Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Kerbala
College of Engineering
Department of Electrical and Electronic Engineering

Face Feature Fusion for Face Race Classification

A project
Submitted to the Department of Electrical and Electronic Engineering, University of Kerbala in a Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Electrical Engineering

by

Ali Ismail H. Ismail **Yousif M. Hussain M. Ali**

Supervised by

Prof. Dr. Hawraa H. Abbas

July 2021

Dedication

This research is lovingly dedicated to our parents, families and friends who have been our constant source of inspiration. They have given us the drive and discipline to tackle a task with enthusiasm and determination. Without their love and support, this project would not have been made possible. I would also like to dedicate this work to my vocational school teachers.

Acknowledgment

We would like to show our gratitude and appreciation to Eng. Ruaa Mohammed and Eng. Nawal Nihad for their cooperation in preparing the project dataset, and we wish to extend our special thanks to Eng. Qutaiba Ahmed for providing assistance in the project code.

We would also like to thank and appreciate our supervisor Prof. Dr Hawra H. Abbas for her guidance and support throughout the project.

Undertaking

This is to declare that the project entitled "***Face Feature Fusion for Face Race Classification***" is an original work done by undersigned, in partial fulfillment of the requirements for the degree of "Bachelor of Science in Electrical Engineering" at Electrical and Electronic Engineering Department, College of Engineering, University of Kerbala.

All the analysis, design and system development have been accomplished by the undersigned. Moreover, this project has not been submitted to any other college or university.

Ali Ismail H. Ismail Ali Ismail

Yousif M. Hussain M. Ali Yousif

Abstract

Racial and ethnic identity, commonly defined as the significance and meaning of race and ethnicity to one's self-concept represent crucial components of adolescent development and exploration among the people. From medical aspect understanding many diseases and face syndrome, which are caused by the human genes' mutation, should started by ethnic differences and classification understanding since many of human genes vary between the people according to their ethnic. Consequently, in this project the human face is used to classify the ethnic group of people. An automatic system is developed for that purpose, the deep learning artificial intelligent algorithm is used for extract the face features that used for race classification. The proposed work of the project is as follows (Collect Dataset – Crop Facial Landmarks – Feature Extraction – Training and classification). The project experiments are conducted on three datasets contain 120, 160, and 240 images from four ethnic groups (Asian, African-American, Asian Middle-eastern, and White)

List of Contents

List of Abbreviations

Abbreviation	Definition
AI	Artificial Intelligence
ANNs	Artificial Neural Networks
CNNs	Convolutional Neural Networks
DNNs	Deep Neural Networks
FERET	Face Recognition Technology
GANs	Generative Adversarial Networks
GPU	Graphical Processing Units
GUI	Graphical User Interface
IDE	Integrated Development Environment
ILSVRC	Imagenet Large Scale Visual Recognition Challenge
K-NN	K-Nearest Neighbors
LSTMs	Long Short-Term Memory Networks
MIT	Massachusetts Institute of Technology
MLP	Multi-Layer Perceptron
NIST	National Institute of Standards and Technology
PCA	Principal Component Analysis
RBFNs	Radial Basis Function Networks
RNNs	Recurrent Neural Networks
SNNs	Simulated Neural Networks
SVD	Singular Value Decomposition
SVMs	Support Vector Machines
VGG	Visual Geometry Group

List of Figures

List of Tables

Chapter One: Introduction

1.1 Introduction

Face analysis is one of the most studied research topics in the field of computer vision and pattern recognition for the past few decades. Although face of a person provides a variety of demographic information like gender, age, ethnicity, etc yet ethnicity remains one of the invariant and fundamental attributes that cannot be easily masked like age and gender even in disguise. Therefore, grouping people based on age and gender would not only complex the problem but will also yield wrong results. For this reason, ethnicity classification is a key component that can be deployed in various video surveillance systems at security checkpoints. Furthermore, this classification statement has potential application in image search query where prior knowledge of race would narrow down the search space in the database, thus simplifying the process.

Biometric identification has become a better method for classifying the human based on their physical appearance with a particular identity. Every human in a particular society will probably have the same racial features. This idea helps in various fields such as archaeological and biomedical researchers as they are undergoing identification of ancestral and ancient humans and pictures of human to locate them easily into a continent. Face recognition is the most common system deployed in places under keen vigilance like international airports, trade centres, and stadiums which performs passive identification unnoticed by the suspect. It is non-intrusive and accurate. Human facial images provide the demographic information, such as ethnicity, age and gender. Age and gender of a person can be easily masked, if in disguise. Classifying people based on age would be complex and cannot be strategically standardized. If there is a prior knowledge of the

race of the suspect, identification will be more efficient, and essentially narrow down the search space.

Race classification is the inherent ability for humans. When a person encounters a new individual, three primitive concepts in his brain will be stimulated by image captured by eyes, including race, gender and age. These concepts will appear in 150ms at the first stage of human face cognition. There exists a specific race processing unit involved with other cortical processing (similar to face processing).6 In the research of machine vision and artificial intelligence, we purchase the purpose of simulating the versatile ability of human eyes and brain. So, race classification by computer plays a key role in both theory research and practical application.

Human Recognition AI uses a lot of features like iris, thumbprint, faces, gait, voice etc to identify humans uniquely. However, over all these parameters facial images have been noted as a highly reliable metric for identifying humans. Every human being has a unique face which is distinguishable from other human beings. Demographic factors like race, gender, age and place of origin can be extracted from the face itself. Some factors like gender and age can be used by corporations to recommend products accordingly and thereby increase their sales. Thus, a successful gender classification approach can help in boosting the performance of a lot of areas of machine learning applications like human recognition, bio-metric verification and smart human-computer interfaces.

The application fields of race classification include security surveillance, such as unusual group gathering; human computer interaction, such as product recommendation based on photos in social network site; bioinformatics, such as improving the performance of face recognition and retrieval.

In recent years, social networks have become popular with billions of users around the world, with millions of pieces of information shared daily. Many studies on the application of social networks have been analysed recently. In 2016, Farnadi et al. gave a detailed analysis of various state-of-the-art methods for personality recognition in many datasets from Facebook, Twitter, and YouTube. In this year, Nguyen et al. used a Deep Neural Network (DNN) to meet two types of information needs of response organizations: (i) informative tweet identification and (ii) topical classes classification. They also provided a new learning algorithm using stochastic gradient descent to train DNNs in online learning during disaster situations. Recently, Carvalhoa et al. presented a smart platform to efficiently collect, manage, mine, and visualize large Twitter corpora.

In recent times, deep learning, which tries to learn good representations from raw data automatically with multilayers stacked on each other, has attracted significant attention from researchers due to its various applications in computer vision, natural language processing, and speech processing. Convolutional neural networks (CNNs), a type of deep learning model, have recently achieved many promising results in large-scale image and video recognition. The VGG model, which was first introduced by Simonyan and Zisserman in 2015, achieved very good performance on ImageNet and has been widely used in computer vision studies.

1.2 Problem Statement

The aim of this project is classifying the human ethnic according to their faces features and analyse the face morphology which is more related to race change.

1.3 Project Objectives

1. Collect a dataset for Arabic ethnic group faces.
2. Establish an automatic race classification system.
3. Analyse the face morphology (eyes, mouth, and nose) and test which face morphology is more effected in ethnic classification.

1.4 Project Organisation

This research consists of five chapters:

1. Chapter One Introduction: contains the introduction of the research.
2. Chapter Two Literature Review: covers the related studies and related works
3. Chapter Three Background Algorithms: covers the methodology and the proposed work.
4. Chapter Four Experimental Results: covers the results of the experiments.
5. Chapter Five Conclusion and Future Work: it is the last chapter and it contains the conclusion of the results and the future work.

Chapter Two: Literature Review

2.1 Related Studies

Last decade, many effective methods were proposed in this active research field. On the whole, there are three categories method about race classification:

1. Holistic face methods. Most existing approaches have focused on holistic face image. They adopt mathematic and statistic methods to simulate neural working process for extracting discriminating features. Integrating with the LDA analysis for the input face images at different scales, Lus method presents for the two-class (Asian versus non-Asian) ethnicity classification task and gets 97.7% accurate rate performance. Guo propose a biologically inspired feature in their race classification system and the experiments on African American versus Caucasian show the 98% accurate rate. And then, they propose a canonical correlation analysis (CCA)-based method to estimate age, gender and ethnicity simultaneously. Manesh et al. employs the optimum decision-making rule on the confidence level of automatically separated face regions using the modified Golden ratio mask, and adopts SVM as the classifier on the extracted Gabor features of each patch to get its confidence level. This method can get 98% accurate rate for distinguishing Asian from Non-Asian.
2. Local feature methods. Iris texture is proved the one of the most effective features for race classification. Sun et al. put forward to a hierarchical visual codebook-based method for iris image classification. And their experiments show the remarkable performance. But, the shortage of these methods is obvious, because the iris texture is difficult to be extracted for the reason of long-distance camera in practical applications. Except for iris, periocular

region also has plenty of information about the essential characteristic of race. Lyle et al. propose an LBP feature-based method to employ texture feature of whole periocular region to classify Asian from Non-Asian. A new method integrated Kernel Class-dependent Feature Analysis and facial colour-based features are proposed by Xie et al. to accomplish identifying race class (Asian/Africa America/Caucasian). In this method, only the upper image region including two whole periocular sub regions is considered, but no any local structure within periocular region.

3. 3D methods. 3D face model has more essential craniofacial feature for race than 2D image. Moreover, 2D-based race classification task can be interfered by the variant face expression and illumination. So, many scholars contribute their work on adopting 3D deep data to represent the essential face structure feature. Although perfect performance can be obtained, the difficult extraction method for 3D information limits the application field for 3D-based race classification.

2.2 Related Works

There are a number of algorithms that have been devised over the years for the ethnicity identification of humans.

Work done by P. Viola and M. Jones has provided the efficient and rapid method of detecting face in input image. This is a novel approach which uses Adaboost classifier. This has high detection rate with very less computation time on the dataset consisting of images under varying condition like illumination, pose, colour, camera variation; etc. This algorithm is used in our approach to detect face in the image which will be later processed further.

Lu et al. has proposed ethnicity classification algorithm in which image of the faces were examined at multiple scales. The Linear

Discriminant Analysis (LDA) scheme is used for input face images to improve the classification result. The accuracy of the performance of this approach is 96.3% on the database of 2,630 sample images of 263 subjects. However, the dataset considered in this work consisted only of two classes i.e., Asian and non-Asian.

Hosoi et al. have integrated the Gabor wavelet features and retina sampling for their work. These features were then used with the Support Vector Machines (SVM) classifier. This approach has used three categories: Asian, African and European. And the accuracy achieved for each category is: 96%, 94% and 93% respectively. However, their approach seemed to have issues when considering other ethnicities.

S. Md. Mansoor has used Viola Jones algorithm for face detection problem. After the detection of face, various features namely skin colour; lip colour and normalized forehead area were extracted from the image. This classification problem has used the Yale, FERET dataset of Mongolian, Caucasian and Negroid images. The overall accuracy achieved in this work with these features was 81%.

It was evident from the above mentioned works that none had considered geometric features for their solution. Also, the scope of pre-trained Convolution Neural Network has yet not been explored for this problem so far.

