the mouth, and the positioning of the nose and eyes are some of the identifying features of the Face that are used in facial identification. The nose's size, the jaw's angle, and the space between the eyes (Solanki, 2016).

**Template matching method**

These methods are based on looking for a pre-made facial pattern in a photograph. This may be an edged template. The problem with such techniques is that the template we intend to utilise must be extremely representative because different ethnic groups have distinct traits. Furthermore, the pre-processing must be very good, as such systems are sensitive to lighting conditions. Moreover, these algorithms are sluggish and cannot function in real time since you must experiment with multiple orientations and scalings across the image.

**Appearance-based methods**

Based on a collection of pre-made teaching samples, a "model" is built, and portions matching this model are sought after in the image. Most of the time, appearance-based algorithms use statistical analysis and machine learning to figure out what's important in a photo of a face (Solanki, 2016).

* Eigenface based method – PCA Algorithm:

The Eigenfaces approach involves identifying a face's usual features and expressing those features as a linear combination of the so-called "eigenfaces" discovered throughout the feature extraction procedure (Çarıkçı, 2012). The faces in the training set's main components are computed. To perform recognition, the Face is projected into the area created by the eigenfaces. A comparison is made using the Euclidian distance between the eigenvectors of the eigenfaces and the eigenface of the picture in question. The individual has recognised if the distance is narrow enough. If the distance is too great, the image is treated as belonging to an individual for whom the system must be taught (Çarıkçı, 2012).

* Distribution-based Methods – LDA Algorithm:

Another method of dimensionality reduction is Fisher's Discriminant Analysis or LDA. Since LDA optimises the between-class scattering matrix measure while minimising the within-class scatter matrix measure, it is an example of a class-specific approach. It is more trustworthy for classification (Solanki, 2016). For both face identification and verification, LDA-based algorithms beat PCA. Fisher's faces are one of the most extensively utilised methods for face recognition. It is dependent on the way of appearance. Fisher invented linear/fisher discriminant analysis for face identification in 1930, which yielded successful results in the face recognition process (Solanki, 2016).

# Facial recognition technology in use

Artificial neural networks are one of the most widely utilised machine learning methods. Let's discuss computer vision, language processing, and self-driving cars to illustrate how they can tackle nearly any issue. It is frequently used, for instance, to analyse deep learning, complicated relationships, or interactions with plenty of data. When we hear the words "neural network," neurons in the brain are the first thing that comes to mind. The brain's ability to make decisions comes from these units and the networks they form. One of the most significant differences between a person and a robot or computer program is that a person's brain can recognise, evaluate, and learn from what it sees.

## Artificial Neural Network

The neurons inspire the names of artificial neural networks in the brains of more advanced biological creatures (Beresford & Agatonovic-Kustrin, 2000). Through its synapses, a biological neuron may receive and send electrical messages to other neurons, allowing impulse transmission (Jain, et al., 1996). Neurons are all connected in a much more complex and dense way than phone networks. Each neuron is linked between 103 - 104 with other neurons; the human brain has 1014 to 1015. (Jain, et al., 1996). No matter how big they are overall, neurons transfer information in smaller chunks, which is an essential characteristic from the perspective of ANNs. This led scientists to conclude that neurons do not directly convey information but via their interactions (Jain, et al., 1996). The development of ANNs was based on these discoveries. The initial neurons' mathematical models are simple: for a given number of weighted input signals, one is returned if the sum of the inputs exceeds a specified u value; otherwise, 0 is returned. Jain et al. summarised the action of early neurons as follows: where is a unit jump function in 0, xj is the jth input, wj is the synaptic weight associated with it, and u is the limit to be reached (Jain, et al., 1996).

In practice, the threshold value u is often written as w0 and is called a constant weight of value x0 (McCulloch & Pitts, 2014). He also says that the different weights mean different things. From the point of view of neuron activity, a positive weight makes the synapse more active, while a negative weight makes it less active. In today's world, the number of inputs in the original may be very different from the amount in the model (but none of its element numbers can be 0) (Kristof, 2002).

|  |
| --- |
| A picture containing text, clock  Description automatically generated  Figure . A multi-layer neural network's theoretical structure (GeeksforGeeks, 2021). |

It has been shown that this model can do general-purpose computation (Jain, et al., 1996)s, but since it is built on a number of simplifications that do not correspond to how actual neurons work, it has undergone multiple generalisations to reach its present state. Alternative activation functions are used in place of the threshold function. These processes and the threshold function determine how the given neuron will react to the input. One of the most widely utilised functions in use today is the sigmoid activation function (Shanu, n.d.), which has the following formula:

