Faculty of engineering Helwan university

Computer & Engineering & Systems Department

Graduation project

Academic Year: 4th year

Project Title: Blind Deaf System (BDS)

Supervisor: DR. Mahmoud Zaki

Prepared by:

Seif Eldin khaled Seif Elnasr

Omar Abdelmohsen Hussien

Nada Mohamed Sayed

Mennat-Allah Hesham Sayed

Mennat-Allah Ahmed Farrag

Table of Contents

Abbreviations

WHO	World Health Organization
ML	Machine Learning
DL	Deep Learning
AI	Artificial Intelligence
RL	Reinforcement Learning
ANN	Artificial Neural Network
ReLU	Rectified Linear Unit Function
CNN/ConvNet	Convolutional Neural Networks
RCNN	Region Based Convolutional Neural Networks
Fast RCNN	Fast Region Based Convolutional Neural Networks
SVM	Support Vector Machine
ROI	Region Of Interest
SSD	Single Shot detector
RPN	Region Proposal Network
IoU	Intersection-Over-Union
ASL	American Sign Language
RNN	Recurrent Neural Networks
LSTM	long short-term memory
ESL	Egyptian sign language

List of tables

List of figures

Chapter One

1 Introduction

1.1 History

The miscommunication between deaf people and normal people made a psychological gap so in 1892 electrical hearing aids were invented such as the ear trumpet, a funnel-shaped device which collects sound waves and leads them into the ear. Since then, technology has improved our hearing abilities with digital hearing aids and cochlear implants. In 1985 the cochlear implant was approved for people aged 18 and older. **[1]**

1.2 Problem statement

Over five per cent of the global population 432 million adults and 34 million children have disabling hearing loss. According to the World Health Organization, around 466 million people worldwide have disabling hearing loss and it is estimated that by 2050, the number will rise to over 900 million.

To satisfy the need for communication between sign language speakers and non-sign language speakers, people usually use text or a translator. Both methods arise some problems: In the first case, text conversations are not as comfortable as spoken ones, they go slower, and expressions are not visible. In the second case, the need of a person to communicate everything someone says can be expensive and eliminates privacy.

The recognition of signs with artificial vision could solve both problems. as this huge numbers and the only way of communication between blind and deaf that they will need third person with no disabilities. As we all see that everyone in this world have the right to communicatee with everyone. Communication with people is one of the biggest problems that faces deaf persons.

The only way that they can communicate with other is sign language. Normal people can’t understand sign language as it is not a traded language, and it is difficult to learn.

Likewise, in the case of communication between a blind person and a deaf person, communication via sign language will be impossible. We can't communicate with just the language, but we need some gestures to understand each other:

Sign Language Features:

Sign Language is used by Deaf people to communicate with each other. Sign Language is a full featured language (with lexicons, a grammar, etc) and it is the most advanced form of gestural communication.

Sign Composition:

Any manual sign can be broken up into four parameters. Each of these parameters is independent of each other, and is dynamic or static during a sign, Static gesture is used for alphabet and number representation, whereas dynamic gesture is used for specific concepts. Dynamic also includes words, sentences etc. Static gesture consists of poses of hand, whereas the latter include motion of hands, head, or both.

1. **The Handshape:** is defined by fingers and palm.

Figure 1: Examples of handshapes

2. **The Orientation:** is defined by two axes of the hand.
3. **The Movement:** is the hand trajectory (line, circle, curve, etc...).
4. **The Location:** is the hand position in relation to the body, The location is mainly used to express spatial and temporal information or relationship and the location granularity is according to needs. Each of the four parameters carries information and is part of the sign meaning

1.3 Problems that are facing the Deaf-Mute people

1.3.1 Social Effects:

- Reducing the miscommunication with normal/blind people.
- Reducing the bullying on deaf people.

1.3.2 Environmental Effects:

- Remote communication (COVID-19 effect).

1.3.3 Commercial Effects:

- Solve some educational problems.
- Reducing unemployment.