In recent years, deep CNNs have been used extensively in computer vision especially due to their promising performance. For the problem of image classification, Zhang et al. proposed a novel feature learning method for halftone image classification with very good performance. This method uses stacked sparse auto-encoders (SAE) to extract features of halftone images by using unsupervised learning, and then uses SoftMax regression to fine-tune the deep neural network using supervised learning to classify

halftone images. Wei et al. proposed a flexible deep CNN model, called Hypotheses-CNN-Pooling, for multilabel image classification. The input of this framework is an arbitrary number of object segment hypotheses. Then, a shared CNN is linked to each hypothesis. Finally, the results from different hypotheses are summed with max pooling to generate the final multilabel predictions.

Recognizing faces of their own ethnicity/race by humans. More activity in brain regions linked to face recognition discussed by Golby et al. The same-race for face identification involves Form Face Area (FFA) is examined by Functional magnetic resonance imaging (FMRI). Investigate the differences in the way people perceive own- versus other-race faces shown by O'Toole et al. People categorize faces of their own-race by sex more efficiently by O'Toole et al. Database is focused by Identity-related features so; retrieval of information is very effective. Same-race faces elicit more activity in brain regions linked by face recognition Golby et al. Important for face recognition identified by the Functional Magnetic Resonance Imaging (FMRI), same-race for face identification involves form FFA, The way people perceive own- versus other-race faces investigated by O'Toole et al. The people categorize faces of their own-race by sex more efficiently than another race discussed by O'Toole et al. The face recognition system of race and gender can help identification to focus more on the identity-related features, and limit the number of entries to be searched in a large database, the search speed and efficiency of the retrieval systems improved. The demographic statistics in many social applications ethnicity and gender are also useful. The ethnic categories are loosely defined classes than the identity. Ethnicity classification into a two-category (Asian and non-Asian) classification problem is discussed in this paper. In, two types of features were used, Linear Discriminant Analysis based algebraic features an elastic model based geometric features. They classified images into three

minority Chinese groups. An accuracy of 79% was reported using algebraic features and 90.95% with geometric features with K-Nearest Neighbours (k-NN) and C5.0 classifiers.

Chapter Three: Backgrounds Algorithms

3.1 Artificial Intelligence (AI)

It is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable.

Decades before this definition, the birth of the artificial intelligence conversation was denoted by Alan Turing's seminal work, "Computing Machinery and Intelligence", which was published in 1950. In this paper, Turing, often referred to as the "father of computer science", asks the following question, "Can machines think?" From there, he offers a test, now famously known as the "Turing Test", where a human interrogator would try to distinguish between a computer and human text response. While this test has undergone much scrutiny since it publishes, it remains an important part of the history of AI as well as an ongoing concept within philosophy as it utilizes ideas around linguistics.

At its simplest form, artificial intelligence is a field, which combines computer science and robust datasets, to enable problem-solving. It also encompasses sub-fields of machine learning and deep learning, which are frequently mentioned in conjunction with artificial intelligence. These disciplines are comprised of AI algorithms which seek to create expert systems which make predictions or classifications based on input data.

3.2 Machine Learning

Machine learning is a continually advancing technology that allows computers to learn from previous data automatically. Machine learning uses various algorithms for building mathematical models and making

predictions using historical data or information. It is now applied for Image recognition, speech recognition, email filtering, healthcare, etc.

Machine learning involves computers discovering how they can perform tasks without being explicitly programmed to do so. It involves computers learning from data provided so that they carry out certain tasks. For simple tasks assigned to computers, it is possible to program algorithms telling the machine how to execute all steps required to solve the problem at hand; on the computer's part, no learning is needed. For more advanced tasks, it can be challenging for a human to manually create the needed algorithms. In practice, it can turn out to be more effective to help the machine develop its own algorithm, rather than having human programmers specify every needed step.

Machine learning is an important component of the growing field of data science. Through the use of statistical methods, algorithms are trained to make classifications or predictions, uncovering key insights within data mining projects. These insights subsequently drive decision making within applications and businesses, ideally impacting key growth metrics. As big data continues to expand and grow, the market demand for data scientists will increase, requiring them to assist in the identification of the most relevant business questions and subsequently the data to answer them.

3.2.1 *Machine Learning Methodology*

The learning system of a machine learning algorithm can be divided into three main parts:

1. A Decision Process: In general, machine learning algorithms are used to make a prediction or classification. Based on some input data, which can be labelled or unlabelled, your algorithm will produce an estimate about a pattern in the data.

2. An Error Function: An error function serves to evaluate the prediction of the model. If there are known examples, an error function can make a comparison to assess the accuracy of the model.
3. A Model Optimization Process: If the model can fit better to the data points in the training set, then weights are adjusted to reduce the discrepancy between the known example and the model estimate. The algorithm will repeat this evaluate and optimize process, updating weights autonomously until a threshold of accuracy has been met.

3.2.2 *Classification of Machine Learning*

Machine learning classifiers fall into four primary categories.

A. Supervised learning
B. Unsupervised learning
C. Semi-supervised learning
D. Reinforcement learning

A. Supervised Learning

Supervised learning, also known as supervised machine learning, is defined by its use of labelled datasets to train algorithms that to classify data or predict outcomes accurately. As input data is fed into the model, it adjusts its weights until the model has been fitted appropriately. This occurs as part of the cross-validation process to ensure that the model avoids overfitting or underfitting. Supervised learning helps organizations solve for a variety of real-world problems at scale, such as classifying spam in a separate folder from your inbox. Some methods used in supervised learning include neural networks, naïve bayes, linear regression, logistic regression, random forest, support vector machine (SVM), and more.

B. Unsupervised learning

Unsupervised learning, also known as unsupervised machine learning, uses machine learning algorithms to analyse and cluster unlabelled datasets.

These algorithms discover hidden patterns or data groupings without the need for human intervention. Its ability to discover similarities and differences in information make it the ideal solution for exploratory data analysis, cross-selling strategies, customer segmentation, image and pattern recognition. It's also used to reduce the number of features in a model through the process of dimensionality reduction; principal component analysis (PCA) and singular value decomposition (SVD) are two common approaches for this. Other algorithms used in unsupervised learning include neural networks, k-means clustering, probabilistic clustering methods, and more.

C. Semi-supervised learning

Semi-supervised learning offers a medium between supervised and unsupervised learning. During training, it uses a smaller labelled data set to guide classification and feature extraction from a larger, unlabelled data set. it can solve the problem of having not enough labelled data (or not being able to afford to label enough data) to train a supervised learning algorithm.

D. Reinforcement Learning

Reinforcement machine learning is a behavioural machine learning model that is similar to supervised learning, but the algorithm isn't trained using sample data. This model learns by using trial and error. A sequence of successful outcomes will be reinforced to develop the best recommendation for a given problem. The IBM Watson system that won "the Jeopardy challenge" in 2011 makes a good example. The system used reinforcement learning to decide whether to attempt an answer or question, which square to select on the board, and how much to wager—especially on daily doubles.

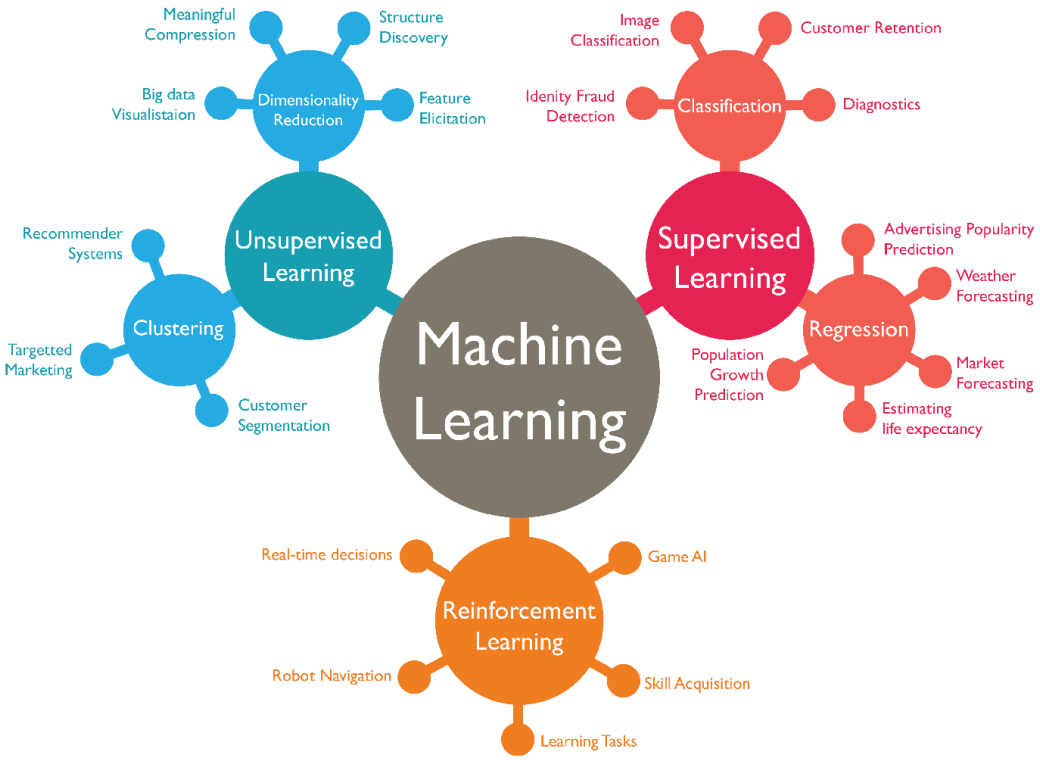


Fig. 3.1 Classification of Machine Learning

3.3 AI Neural Networks

Neural networks—and more specifically, artificial neural networks (ANNs)—mimic the human brain through a set of algorithms. Neural networks, also known as simulated neural networks (SNNs), are a subset of machine learning and are at the heart of deep learning algorithms. At a basic level, a neural network is comprised of four main components: inputs, weights, a bias or threshold, and an output. Similar to linear regression.

Artificial neural networks (ANNs) are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

Neural networks rely on training data to learn and improve their accuracy over time. However, once these learning algorithms are fine-tuned for accuracy, they are powerful tools in artificial intelligence, allowing us to classify and cluster data at a high velocity. Tasks in speech recognition or image recognition can take minutes versus hours when compared to the manual identification by human experts.

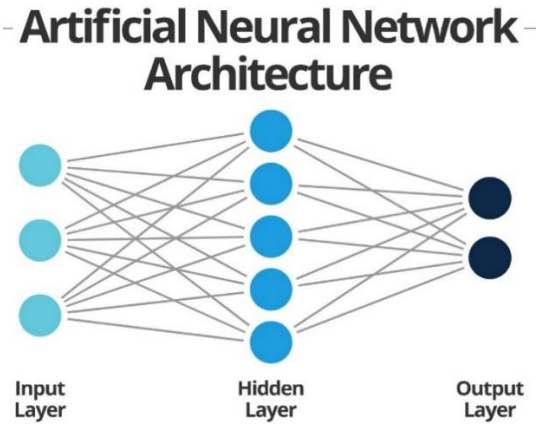


Fig. 3.2 Artificial Neural Network

The main difference between regression and a neural network is the impact of change on a single weight. In regression, you can change a weight without affecting the other inputs in a function. However, this isn't the case with neural networks. Since the output of one layer is passed into the next layer of the network, a single change can have a cascading effect on the other neurons in the network.