Where a represents the activation and b determines the curve's shape. One of the essential things about neural networks is that they can learn because they are made up of many layers, and neurons can talk to each other across the layers. Generally speaking, but not always, these networks have an input layer, one output layer, and layers in between. Every neuron in the neural network's input layer, called the "input layer," is connected to the "intermediate layer" so that the data it receives can be sent further. Each neuron in the neural network's input layer is linked to the intermediate layer, allowing it to continue transmitting the data it has received as input (Kristof, 2002). Weighted synapses link the neurons of the input layer to those of the inner layers. The significant features from the perspective of the particular issue are used to determine how much weight to initially assign to each job, which is pretty task-specific. While the neurons in the input layer have fixed weights, the consequences in the intermediate layer are dynamic, demonstrating the potential to learn. The teaching samples that are sent to the network influence these changes. The rules seem to be learned from the supplied representative sample rather than being explicitly provided by the researchers; in other words, neural network learning is simply the act of dynamically allocating weights to intermediate components (Jain, et al., 1996). In the early stages of learning, neurons provide a random response to the issue, then compare the outcome to the teaching sample and adjust the internal weighting accordingly. Despite its outstanding findings, the neural in the case of networks, the output provided after the process is not guaranteed to be the ideal result. Throughout learning, the network adapts to a particular local minimum value via serial mutations. Most training networks use supervised, unsupervised, or hybrid training methods. The quality of the information given to the network varies depending on which training method is used. In the first approach, the network aims to adjust the weighting to produce outputs near the set of predicted results (the sample with the most essential information) as feasible (Jain, et al., 1996). In a variant known as reinforcement learning, the network doesn't receive the anticipated result but instead works on a value obtained from the discrepancy between the actual and expected outcomes. Instead of producing outputs for various inputs, networks in unsupervised learning aim to reveal the basic patterns present in the training sample and their interactions. In the case of hybrid learning, half of the weights are mapped manually, while the other half is created via directed learning (Jain, et al., 1996). For neural networks to be used in the real world, they need to be well-trained and given data with a similar structure they have never seen before. They can then use the knowledge they gained during training to complete the task.

## Descriptions of well-known neural networks

In addition to the many theoretical models, researchers may choose from a wide range of real-world applications. The present explanation focuses on elements that often come to the forefront in face recognition research. Since these application techniques differ, they may also be investigated according to qualities (e.g., topology, layer depth, neuron composition, information propagation direction). The facial recognition scientific literature is hence also referred to as literature.

Two main kinds of configurations often appear in the literature: so-called feedforward and recurrent networks, depending on the direction in which information might travel between the network's neurons(). The neuronal architecture of the earlier networks results in a non-circular graph, which causes the outputs to flow from the input directly to the output direction. In contrast, the architecture of the later networks contains neurons with multi-directional connections, which causes their architecture to remain circular. As stated before, neurons are grouped in layers in both scenarios and produce outputs dependent on the inputs provided by layers that came before them. As they travel down the whole length of the mesh, these outputs are constantly compared to the desired result. The outcome, often referred to as the loss, is the difference between the predicted and actual results. The error rate generated during the performance test travels back to the earlier layers through a process called backpropagation. It helps to dynamically modify the weights representing the importance of the different input neurons (Tang & Fishwick, 1993). Thus, by continually distributing weights in the solution, such nets essentially teach themselves which parts are involved in the problem and in what proportion. Topology, or the connections between the neurons that make up a network, is a characteristic of the network that significantly influences how learning generally proceeds. The most obvious illustration of this is the amount (depth) of the layers that make up the network; deep neural networks are networks with many layers, and their typical learning method is deep learning, although exact numbers are sometimes not explicitly provided (Yi, et al., 2015).

### Convolutional Neural Network CNN

These networks are distinguished from others because image processing is their major function. They are also capable of comprehending a variety of information types (video, audio, etc.). Our ultimate goal is to classify things based on the information in the picture, which is accomplished by transferring the data of an image (at the pixel level) via the network in a typical application.

A convolutional neural network (CNN) scans the details of incoming data rather than interpreting it as a whole. Using a layer of 1,000,000 (1,000×1,000) neurons for pixel-level interpretation is not the most efficient when a 1,000×1,000 pixel image is the input. Instead, the data is sent over the network in detail using a 100x100 pixel filter (Dertat, 2017).

|  |
| --- |
| Diagram  Description automatically generated  Figure ..Operation of the filter unit used by convolutional neural networks (Dertat, 2017). |

The two primary components of CNNs are feature discovery and categorisation. During the feature detection process, averaging (convolution), merging, and compression operations are carried out on the image's component units. For instance, if a photo of a dog is submitted into the system, feature recognition will compress the pixels to produce the image's recognisable forms (ears, mouth, legs). The properly trained and parameterised classification can look at all the features based on the ones that have already been found. It can then interpret and classify the whole scene in the image. To train sophisticated neural networks, pre-tagged data sets are required. The most challenging job is to identify system-specific hyperparameters, "structure-defining parameters": construct a neural network, i.e. how many layers are used, how many neurons should be put in the layer, activation function defined in the most layers, size and structure of the filter utilised (Dertat, 2017).

### Recurrent Neural Network RNN

A feedforward neural network has no memory, so it cannot recall past events. Of course, the exception to this rule is the training process, where neuron weight values ​​change. In a recurrent neural network (RNN), information loops so that neurons can think about how they responded to inputs from before when deciding what to send out.

Let's say that the word "neuron" is fed into a feedforward network, and the system goes through it character by character. When figuring out the output for the 'r' input, characters that have already been used cannot be considered. To determine what a complicated object means, you must think about how its parts work together.

The problem is solved via recurrent neural networks. They are recursive networks where data may be stored indefinitely (Olah, 2015).

|  |
| --- |
| Diagram  Description automatically generated  Figure .. Recurrent Neural Networks (Olah, 2015). |

As seen above, a section of the neural network labelled A processes the input xt and returns the result ht. The network's information may be transmitted from node to node through a loop.

Recurrent neural networks are imbued with an aura of mysticism thanks to the presence of these loops. However, if you think about it, you'll see that they aren't all that dissimilar to a standard neural network (Olah, 2015). A recurrent neural network is made up of several copies of the same network, each of which transmits a message to the network that comes after it. Think about what would happen if the loop were unrolled (Olah, 2015).

|  |
| --- |
| A picture containing text, clock  Description automatically generated  Figure .. An unrolled recurrent neural network (Olah, 2015). |

# Analysis and Design

## Tools

### Jupiter Notebook

Jupyter is an abbreviation for Julia, Python, and R; these are the three programming languages with which Jupyter originated, but it now supports various languages. Jupyter Notebook is an online application that is both open-source and free to use. Jupyter Notebook is made so that programming work can be shown and others can join easily. With Jupyter Notebook, programmers can combine code, comments, multimedia, and visuals into an interactive document called a "notebook." Recycled and used again. It allows users to write and execute computer code directly in the web browser. It is useful in teaching as we can show examples of how a script or a language works.

**Advantages of using Jupyter Notebook:**

* **Data visualisation.** Most people learn about Jupyter Notebooks for the first time through data visualisation, which is a shared notebook with some datasets shown as graphics. Jupyter Notebook lets users make visualisations, share them, and make real-time changes to the shared code and dataset.
* **Code Share.** Cloud platforms such as GitHub and Pastebin allow developers to exchange code but are generally inactive. Jupyter Notebook will enable developers to preview your code, run it, and examine the results immediately in your browser**.**
* **Live interaction with code.** The code in Jupyter Notebook is not static; it is real-time, gradually editable, and replayable, with feedback provided immediately in the browser. Notebooks can have user controls that can be utilised as code input sources.
* **Documenting code samples.** A developer might put some code in a Jupyter Notebook if they want to show how it works step by step with real-time feedback. The code is still fully functional, and the developer can add interaction by explaining, delivering, and talking simultaneously.

A Jupyter Notebook can have many parts, and each one is made up of different blocks.

**Components of Jupyter Notebook:**

* **Text and HTML** Anywhere on the page, developers can enter plain text or content written in Markdown syntax to turn it into HTML. The notebook template can have a built-in CSS style or be added to it.
* **Code and output.** Although developers can add support for other languages in the Jupyter environment, such as R or Julia, Jupyter Notebooks' programming is usually written in Python. The code blocks can be run and repeated in whatever order as many times as desired, and the results of the executed code appear immediately after the code blocks.
* **Multimedia** Due to the fact that it is built on web technologies, Jupyter Notebook can display supported forms of multimedia on a web page. They can be pre-programmed by developers with the help of the IPython.display module, or they can be inserted into a notebook as HTML elements.
* **Data** In addition to the .ipynb file that makes up a Jupyter Notebook, data can be given as a separate file or imported programmatically. For instance, code might be inserted into the notebook to download data from a public Internet repository or access a database connection.

With certain limits, Jupyter Notebook can be equally as powerful and helpful.