3.3.1 *Neural Networks Methodology*

each individual node as its own linear regression model, composed of input data, weights, a bias (or threshold), and an output. The formula would look like this:

$$\sum_{i=1}^{m} W_i + bias = W_1X_1 + W_2X_2 + W_3X_3 + bias \qquad (3.1)$$

$$Output = f(x) = \begin{cases} 1 \; if \; \sum W_1 X_1 + b \geq 0 \\ 0 \; if \; \sum W_1 X_1 + b < 0 \end{cases} \quad (3.2)$$

Once an input layer is determined, weights are assigned. These weights help determine the importance of any given variable, with larger ones contributing more significantly to the output compared to other inputs. All inputs are then multiplied by their respective weights and then summed. Afterward, the output is passed through an activation function, which determines the output. If that output exceeds a given threshold, its "fires" (or activates) the node, passing data to the next layer in the network. This results in the output of one node becoming in the input of the next node. This process of passing data from one layer to the next layer defines this neural network as a feedforward network.

Let's break down what one single node might look like using binary values. We can apply this concept to a more tangible example, like whether you should go surfing (Yes: 1, No: 0). The decision to go or not to go is our predicted outcome, or y-hat. Let's assume that there are three factors influencing your decision-making:

- Are the waves good? (Yes: 1, No: 0)
- Is the line-up empty? (Yes: 1, No: 0)
- Has there been a recent shark attack? (Yes: 0, No: 1)

Then, let's assume the following, giving us the following inputs:

- X1 = 1, since the waves are pumping
- X2 = 0, since the crowds are out
- X3 = 1, since there hasn't been a recent shark attack

Now, it needs to assign some weights to determine importance. Larger weights signify those particular variables are of greater importance to the decision or outcome.

- W1 = 5, since large swells don't come around often
- W2 = 2, since you're used to the crowds
- W3 = 4, since you have a fear of sharks

Finally, assuming a threshold value of 3, which would translate to a bias value of –3. With all the various inputs, we can start to plug in values into the formula to get the desired output.

Y-hat = (1*5) + (0*2) + (1*4) – 3 = 6

If the activation function is used from the beginning of this section, it is possible to determine that the output of this node would be 1, since 6 is greater than 0. In this instance, you would go surfing; but if we adjust the weights or the threshold, we can achieve different outcomes from the model. When we observe one decision, like in the above example, we can see how a neural network could make increasingly complex decisions depending on the output of previous decisions or layers.

In the example above, perceptron was used to illustrate some of the mathematics at play here, but neural networks leverage sigmoid neurons, which are distinguished by having values between 0 and 1. Since neural networks behave similarly to decision trees, cascading data from one node to another, having x values between 0 and 1 will reduce the impact of any given change of a single variable on the output of any given node, and subsequently, the output of the neural network.

As we start to think about more practical use cases for neural networks, like image recognition or classification, we'll leverage supervised learning, or labelled datasets, to train the algorithm. As we train the model, we'll want to evaluate its accuracy using a cost (or loss) function. This is also commonly referred to as the mean squared error (MSE). In the equation below:

$$Cost\ Function = MSE = \frac{1}{2m}\sum_{i=1}^{m}(\hat{y} - y)^2 \quad (3.3)$$

- i represents the index of the sample,
- y-hat is the predicted outcome,
- y is the actual value, and
- m is the number of samples.

Ultimately, the goal is to minimize our cost function to ensure correctness of fit for any given observation. As the model adjusts its weights and bias, it uses the cost function and reinforcement learning to reach the point of convergence, or the local minimum. The process in which the algorithm adjusts its weights is through gradient descent, allowing the model to determine the direction to take to reduce errors (or minimize the cost function). With each training example, the parameters of the model adjust to gradually converge at the minimum.

Most deep neural networks are feedforward, meaning they flow in one direction only, from input to output. However, you can also train your model through backpropagation; that is, move in the opposite direction from output to input. Backpropagation allows us to calculate and attribute the error associated with each neuron, allowing us to adjust and fit the parameters of the model(s) appropriately.

3.3.2 *Types of Neural Networks*

Neural networks can be classified into different types, which are used for different purposes.

The perceptron is the oldest neural network, created by Frank Rosenblatt in 1958. It has a single neuron and is the simplest form of a neural network. As shown in Fig. 3.2.

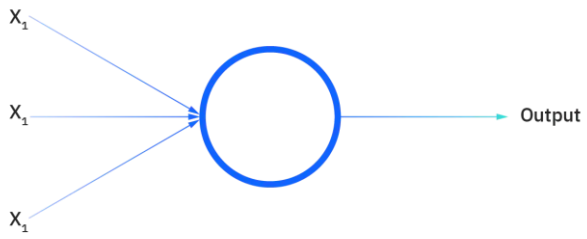


Fig. 3.3 The Perceptron Neural Network

Feedforward neural networks, or multi-layer perceptron (MLPs), are what we've primarily been focusing on within this article. They are comprised of an input layer, a hidden layer or layers, and an output layer. While these neural networks are also commonly referred to as MLPs, it's important to note that they are actually comprised of sigmoid neurons, not perceptron, as most real-world problems are nonlinear. Data usually is fed into these models to train them, and they are the foundation for computer vision, natural language processing, and other neural networks.

Convolutional neural networks (CNNs) are similar to feedforward networks, but they're usually utilized for image recognition, pattern recognition, and/or computer vision. These networks harness principles from linear algebra, particularly matrix multiplication, to identify patterns within an image.

Recurrent neural networks (RNNs) are identified by their feedback loops. These learning algorithms are primarily leveraged when using time-series data to make predictions about future outcomes, such as stock market predictions or sales forecasting.

3.4 Deep Learning and Machine Learning

Since deep learning and machine learning tend to be used interchangeably, it's worth noting the nuances between the two. As mentioned above, both deep learning and machine learning are sub-fields of

artificial intelligence, and deep learning is actually a sub-field of machine learning.

Deep learning is actually comprised of neural networks. “Deep” in deep learning refers to a neural network comprised of more than three layers—which would be inclusive of the inputs and the output—can be considered a deep learning algorithm. This is generally represented in Fig. 3.3.

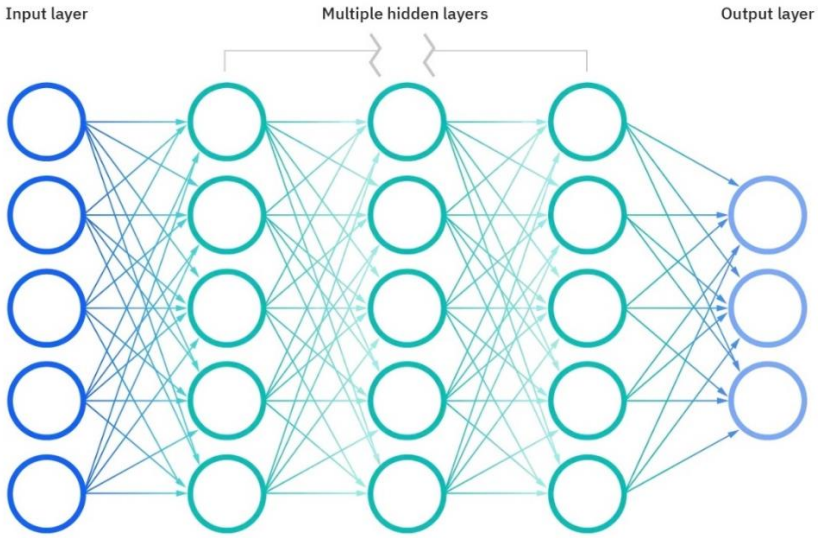


Fig. 3.4 Deep Neural Network

The way in which deep learning and machine learning differ is in how each algorithm learns. Deep learning automates much of the feature extraction piece of the process, eliminating some of the manual human intervention required and enabling the use of larger data sets. It’s possible to consider deep learning as "scalable machine learning" as Lex Fridman noted in an MIT lecture. Classical, or "non-deep", machine learning is more dependent on human intervention to learn. Human experts determine the hierarchy of features to understand the differences between data inputs, usually requiring more structured data to learn.

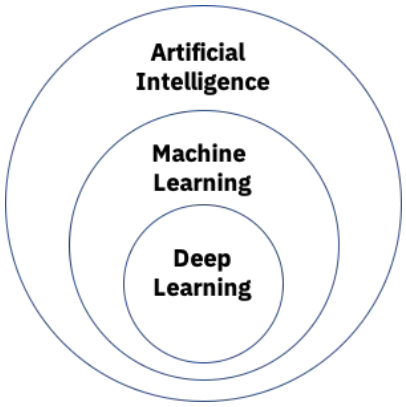


Fig. 3.5 Artificial Intelligence Divisions

"Deep" machine learning can leverage labelled datasets, also known as supervised learning, to inform its algorithm, but it doesn’t necessarily require a labelled dataset. It can ingest unstructured data in its raw form (e.g., text, images), and it can automatically determine the hierarchy of features which distinguish different categories of data from one another. Unlike machine learning, it doesn't require human intervention to process data, allowing us to scale machine learning in more interesting ways.

3.5 Deep Learning

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behaviour of the human brain, allowing it to learn from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. Deep learning technology lies behind everyday products and services (such as digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies (such as self-driving cars).

3.5.1 *Deep Learning Methodology*

Deep learning neural networks, or artificial neural networks, attempts to mimic the human brain through a combination of data inputs, weights, and bias. These elements work together to accurately recognize, classify, and describe objects within the data.

Deep neural networks consist of multiple layers of interconnected nodes, each building upon the previous layer to refine and optimize the prediction or categorization. This progression of computations through the network is called forward propagation. The input and output layers of a deep neural network are called visible layers. The input layer is where the deep learning model ingests the data for processing, and the output layer is where the final prediction or classification is made.

Another process called backpropagation uses algorithms, like gradient descent, to calculate errors in predictions and then adjusts the weights and biases of the function by moving backwards through the layers in an effort to train the model. Together, forward propagation and backpropagation allow a neural network to make predictions and correct for any errors accordingly. Over time, the algorithm becomes gradually more accurate.

The above describes the simplest type of deep neural network in the simplest terms. However, deep learning algorithms are incredibly complex, and there are different types of neural networks to address specific problems or datasets. Some of the types are:

1. Convolutional Neural Networks (CNNs)
2. Long Short-Term Memory Networks (LSTMs)
3. Recurrent Neural Networks (RNNs)
4. Generative Adversarial Networks (GANs)
5. Radial Basis Function Networks (RBFNs)

3.5.2 *Convolutional Neural Networks (CNNs)*

CNN's, also known as ConvNets, consist of multiple layers and are mainly used for image processing and object detection. Yann LeCun developed the first CNN in 1988 when it was called LeNet. It was used for recognizing characters like ZIP codes and digits.

Neural networks are a subset of machine learning, and they are at the heart of deep learning algorithms. They are comprised of node layers, containing an input layer, one or more hidden layers, and an output layer. Each node connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

There are various types of neural nets, which are used for different use cases and data types. For example, recurrent neural networks are commonly used for natural language processing and speech recognition whereas convolutional neural networks (ConvNets or CNNs) are more often utilized for classification and computer vision tasks. Prior to CNNs, manual, time-consuming feature extraction methods were used to identify objects in images. However, convolutional neural networks now provide a more scalable approach to image classification and object recognition tasks, leveraging principles from linear algebra, specifically matrix multiplication, to identify patterns within an image. That said, they can be computationally demanding, requiring graphical processing units (GPUs) to train models.

3.5.3 *Convolutional Neural Networks Methodology*

Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have three main types of layers, which are:

A. Convolutional layer
B. Pooling layer
C. Fully-connected (FC) layer

The convolutional layer is the first layer of a convolutional network. While convolutional layers can be followed by additional convolutional layers or pooling layers, the fully-connected layer is the final layer. With each layer, the CNN increases in its complexity, identifying greater portions of the image. Earlier layers focus on simple features, such as colors and edges. As the image data progresses through the layers of the CNN, it starts to recognize larger elements or shapes of the object until it finally identifies the intended object.

A. Convolutional Layer

The convolutional layer is the core building block of a CNN, and it is where the majority of computation occurs. It requires a few components, which are input data, a filter, and a feature map. Let's assume that the input will be a color image, which is made up of a matrix of pixels in 3D. This means that the input will have three dimensions—a height, width, and depth—which correspond to RGB in an image. We also have a feature detector, also known as a kernel or a filter, which will move across the receptive fields of the image, checking if the feature is present. This process is known as a convolution.

The feature detector is a two-dimensional (2-D) array of weights, which represents part of the image. While they can vary in size, the filter size is typically a 3x3 matrix; this also determines the size of the receptive field. The filter is then applied to an area of the image, and a dot product is calculated between the input pixels and the filter. This dot product is then fed into an output array. Afterwards, the filter shifts by a stride, repeating the process until the kernel has swept across the entire image. The final output

from the series of dot products from the input and the filter is known as a feature map, activation map, or a convolved feature.

As shown in the image above, each output value in the feature map does not have to connect to each pixel value in the input image. It only needs to connect to the receptive field, where the filter is being applied. Since the output array does not need to map directly to each input value, convolutional (and pooling) layers are commonly referred to as "partially connected" layers. However, this characteristic can also be described as local connectivity.

Note that the weights in the feature detector remain fixed as it moves across the image, which is also known as parameter sharing. Some parameters, like the weight values, adjust during training through the process of backpropagation and gradient descent. However, there are three hyperparameters which affect the volume size of the output that need to be set before the training of the neural network begins. These include:

1. The number of filters affects the depth of the output. For example, three distinct filters would yield three different feature maps, creating a depth of three.
2. Stride is the distance, or number of pixels, that the kernel moves over the input matrix. While stride values of two or greater is rare, a larger stride yields a smaller output.
3. Zero-padding is usually used when the filters do not fit the input image. This sets all elements that fall outside of the input matrix to zero, producing a larger or equally sized output. There are three types of padding:
    a) Valid padding: This is also known as no padding. In this case, the last convolution is dropped if dimensions do not align.

b) Same padding: This padding ensures that the output layer has the same size as the input layer

c) Full padding: This type of padding increases the size of the output by adding zeros to the border of the input.

After each convolution operation, a CNN applies a Rectified Linear Unit (ReLU) transformation to the feature map, introducing nonlinearity to the model.

As mentioned earlier, another convolution layer can follow the initial convolution layer. When this happens, the structure of the CNN can become hierarchical as the later layers can see the pixels within the receptive fields of prior layers. As an example, let's assume that we're trying to determine if an image contains a bicycle. You can think of the bicycle as a sum of parts. It is comprised of a frame, handlebars, wheels, pedals, et cetera. Each individual part of the bicycle makes up a lower-level pattern in the neural net, and the combination of its parts represents a higher-level pattern, creating a feature hierarchy within the CNN.

Ultimately, the convolutional layer converts the image into numerical values, allowing the neural network to interpret and extract relevant patterns.

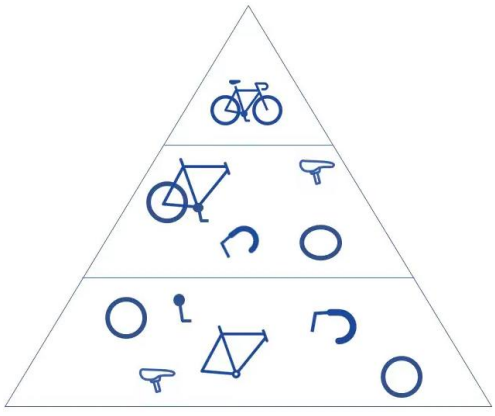


Fig. 3.6 An Example of Convolutional Layer Working Principle

B. Pooling Layer:

Pooling layers, also known as down sampling, conducts dimensionality reduction, reducing the number of parameters in the input. Similar to the convolutional layer, the pooling operation sweeps a filter across the entire input, but the difference is that this filter does not have any weights. Instead, the kernel applies an aggregation function to the values within the receptive field, populating the output array. There are two main types of pooling:

1. Max pooling: As the filter moves across the input, it selects the pixel with the maximum value to send to the output array. As an aside, this approach tends to be used more often compared to average pooling.
2. Average pooling: As the filter moves across the input, it calculates the average value within the receptive field to send to the output array.

While a lot of information is lost in the pooling layer, it also has a number of benefits to the CNN. They help to reduce complexity, improve efficiency, and limit risk of overfitting.

C. Fully-Connected Layer

The name of the full-connected layer aptly describes itself. As mentioned earlier, the pixel values of the input image are not directly connected to the output layer in partially connected layers. However, in the fully-connected layer, each node in the output layer connects directly to a node in the previous layer.

This layer performs the task of classification based on the features extracted through the previous layers and their different filters. While convolutional and pooling layers tend to use ReLu functions, FC layers usually leverage a SoftMax activation function to classify inputs appropriately, producing a probability from 0 to 1.

3.5.4 *CNN Architectures*

The powerful learning ability of deep CNN is primarily due to the use of multiple feature extraction stages that can automatically learn representations from the data. The availability of a large amount of data and improvement in the hardware technology has accelerated the research in CNNs, and recently interesting deep CNN architectures have been reported. Several inspiring ideas to bring advancements in CNNs have been explored, such as the use of different activation and loss functions, parameter optimization, regularization, and architectural innovations. However, the significant improvement in the representational capacity of the deep CNN is achieved through architectural innovations. Notably, the ideas of exploiting spatial and channel information, depth and width of architecture, and multi-path information processing have gained substantial attention. Similarly, the idea of using a block of layers as a structural unit is also gaining popularity. Some of the types of CNN architectures are (LeNet-5 (1998), AlexNet (2012), ZFNet (2013), GoogLeNet/Inception (2014), VGGNet (2014), and ResNet (2015)) in the next section the **VGGNet** architecture will be covered.

3.5.4.1 VGGNet (2014)

The successful use of CNNs in image recognition tasks has accelerated the research in architectural design. In this regard, Simonyan et al. proposed a simple and effective design principle for CNN architectures. Their architecture, named as VGG, was modular in layers pattern (Simonyan and Zisserman 2015). VGG was made 19 layers deep compared to AlexNet and ZfNet to simulate the relation of depth with the representational capacity of the network (Krizhevsky et al. 2012; Zeiler and Fergus 2013). ZfNet, which was a frontline network of 2013-ILSVRC competition, suggested that small size filters can improve the performance of the CNNs. Based on these findings, VGG replaced the 11x11 and 5x5 filters with a stack of 3x3 filters

layer and experimentally demonstrated that concurrent placement of small size (3x3) filters could induce the effect of the large size filter (5x5 and 7x7). The use of the small size filters provides an additional benefit of low computational complexity by reducing the number of parameters. These findings set a new trend in research to work with smaller size filters in CNN. VGG regulates the complexity of a network by placing 1x1 convolutions in between the convolutional layers, which, besides, learn a linear combination of the resultant feature-maps. For the tuning of the network, max-pooling is placed after the convolutional layer, while padding was performed to maintain the spatial resolution. VGG showed good results both for image classification and localization problems.

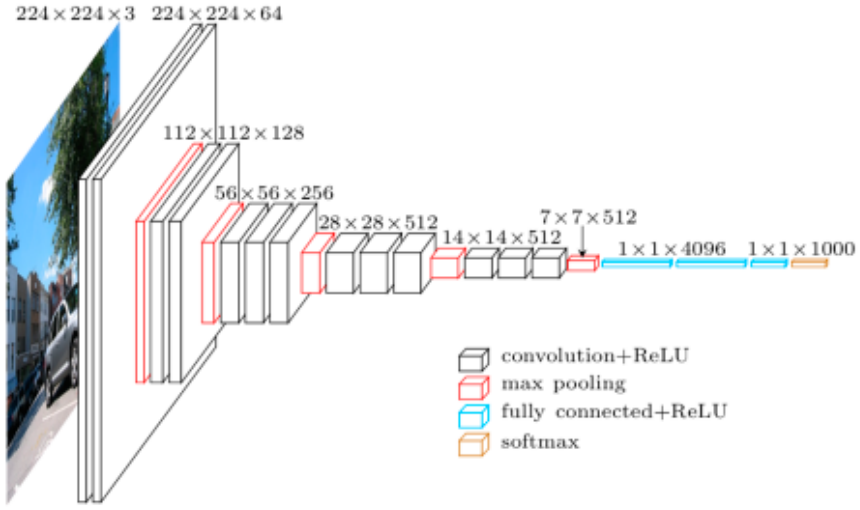


Fig. 3.7 Microarchitecture of VGG-16.

VGG was at 2nd place in the 2014-ILSVRC competition but, got fame due to its simplicity, homogenous topology, and increased depth. The main limitation associated with VGG was the use of 138 million parameters, which make it computationally expensive and difficult to deploy it on low resource systems.

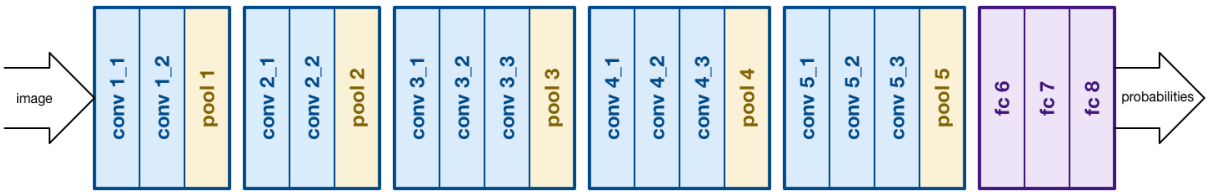


Fig. 3.8 An Example of VGGNet (2014)

3.5.5 *Transfer learning for deep learning*

In the early days of AI, a common problem was the lack of general intelligence. Models could be built to do things like play checkers, but the knowledge encapsulated in that model was restricted to that domain. This problem is being explored today under the name transfer learning with the goal of building a model that can be applied to multiple related problem areas.

Since the introduction of deep learning, there's been a renewed interest in neural networks for a range of applications. Deep learning has solved problems viewed as impossible not more than a decade ago. But deep learning neural networks require large clusters of compute servers, large amounts of training data, and a large amount of time to train the deep neural network.