**The restriction set by Jupyter Notebook**

* **Notebooks are not self-contained.** The fact that Jupyter Notebook requires the Jupyter runtime and the libraries the developer wants to use is its biggest drawback. There are a few ways to create independent Jupyter Notebooks, but none of them is supported by the project. It's best to install or give laptops to those with the necessary infrastructure (via Anaconda, for example).
* **The session state is difficult to save.** Using the Jupyter Notebook toolset, you can't save the current state of the code in a Jupyter notebook and then load it again. Every time you load the notebook, the developer must run the code again to get it back to the way it was.
* **There is no interactive debugging or other IDE functionality.** Jupyter Notebook does not provide a full Python programming environment. Many functions that users anticipate from an IDE, such as interactive debugging, code completion, and module management, are missing.

### Python

**What is Python?**

Python is a powerful, high-level programming language that aims to be easy to read and understand. The term "high-level" refers to how closely related to human languages it is in comparison to other computer languages. Since the Python community is vibrant and active, learning this language will provide developers access to a wealth of useful tools that Python users have created since its inception.

**Why Python?**

In a nutshell, Python's popularity among developers is a result of its widespread use in a variety of industries, including data science, artificial intelligence, desktop and online application development, statistics, mathematics, and scientific research. Developers have access to several publicly accessible libraries, and the number of ready-mades, user-friendly tools is always growing. Its development community, which is already sizable, is expanding quickly due to the language's ease of learning and suitability for novice programmers. And if someone gets stuck on a work, it's always simple to find information, solutions, and recommendations owing to the enormous Python community.

**What does Python's simplicity mean?**

Comparing the approaches that Python and other programming languages (in this case, Java and C++) take to solve the same straightforward problem is the greatest way to illustrate Python's openness and accessibility. Let's use a classic programming example: get the terminal to say, "Hello World!"

|  |  |  |
| --- | --- | --- |
| C++ | Java | Python |
| #include <iostream>  int main()  {  std::cout<<"Hello,world!\n";  return 0;  } | class HelloWorldApp {  publicstaticvoid main(String[] args) {  System.out.println("Hello World!");  }  } | print("Hello World") |

## Framework

### TensorFlow

Since Google released version 1.0 of its open-source AI framework in 2017, it has grown into a platform that defines the Industry as a whole. With the framework and the detailed documentation that comes with it, it's very easy to put together neural networks with only a few lines of code that can solve a wide range of problems. These networks can run on both CPUs and GPUs (not to mention those specifically designed for solving AI problems). About the hardware for the TPU (Tensor Processing Unit) target (TensorFlow, 2019).

Tensorflow's operation is built on a data flow-based implementation in which the computations and states of the algorithms are represented by data flow graphs, allowing for the parallelisation of diverse processes (TensorFlow, 2019). The technology also can continually modify the internal states, allowing the vast number of parameters that occur in big models to be managed (TensorFlow, 2019). The framework was created in C++ (and CUDA-C), but it is now accessible in several programming languages, including C# and Java; however, it is most commonly used through Python language integration.

They often don't work directly with TensorFlow but rather with Keras, a higher-level software package that was not originally made for TensorFlow but is now part of it. Keras supported lower-level backends and was not originally made for TensorFlow (Team, n.d.). The project's original goal was to create a more user-friendly API interface, achieved with network configurations that can be set up with just a few lines of code. The API also makes it easy to access important functions, such as the different activation functions used when building a network (e.g. ReLu, Sigmoid, or Softmax42). It might seem obvious that the same backend should run both the neural networks made with TensorFlow and the training, which needs the same processing power as the net size. However, the software package's popularity is demonstrated by the addition of several solutions over the years, such as TensorFlow.js, which targets browsers and other runtime environments that support JavaScript execution, or TensorFlow Lite, which can be used to drive pre-trained networks in a mobile environment (TensorFlow, 2019). They also created APIs for many languages. (An example is Tensorflow Java.) We need to modify the model for both strategies, although the first can provide more training.

The Author is choosing TensorFlow to build his system because TensorFlow's most well-known Deep Learning framework comes with pre-trained models that can help with image classification. CNN is used to put photos into groups. Most of the time, all it takes to make a model is to put photos into groups to make a similar image, which is the positive image. The picture is then taught and retaught using a method called anchoring or Transfer Learning.

# Bibliography

Anon., n.d. *Review of Face Recognition Techniques - Pennsylvania State University.* [Online]   
Available at: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.742.1483&rep=rep1&type=pdf  
[Accessed 12 10 2022].

Anon., n.d. *What is Google Cloud Vision?* [Online]   
Available at: www.resourcespace.com  
[Accessed 25 10 2022].