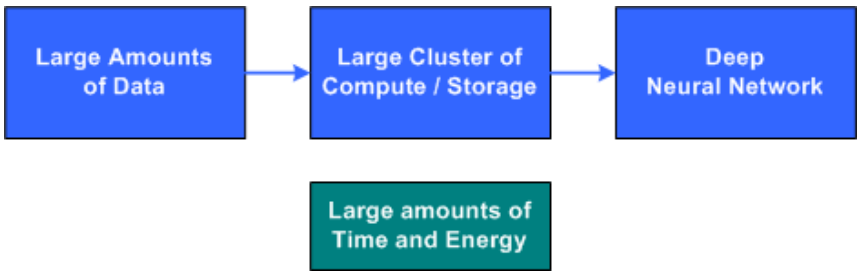


Fig. 3.9 The problem with deep learning

The deep neural network is represented by tens of millions of weights that connect the many layers of neurons of the networks together. These weights (typically real values) are adjusted during the training process and applied to inputs (including inputs from intermediary layers) to feed forward

to an output classification. The basic idea of transfer learning is then to start with a deep learning network that is pre-initialized from training of a similar problem. Using this network, a smaller duration of training is required for the new, but related, problem.

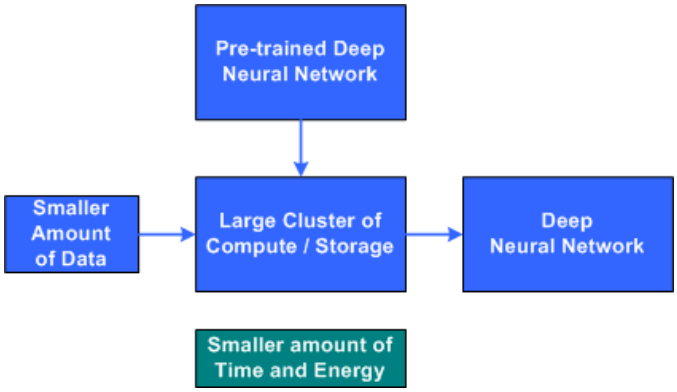


Fig. 3.10 Transfer learning with a pre-trained network

Transfer learning is the method of starting with a pre-trained model and training it for a new related problem domain. The pre-trained network serves as transferred knowledge to be applied in another domain. But there are numerous options that can be used, including feature transfer and fine-tuning (which depend upon the similarity of the problems at hand), in addition to freezing certain layers of the network and retraining others.

3.5.5.1 Advantage of Transfer Learning

Deep learning requires a tremendous amount of computing power. High performance graphical processing units (GPUs) are ideal because they can handle a large volume of calculations in multiple cores with copious memory available. However, managing multiple GPUs on-premises can create a large demand on internal resources and be incredibly costly to scale.

Transfer learning is an important piece of many deep learning applications now and in the future. This is predominantly due to the scale of training production deep learning systems; they’re huge and require significant resources.

A recent paper from the University of Amherst found that in a production deep learning application focused on natural language processing, there were more than 200 million weights required to be trained with the available training data. Training a network of this size with a network of graphical processing units (GPUs) emitted the same amount of CO2 as five average U.S. vehicles over their individual lifetimes. The energy consumed in training the network also paralleled two roundtrip flights between New York and San Francisco (200 passengers each).

Given this cost, it's important to use these resources in the most efficient way possible. Transfer learning is an opportunistic way of reducing machine learning model training to be a better steward of our resources.

3.5.6 *Deep Learning Applications*

Real-world deep learning applications are a part of our daily lives, but in most cases, they are so well-integrated into products and services that users are unaware of the complex data processing that is taking place in the background. Some of these examples include the following:

3.5.6.1 Law Enforcement

Deep learning algorithms can analyse and learn from transactional data to identify dangerous patterns that indicate possible fraudulent or criminal activity. Speech recognition, computer vision, and other deep learning applications can improve the efficiency and effectiveness of investigative analysis by extracting patterns and evidence from sound and video recordings, images, and documents.

3.5.6.2 Financial Services

Financial institutions regularly use predictive analytics to drive algorithmic trading of stocks, assess business risks for loan approvals, detect fraud, and help manage credit and investment portfolios for clients.

3.5.6.3 Customer Service

Many organizations incorporate deep learning technology into their customer service processes. Chatbots—used in a variety of applications, services, and customer service portals—are a straightforward form of AI. Traditional chatbots use natural language and even visual recognition, commonly found in call centre-like menus. However, more sophisticated chatbot solutions attempt to determine, through learning, if there are multiple responses to ambiguous questions. Based on the responses it receives, the chatbot then tries to answer these questions directly or route the conversation to a human user. Virtual assistants like Apple's Siri, Amazon Alexa, or Google Assistant extends the idea of a chatbot by enabling speech recognition functionality. This creates a new method to engage users in a personalized way.

3.5.6.4 Healthcare

The healthcare industry has benefited greatly from deep learning capabilities ever since the digitization of hospital records and images. Image recognition applications can support medical imaging specialists and radiologists, helping them analyse and assess more images in less time.

3.6 Deep Learning Frameworks and Tools

The growth of machine learning and AI has enabled organizations to provide smart solutions and predictive personalization to their customers. However, not all organizations can implement machine learning and AI for their processes due to various reasons.

This is where the services of various deep learning frameworks come in. These are interfaces, libraries, or tools, which are generally open-source that people with little to no knowledge of machine learning and AI can easily integrate. Deep learning frameworks can help you upload data and train a

deep learning model that would lead to accurate and intuitive predictive analysis. Some of the important frameworks that used nowadays are (TensorFlow, Keras, and PyTorch)

1. **TensorFlow:** It is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. TensorFlow is a symbolic math library based on dataflow and differentiable programming. This framework was developed by Google's Brain team.
2. **Keras:** It is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. Up until version 2.3, Keras supported multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, Theano, and PlaidML
3. **PyTorch:** PyTorch is an open-source machine learning library based on the Torch library, used for applications such as computer vision and natural language processing, primarily developed by Facebook's AI Research lab. It is free and open-source software released under the Modified BSD license.

3.7 Image Feature Extraction

In computer vision and image processing, a feature is a piece of information about the content of an image; typically, about whether a certain region of the image has certain properties. Features may be specific structures in the image such as points, edges or objects. Features may also be the result of a general neighbourhood operation or feature detection applied to the image. Other examples of features are related to motion in image sequences, or to shapes defined in terms of curves or boundaries between different image regions

Feature extraction is a part of the dimensionality reduction process, in which, an initial set of the raw data is divided and reduced to more manageable groups. So, when it is needed to be processed it will be easier. The most important characteristic of these large data sets is that they have a large number of variables. These variables require a lot of computing resources to process them. thus, feature extraction helps to get the best feature from those big data sets by select and combine variables into features, therefore, effectively reducing the amount of data. These features are easy to process, but still able to describe the actual data set with the accuracy and originality.

3.8 Image Classification

Image classification is the process of a computer analysing an image and identifying which "class" it belongs to. (Or a probability that the image belongs to a ‘class.') A class is simply a label, such as "car," "animal," "building," and so on.

Raw pixel data was used to classify images in the beginnings. Computers would deconstruct images into individual pixels. The issue is that two images of the same object might appear to be quite different. They can consist of a variety of backgrounds, angles, and positions. Computers found it difficult to accurately ‘recognize' and categorize images.

3.9 Project Tools and Dataset

There are multiple tools that were used in this project and they are divided into 4 types (IDE, Programming Language, Frameworks, Datasets).

3.9.1 Programming Language

Deep learning has many approaches when it comes to implementation, there are many programming languages and tools that can be used to

implement AI algorithms, some of them are (Python, R, MATLAB, Java, C++, …Etc).

In this project "Python Programming Language" is used for implementing deep learning algorithms.

3.9.1.1 Python (Programming Language)

Python is an interpreted high-level general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation.

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by meta programming and metaobjects (magic methods)).

3.9.1.1.1 Advantages of Using Python for Deep Learning

Python offers a vast choice of libraries for AI development, which contain base-level items that save coding time. These libraries also make it easy to access, handle, and transform data. Some of the important libraries like (Keras and TensorFlow) were mentioned earlier in this chapter.

It is famous for its compact, readable code, and is practically unmatched with regards to usability, especially for new developers. This has made it a preferred language for AI and deep learning.

Python's flexibility decreases the possibility of errors by empowering software engineers to take charge of the issue and operate in a comfortable environment.

Python is an open-source programming language with a wealth of resources and solid documentation. It also includes a large and productive

network of developers that can provide guidance and assistance at any stage of the development process.

Python programming language, as well as its many useful libraries and tools, is completely free.

3.9.2 *Integrated Development Environment (IDE)*

It is a software for building applications that combines common developer tools into a single graphical user interface (GUI). An IDE typically consists of:

- Source code editor: A text editor that can assist in writing software code with features such as syntax highlighting with visual cues, providing language specific auto-completion, and checking for bugs as code is being written.
- Local build automation: Utilities that automate simple, repeatable tasks as part of creating a local build of the software for use by the developer, like compiling computer source code into binary code, packaging binary code, and running automated tests.
- Debugger: A program for testing other programs that can graphically display the location of a bug in the original code.

There are many types of IDEs that can be used for deep learning like (PyCharm, Anaconda, Spyder, and Atom) in this project the IDE that were used is “PyCharm”.

3.9.2.1 PyCharm IDE

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes).

PyCharm Professional Edition was used in the project, it is paid software. The organization has made this edition free for college students, therefore a request was sent to them including the students' academic emails as verification, and the request was immediately accepted as well as a free license was issued.

3.9.3 *Frameworks and Libraries*

There are multiple frameworks were included in the project each one has a different purpose in addition to the main frameworks (Keras and TensorFlow) Pandas and NumPy are used too.

- Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license.
- NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

3.9.4 *Datasets*

A data set is a collection of data. In the case of tabular data, a data set corresponds to one or more database tables, where every column of a table represents a particular variable, and each row corresponds to a given record of the data set in question

The publicly available datasets for ethnicity recognition, summarized in Table 3.1

Table 3.1 Available Datasets of Face Images with Ethnicity Groups

Dataset	**Images (People)**	**Ethnicity Groups**
FERET	14,126 (1199)	Caucasian, Asian. Oriental African
JAFFE	2130 (10)	Japanese
IFDB	3600 (616)	Iranian
CASPEAL	30,900 (1040)	Chinese
MORPH-II	55,134 (13,618)	African, European, Asian, Hispanic, Others
FEI	2800 (200)	Brazilian
PubFig	58,797 (200)	Asian, Caucasian, African American, Indian
CUN	112,000 (1120)	Chinese
HUDA	N/A	Saudi Arabia
EGA	72,266 (469)	African American, Asian, Caucasian, Indian, Latin
CAFE	1192 (154)	Caucasian, East Asians, Pacific Region
LFWA+	13,233 (5749)	White, Black, Asian
UTKFace	20,000 (N/A)	White, Black, Asian, Indian, Others
FairFace	108,192 (N/A)	White, Black, East Asian, Southeast Asian, Indian, Middle Eastern, Latin

Each dataset has its own category, the main disadvantage of the datasets is the number of people in each one. In deep learning for recognition applications there must be a large set of data for the model to work properly and produce fine results. As shown in the table most of the datasets are not enough for such purposes.

The deep learning model of this project depended on FERET dataset. The reason for choosing this one because it’s an official dataset with

complete information about each individual, and it has a diverse of ethnicities.

Some of the datasets like (UTKFace, PubFig, FairFace… Etc) can't be used for the project because they have random images and they are not classified into groups. MORPH-II dataset can be a good choice but it is not for free and has an expensive price. Because of the mentioned difficulties "FERET" was the perfect choice compared to the rest datasets.

3.9.4.1 FERET Dataset

The Facial Recognition Technology (FERET) database is a dataset used for facial recognition system evaluation as part of the Face Recognition Technology (FERET) program. The FERET program set out to establish a large database of facial images that was gathered independently from the algorithm developers. Dr. Harry Wechsler at George Mason University was selected to direct the collection of this database. The database collection was a collaborative effort between Dr. Wechsler and Dr. Phillips. The images were collected in a semi-controlled environment. To maintain a degree of consistency throughout the database, the same physical setup was used in each photography session. Because the equipment had to be reassembled for each session, there was some minor variation in images collected on different dates.

The FERET database was collected in 15 sessions between August 1993 and July 1996. The database contains 1564 sets of images for a total of 14,126 images that includes 1199 individuals and 365 duplicate sets of images. A duplicate set is a second set of images of a person already in the database and was usually taken on a different day.

For some individuals, over two years had elapsed between their first and last sittings, with some subjects being photographed multiple times. This time lapse was important because it enabled researchers to study, for the first

time, changes in a subject's appearance that occur over a year. The FERET database has been used by more than 460 research groups and is managed by the National Institute of Standards and Technology (NIST).

3.9.4.2 Local Dataset

It is a small dataset that was collected from students at the University of Kerbala to be included in the project dataset for training and testing. The data was collected by creating a "Google Form" to acquire the following information: "student name," "student age," and "student facial photo." Unfortunately, due to privacy concerns, only 15 students have volunteered for the dataset.

3.10 Proposed Work

The proposed solution for the problem will be done by using deep learning with python and the proposed work for achieving the objective is as the following:

1. Collect Dataset (Project Dataset)
2. Cropping Facial Landmarks
3. Feature Extraction by Deep Learning
4. Training and Classification

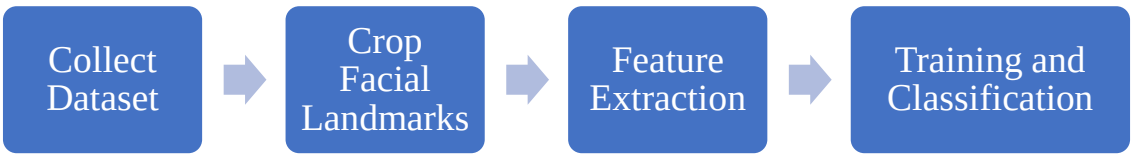


Fig. 3.11 Proposed Work

In the next sections every step will be explained as well as the used code.

3.10.1 *Project Dataset*

The project dataset is divided into three parts. The first is made up of exclusively males' images, whereas the second is made up of images from both genders, and the third one is made up of females' images. The Male-Dataset has 160 images that are divided into four ethnicity groups which are "Asian", "Asian-Middle Eastern", "African-American", and "White". Each group has 40 photos, by allocating 70% of the images for training and 30% for testing, yielding in 28 images for training and 12 for testing. The Male-Female-Dataset contains 240 images divided into three groups: "Asian," "African-American," and "White." Each group has 80 images, 40 for males and 40 for females, all in one folder. 70% of the images were allocated for training (56 images out of 80) and the remaining 30% was given for testing (24 images). The Female-Dataset contains 120 images divided into 3 groups: "Asian," "African-American," and "White.", 70% of the dataset is used for training and 30% is used for testing, yielding in 28 images for each group of training and 12 Images for each group in testing.

The main disadvantage of this dataset is the limited number of individuals available. For a deep learning application, a dataset of 160 images or 240 images is considered to be extremely small and this is due to the limited number of people in the major dataset.

3.10.2 *Cropping Facial Landmarks*

In this stage of the project every image in the dataset will be cropped into 5 parts, each part represents a different part of the human face. The 5 parts are as following; (Face Border, Eyes, Eyebrows, Nose, and Mouth) Fig. 3.10 shows the results of the cropping step.

Fig. 3.12 A cropped sample of The Dataset

The cropping step generates five datasets that will be used separately for the next stage, with the results of each being compared later in the results chapter.

3.10.3 *Feature Extraction by Deep Learning*

This section will cover the coding part of the project. Before explaining the feature extraction code, there are some parts of the code must me mentioned in the first place. Before explaining the code, it must be mentioned that this code is open-source and was edited to suit the project purpose.

```
from os import listdir
from os.path import isfile, join
import pandas as pd
import numpy as np
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Convolution2D, MaxPooling2D, Flatten, Dense, Dropout,
GlobalAveragePooling2D
from keras.applications.vgg16 import VGG16

train_dir = r"./dataset/train"  # training data directory
valid_dir = r"./dataset/test"   # validation data directory

img_width, img_height = 224, 224  # Default input size for VGG16

# Instantiate convolutional base
conv_base = VGG16(weights='imagenet',
            include_top=False,
            input_shape=(img_width, img_height, 3))
```

```
# Show architecture
conv_base.summary()
```

In this part of the code above, the datasets directories were initialised and also the "VGG16 default input size". The convolutional base was initialised too. after running the code, the convolutional base will be initialised according to the type of data that were given, and the output of it will be passed to the feature extraction function, which it was (7,7,512).

```
# Extract features
import os, shutil
from keras.preprocessing.image import ImageDataGenerator

datagen = ImageDataGenerator(rescale=1. / 255)
batch_size = 8

def extract_features(directory, sample_count):
    features = np.zeros(shape=(sample_count, 7, 7, 512))  # Must be equal to the output of
the convolutional base
    labels = np.zeros(shape=(sample_count, 4))
    # Preprocess data
    generator = datagen.flow_from_directory(directory,
                                            target_size=(img_width, img_height),
                                            batch_size=batch_size,
                                            class_mode='categorical')
    # Pass data through convolutional base
    i = 0
    for inputs_batch, labels_batch in generator:
        features_batch = conv_base.predict(inputs_batch)
        features[i * batch_size: (i + 1) * batch_size] = features_batch
        labels[i * batch_size: (i + 1) * batch_size] = labels_batch
        i += 1
        if i * batch_size >= sample_count:
            break
    return features, labels
```

The second part of this code is about the feature extraction function. The batch size is set to be "8" depending on the previous output, the convolutional base will be passed which its (7,7,512) with the sample count "4" this represents the number of ethnicity groups in the dataset. The convolutional base output can be different for every dataset, it can be changed according to the input dataset. Also, in this code the classification mode is set to be "categorical" because we have multiple categories. The last

part of this block is a for loop that is responsible to pass the input data and convolutional base and return features and labels.

3.10.4 *Training*

This section will cover the training step of the project. In the following code, this important part defines training. the selected "epochs" number is 2000, The number of epochs is the number of complete passes through the training dataset. This number can be set from zero to infinity and it should be as high as possible then terminate the training when validation error start increasing. In this project choosing the right epochs was a bit difficult due to shortage in the main dataset, the number of epochs is not that significant. more important is the validation and training error. As long as these two errors keeps dropping, training should continue.

```
train_features, train_labels = extract_features(train_dir, 112)  # Agree with our small
dataset size
validation_features, validation_labels = extract_features(valid_dir, 48)
# test_features, test_labels = extract_features(test_dir, test_size)
train_labels
# print(len(train_labels))
epochs = 2000

model = Sequential()
model.add(GlobalAveragePooling2D(input_shape=(7, 7, 512)))
model.add(Dense(4, activation='softmax'))
model.summary()
from keras.callbacks import ModelCheckpoint

checkpoint = ModelCheckpoint('model-{epoch:03d}-{acc:03f}-{val_acc:03f}.h5', verbose=1,
monitor='val_loss',
                    save_best_only=True, mode='auto')

# Compile model
from keras.optimizers import Adam

model.compile(optimizer=Adam(),
          loss='categorical_crossentropy',
          metrics=['acc'])

# Train model
history = model.fit(train_features, train_labels,
                epochs=epochs,
                batch_size=batch_size,
```

```
    callbacks=[checkpoint],
    validation_data=(validation_features, validation_labels))
```

The problem here there was an overfitting in the results thus the epochs number should be decreased but in the same time the results are not fine when the number of epochs is less than 2000, therefore this number was the best possible choice. The other part of the code is about training accuracy and validation accuracy depending on the previous settings.

After finishing the training process, the following part of code which it is the last part is responsible of plotting the result which will be discussed in the next chapter.

```
# Plot results
import matplotlib.pyplot as plt

acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(1, len(acc) + 1)

plt.plot(epochs, acc, 'g', label='Training accuracy')
plt.plot(epochs, val_acc, 'r', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.legend()

plt.figure()

plt.plot(epochs, loss, 'g', label='Training loss')
plt.plot(epochs, val_loss, 'r', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()

plt.show()
```

Chapter Four: Experimental Results

4.1 Introduction

This chapter will cover the results of the project's experiments.

The project has three experiments, the first one is divided into 5 experiments, all of them are done using the "Male-Dataset". The second one is done by using Male-Female-Dataset, and the last one is done by using Female-Dataset. All of the three datasets were mentioned earlier in chapter3.

4.2 Experiment 1 by using Male-Dataset

This experiment was done by cropping the dataset into 5 parts; (face, eyes, eyebrows, nose, mouth). As mentioned in chapter 1 the project aims to find which part of the human face is more effective on human race change, for this reason the Male-Dataset was cropped into 5 parts, those parts were considered as different datasets, each one was used alone in the project code, and later each dataset will have its own result to be compared later to achieve the project objective.

4.2.1 *Results of Exp. 1*

The results of the first experiment are shown in Table 4.1

Table 4.1 Experiment 1 Results

Type	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
Face	0.0166	1	0.9386	0.7708
Eyes	0.1011	0.9826	0.6817	0.8
Mouth	0.4598	0.8479	1.168	0.55
Eyebrows	0.451	0.8882	0.8816	0.725

Nose	0.4187	0.9301	1.2271	0.425

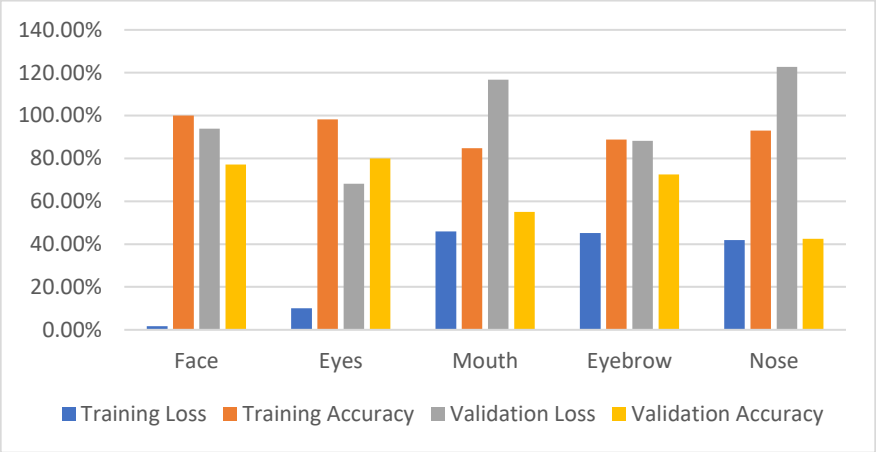


Fig. 4.1 Exp. 1 Results Chart

After making this experiment it is possible to observe that the "eyes" has the best validation accuracy "80%" compared to other parts, and the "nose" has the worst validation accuracy "42%". Also, it is possible to notice the overfitting in the results curves below and that is due to the shortage in the dataset which were discussed in chapter 3.