Beresford, R. & Agatonovic-Kustrin, S., 2000. Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research. *Journal of Pharmaceutical and Biomedical Analysis,* 22(5), pp. 717-727.

Çarıkçı, M., 2012. A Face Recognition System Based on Eigenfaces Method. p. 6.

DeepFace, 2022. *Wikipedia.* [Online]   
Available at: https://en.wikipedia.org/wiki/DeepFace#:~:text=DeepFace%20is%20a%20deep%20learning  
[Accessed 23 10 2022].

Dertat, A., 2017. *towards data science.* [Online]   
Available at: https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2  
[Accessed 4 November 2022].

GeeksforGeeks, 2021. *GeeksforGeeks Multi-Layer Perceptron Learning in Tensorflow.* [Online]   
Available at: https://www.geeksforgeeks.org/multi-layer-perceptron-learning-in-tensorflow/  
[Accessed 1 11 2022].

Goyal, K., Agarwal, K. & Kumar, R., 2017. *ieeexplore.* [Online]   
Available at: https://ieeexplore.ieee.org/document/8203730/authors#authors  
[Accessed 6 November 2022].

Jain, A. K., Mao, J. & Mohiuddin, K., 1996. Artificial neural networks: a tutorial. *IEEE Computer,* , 29(3), pp. 31-44.

Karamizadeh, S. et al., 2013. *Journal of Signal and Information Processing.* [Online]   
Available at: https://file.scirp.org/pdf/JSIP\_2013101711003963.pdf  
[Accessed 31 10 2022].

Khan, F., 2018. Facial Expression Recognition using Facial Landmark Detection and Feature Extraction via Neural Networks. *ArXiv,* Volume abs/1812.04510, p. 7.

Kristof, T., 2002. *researchgate.* [Online]   
Available at: https://www.researchgate.net/publication/283463205\_A\_mesterseges\_neuralis\_halok\_a\_jovokutatas\_szolgalataban\_Artificial\_neural\_networks\_in\_Futures\_Studies  
[Accessed 1 11 2022].

McCulloch, W. S. & Pitts, W., 2014. *Cambridge Core.* [Online]   
Available at: https://www.cambridge.org/core/journals/journal-of-symbolic-logic/article/warren-s-mcculloch-and-walter-pitts-a-logical-calculus-of-the-ideas-immanent-in-nervous-activity-bulletin-of-mathematical-biophysics-vol-5-1943-pp-115133/7DFDC43EC1E5BD05E9DA85E1C41  
[Accessed 1 11 2022].

Olah, C., 2015. *GitHub Blog.* [Online]   
Available at: http://colah.github.io/posts/2015-08-Understanding-LSTMs/  
[Accessed 4 November 2022].

Pocket-lint, 2018. *Pocket-lint.* [Online]   
Available at: https://www.pocket-lint.com/phones/news/apple/142207-what-is-apple-face-id-and-how-does-it-work  
[Accessed 23 10 2022].

Shanu, S., n.d. *InsideAIML.* [Online]   
Available at: https://www.insideaiml.com/blog/Activation-Functions-In-Neural-Network-1089  
[Accessed 1 11 2022].

Solanki, K., 2016. Review of Face Recognition Techniques. *Review of Face Recognition Techniques,* p. 5.

Taigman, Y., Yang, M., Ranzato, M. & Wolf, L., 2014. *DeepFace: Closing the Gap to Human-Level Performance in Face Verification.* [Online]   
Available at: http://cs.toronto.edu/~ranzato/publications/taigman\_cvpr14.pdf  
[Accessed 23 10 2022].

Tang, Z. & Fishwick, P. A., 1993. *Researchgate.* [Online]   
Available at: https://www.researchgate.net/publication/220668844\_Feedforward\_Neural\_Nets\_as\_Models\_for\_Time\_Series\_Forecasting  
[Accessed 3 November 2022].

Viola, P. & M.Jones, 2003. *ieeexplore.* [Online]   
Available at: https://ieeexplore.ieee.org/document/990517  
[Accessed 6 November 2022].

Vision?, W. i. G. C., n.d. *www.resourcespace.com.* [Online]   
Available at: https://www.resourcespace.com/blog/what-is-google-vision  
[Accessed 25 10 2022].

Yi, S., Ding, L., Xiaogang , W. & Xiaoou , T., 2015. *researchgate.* [Online]   
Available at: https://www.researchgate.net/publication/271855676\_DeepID3\_Face\_Recognition\_with\_Very\_Deep\_Neural\_Networks  
[Accessed 03 November 2022].