4.2.1.1 Face Results

The face has the second-best accuracy compared to the rest of the experiments and it has achieved a training accuracy of 100%. The validation loss has caused an overfitting, by making a test with a smaller number of epochs can help reduce the overfitting but in the same time the validation accuracy won't be stable. More details in Fig 4.2 and 4.3.

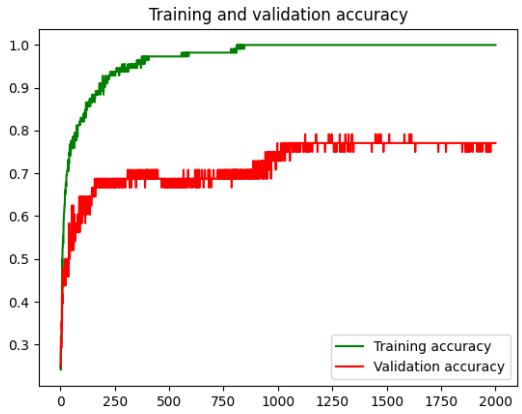


Fig. 4.2 Training and Validation Accuracy of "Face"

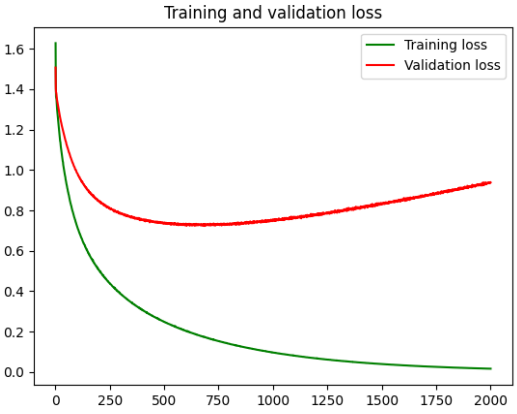


Fig. 4.3 Training and Validation Loss of "Face"

4.2.1.2 Eyes Results

As shown in Fig. 4.4 and 4.5, it is clear to observe that the eyes are the most effective part in deciding the human race. It has achieved a validation accuracy of 80% which makes it in the top of the table. In addition, it has the less validation loss compared to the rest of the objects.

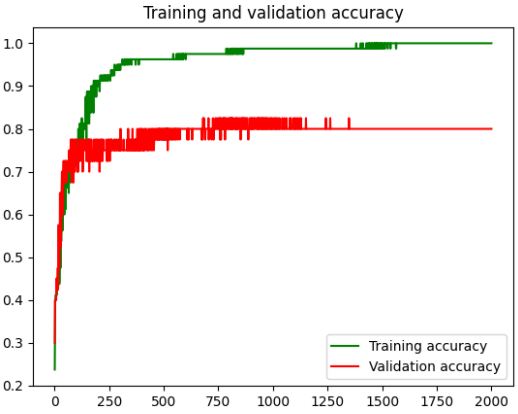


Fig. 4.4 Training and Validation Accuracy of "Eyes"

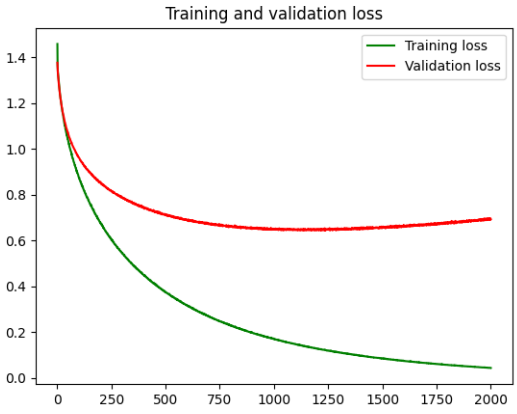


Fig. 4.5 Training and Validation Loss of "Eyes"

4.2.1.3 Mouth Results

The mouth results had a validation accuracy of 55% with the greatest validation loss which it is 116%. Those results mean that the mouth has a little chance in predicting human race. Also, the mouth curves are not stable as shown in Fig. 4.6. and Fig. 4.7 showing a huge overfitting.

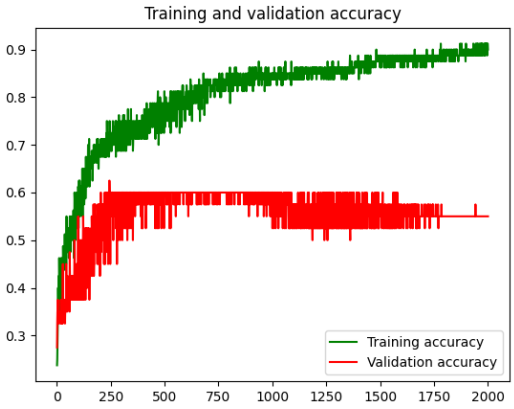


Fig. 4.6 Training and Validation Accuracy of "Mouth"

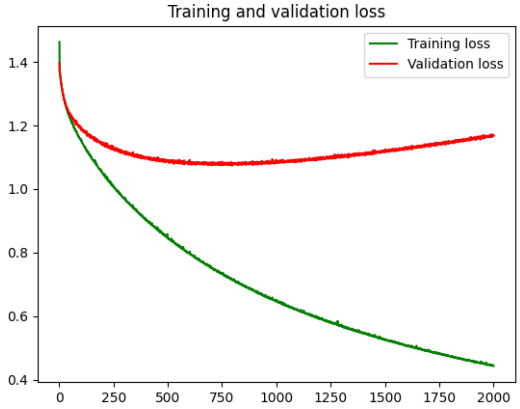


Fig. 4.7 Training and Validation Loss of "Mouth"

4.2.1.4 Eyebrows Results

The eyebrows got a good validation accuracy "72%" but in the same time it has a high validation loss "80%" leading to an overfitting in the curve, And still not comparable with the "eyes accuracy". Another drawback here, the training accuracy is not stable enough as well as the validation accuracy.

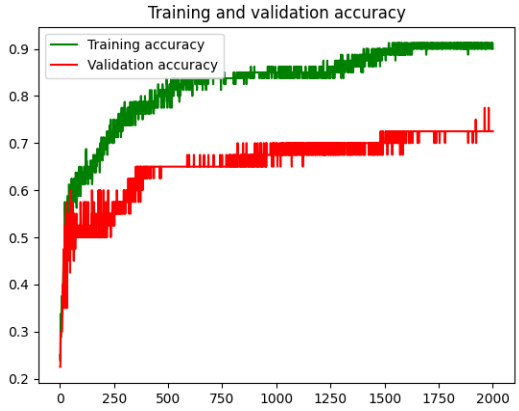


Fig 4.8 Training and Validation Accuracy of "Eyebrows"

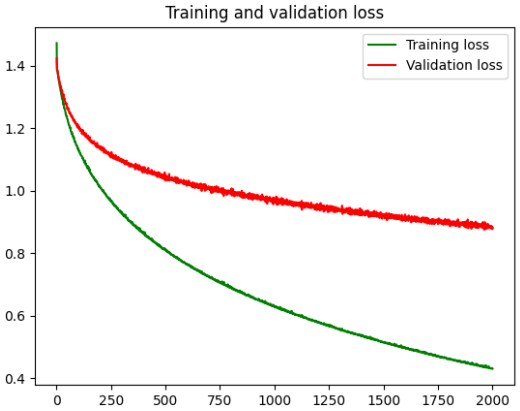


Fig 4.9 Training and Validation Loss of "Eyebrows"

4.2.1.5 Nose Results

Nose results are the most disappointing it was expected that the nose can have a high accuracy but the experiment has proved the opposite. A validation accuracy of 42% with an overfitting from the beginning of the training. As shown in the curves of Fig. 4.10 and 4.11 the validation accuracy is so distorted and the training accuracy too.

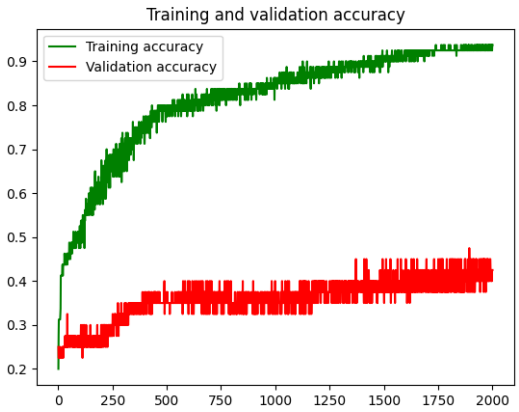


Fig. 4.10 Training and Validation Accuracy of "Nose"

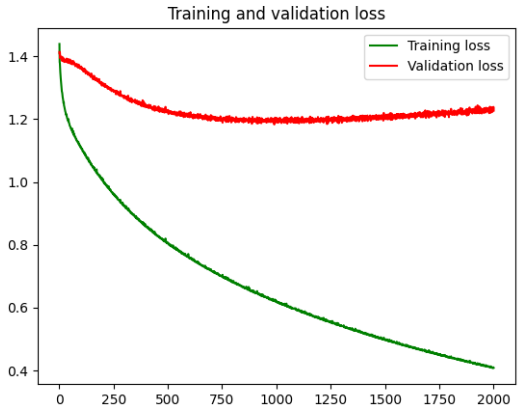


Fig. 4.11 Training and Validation Accuracy of "Nose"

4.3 Experiment 2 by using Male-Female-Dataset

In this experiment the dataset that contains 240 of female and male images were used, divided into 3 groups. Table 4.2 shows the results of the experiment.

Table 4.2 Experiment 2 Results

Type	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
Face-Male-Female	0.0028	1	0.5383	0.8542

Fig. 4.12 shows validation accuracy compared to the validation loss, with accuracy of 85% and loss 53% this makes it the best accuracy among the experiments.

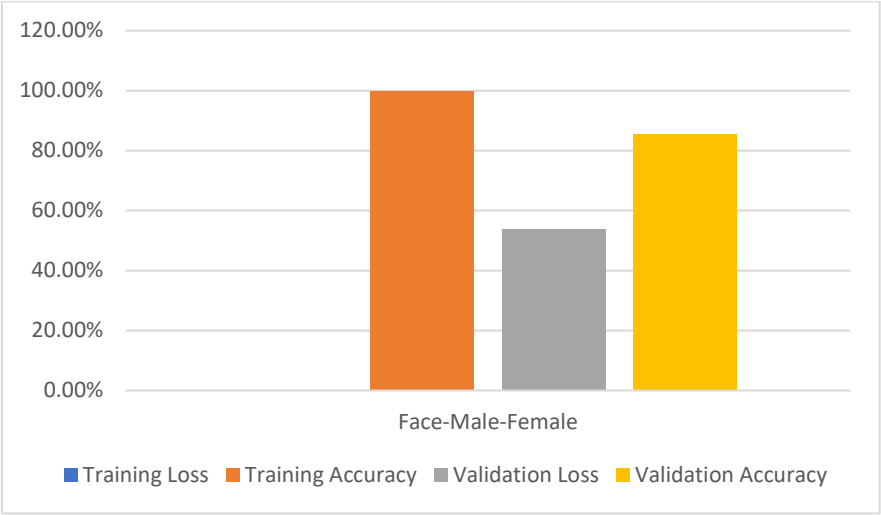


Fig. 4.12 Exp. 2 Results Chart

In this experiment the features of both genders were used equally, the validation accuracy and training accuracy curves were stable earlier and the overfitting is not huge compared to other experiments. As shown in Fig. 4.13 and 4.14.

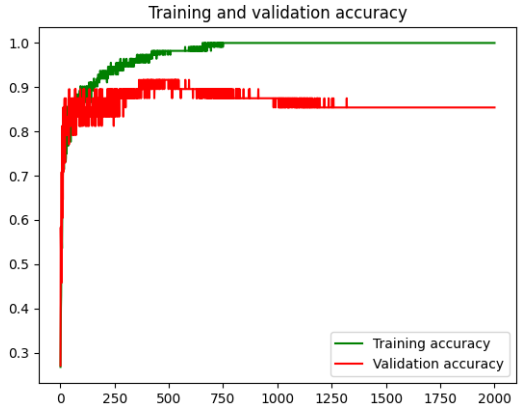


Fig. 4.13 Training and Validation Accuracy of "Face-Male-Female"

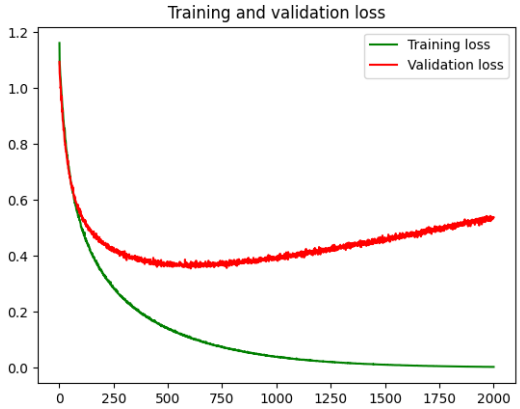


Fig. 4.14 Training and Validation Loss of "Face-Male-Female"

4.4 Experiment 3 by using Female-Dataset

This is the final experiment in the project, it is done by using a dataset that contains 120 of females' images only divided into 3 groups. The results are shown in Table 4.3.

Table 4.3 Experiment 3 Results

Type	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
Face-Female	0.0012	1	1.0975	0.7708

The validation accuracy of this experiment can be considered good but in the same time the validation loss is extremely high. As shown in the curve of Fig. 4.16 causing a huge overfitting. In addition, the validation accuracy was not stable enough as shown in Fig. 4.15. the huge overfitting can be explained, this dataset is smaller in size compared to other datasets this reason can be a great factor to make an overfitting, it is also not possible to reduce the epochs number because the validation accuracy was not stable yet, thus. These results cannot be improved any further.

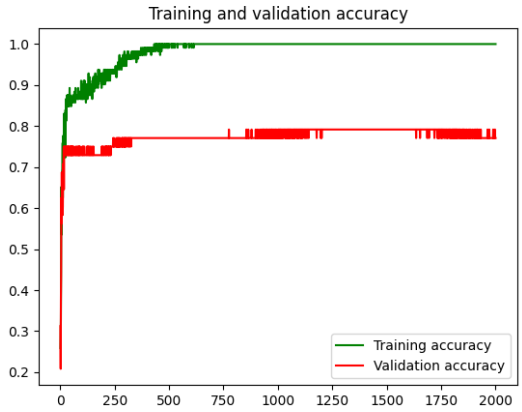


Fig. 4.15 Training and Validation Accuracy of "Face-Female"

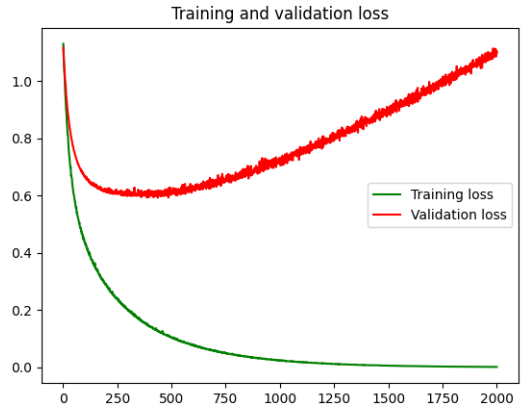
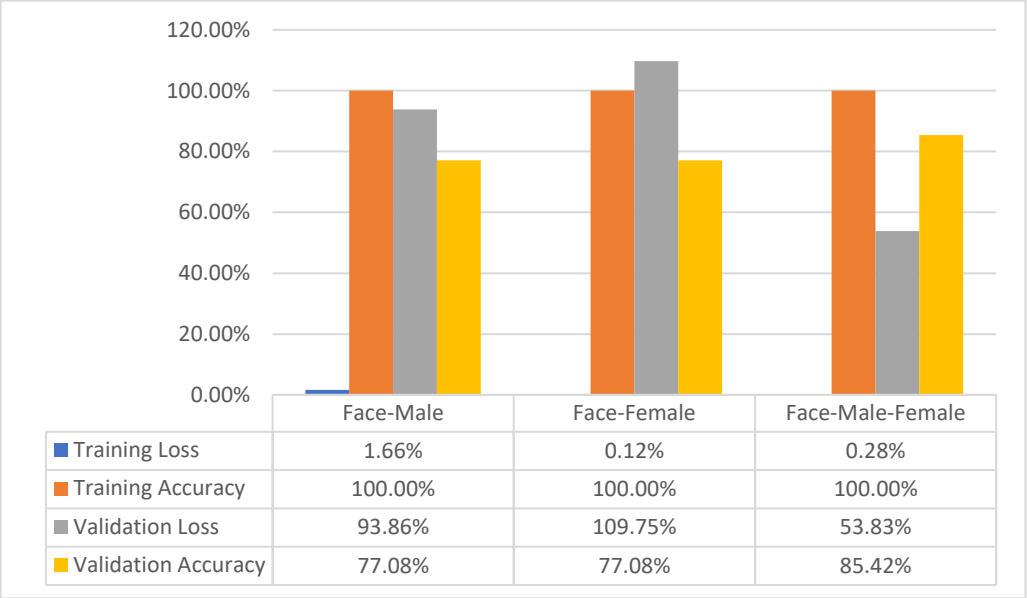


Fig. 4.16 Training and Validation Loss of "Face-Female"

The Fig. 4.17 shows a comparison between three similar types. It is clear that the experiment of "face-female" and "face-Male" both have the same validation accuracy 77.08% and approximately same validation loss, while the experiment of both genders has an extremely better validation

accuracy 85.42% and validation loss of 53.83% this makes it the experiment with the best results.



	Face-Male	Face-Female	Face-Male-Female
Training Loss	1.66%	0.12%	0.28%
Training Accuracy	100.00%	100.00%	100.00%
Validation Loss	93.86%	109.75%	53.83%
Validation Accuracy	77.08%	77.08%	85.42%

Fig. 4.17 A comparison between "Face-Male", "Face-Female", "Face-Male-Female"

Chapter Five: Conclusion and Future Work

5.1 Conclusion

The project has achieved two of its three objectives. Unfortunately collecting a dataset of locals couldn't be achieved well and that is due to the privacy concerns that people have, as mentioned in chapter 3.

The research main objective from the beginning was to identify the most effective region of the face for distinguishing the human race. According to the results of the experiments, it is possible to state that the eyes have the greatest effect, with an accuracy of 80%, which is the best among the rest parts, and even better than the entire face. Of course, there was a problem with overfitting in the curves, which was caused by a lack of images in the datasets. This problem can be resolved by providing a dataset with more images, which would result in more accurate outcomes. The project can now be used as an automatic race classification system, it is possible to change the current datasets and choose a different one to make a test.

5.2 Future Work

Future work can be summarised by repeating the same procedures but using only females' dataset to find if there is a difference between the female face parts and the male face parts in achieving the same objective.

References

[1] T. Vo, T. Nguyen, and C. T. Le, “Race recognition using deep convolutional neural networks,” *Symmetry*, vol. 10, no. 11, pp. 1–15, 2018, doi: 10.3390/sym10110564.

[2] S. M. Mansoor Roomi, S. L. Virasundarii, S. Selvamegala, S. Jeevanandham, and D. Hariharasudhan, “Race classification based on facial features,” *Proceedings - 2011 3rd National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics, NCVPRIPG 2011*, pp. 54–57, 2011, doi: 10.1109/NCVPRIPG.2011.19.

[3] H. Chen, M. Gao, K. Ricanek, W. Xu, and B. Fang, “A Novel Race Classification Method Based on Periocular Features Fusion,” *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 31, no. 8, pp. 1–21, 2017, doi: 10.1142/S0218001417500264.

[4] S. Masood, S. Gupta, A. Wajid, S. Gupta, and M. Ahmed, “Prediction of human ethnicity from facial images using neural networks,” *Advances in Intelligent Systems and Computing*, vol. 542, pp. 217–226, 2018, doi: 10.1007/978-981-10-3223-3_20.

[5] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, “A survey of the recent architectures of deep convolutional neural networks,” *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5455–5516, 2020, doi: 10.1007/s10462-020-09825-6.

[6] A. Swaminathan, M. Chaba, D. K. Sharma, and Y. Chaba, “Gender Classification using Facial Embeddings: A Novel Approach,” *Procedia Computer Science*, vol. 167, no. 2019, pp. 2634–2642, 2020, doi: 10.1016/j.procs.2020.03.342.

[7] C. Christy, S. Arivalagan, and P. Sudhakar, “Human Face Recognition and Identifying Ethnicity Using Machine Learning,” vol. 6, no. 5, pp. 211–215, 2019.

[8] A. Greco, G. Percannella, M. Vento, and V. Vigilante, “Benchmarking deep network architectures for ethnicity recognition using a new large face dataset,” *Machine Vision and Applications*, vol. 31, no. 7–8, pp. 1–13, 2020, doi: 10.1007/s00138-020-01123-z.

[9] "AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the Difference?." IBM, 27 May. 2020, www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks.

[10] "What are Neural Networks?" IBM, 17 Aug. 2020, www.ibm.com/cloud/learn/neural-networks.

[11] "What are convolutional neural networks?" IBM, 20 Oct. 2020, www.ibm.com/cloud/learn/convolutional-neural-networks.

[12] "What is deep learning?" IBM, 1 May 2020, www.ibm.com/cloud/learn/deep-learning.

[13] "What is machine learning?" IBM, 15 July 2020, www.ibm.com/cloud/learn/machine-learning#toc-machine-le-SzgJbkmk.

[14] "What is artificial intelligence?" IBM, 3 June 2020, www.ibm.com/cloud/learn/what-is-artificial-intelligence.

[16] "Face Recognition Technology (FERET)" NIST, July 13, 2017, www.nist.gov/programs-projects/face-recognition-technology-feret.

[17] Donges, Niklas. "What is transfer learning? Exploring the popular deep learning approach" Built In, 16 June 2019, www.builtin.com/data-science/transfer-learning.

[18] Das, Siddharth. "CNN Architectures: LeNet, AlexNet, VGG, GoogLeNet, ResNet and more" Medium, 16 Nov. 2017, www.medium.com/analytics-vidhya/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5.