1.4 Facts about Deaf People

According to WHO, over 5% of the world's population – or 430 million people – require rehabilitation to address their 'disabling' hearing loss (432 million adults and 34 million children). It is estimated that by 2050 over 700 million people – or one in every ten people – will have disabling hearing loss. **[2]**

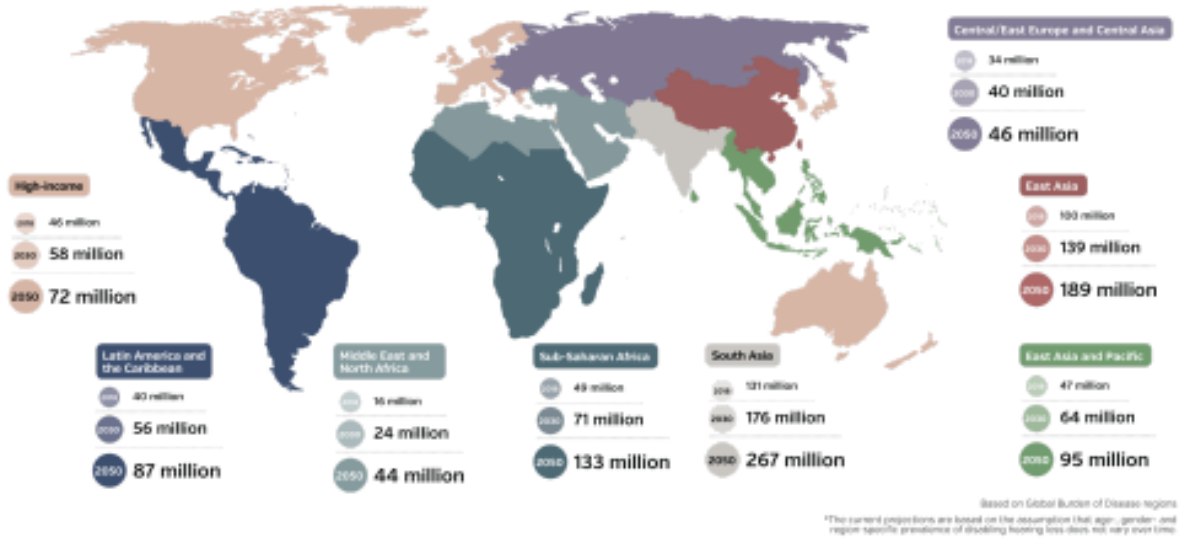


Figure 2: Projected number of people with hearing loss in different world regions until 2050

The map shows the current and projected number of people with hearing loss in different regions. Projections show that the number of people with disabling hearing loss will increase in all regions. **[3]**

1.5 Project Description

Blind-Deaf system comes the make communication between Blinds and Deaf easier. System will contain two parts:

- Glasses for Blind person.
- Mobile App for Deaf person.

First way of communication (Glasses): As Blind/normal person want to understand Deaf person that communicate with sign language here role of glasses comes. Glasses contain (camera, microcontroller, headphones), it will translate sign language into speech that Blind can hear it.

Second way of communication (Mobile Application): As Deaf person want to understand Blind/normal person that communicate with voice here role of Mobile App comes.it will translate voice into Video with sign language that Deaf can see.

1.6 Objectives and Goals

The aim of this project is to build and implement an AI model capable of recognizing sign language with computer vision in real time. The selected sign language is Egyptian Sign Language (ESL) because it is the most used among the Deaf community in Egypt and it is easily translated into spoken or written Arabic and to design other system that will take output of model and then use speaker of headphones to hear the word by normal person or blind person.

Goals:

1. Our project aims to remove the gap between deaf people and blind people or normal people.
2. Reducing the social effects that affect the Deaf-Dumb people.
3. Apply an effective way in translation for sign language with a very simple way.
4. Make a system using which deaf people can significantly communicate with all other people using their normal signs.
5. Minimum cost.

1.7 Overview

2 Machine learning: in this chapter we propose a brief introduction of machine learning showing its life cycle, uses and its various fields: supervised, unsupervised and reinforcement learning, and giving an introduction for each one of them.

3 Deep learning: This chapter shows the difference between machine learning and deep learning, The details of neural networks and the explanation of various deep learning algorithms such as CNN, Fast R-CNN.

4 MediaPipe: This chapter discuss the mediapipe definition, uses, and how to use.

5 Previous Solutions: This chapter discusses the previous solutions that were used before to help the deaf people.

6 Mobile Application: this chapter discusses why we used mobile app , what is android studio , what is java and why we used java programming language.

7 Hardware: This chapter contains details about hardware components used like Raspberry Pi and Raspberry Pi Camera.

8 Proposed Methods: This chapter discusses our solution and its diagrams and models used to translate sign to text then to speech, the chapter shows the flowcharts for each model Also shows the ways we used to translate speech to sign.

9 Results and Discussion: This chapter shows our results and discusses them.

10 Conclusion and Future Work: This chapter shows the whole conclusion and future work that can be done to improve the system.

11 References: This chapter provides the references that are used in our project.

Chapter Two

2 Machine learning

Machine learning is one modern innovation that has helped man enhance not only many industrial and professional processes but also advances everyday living from driving cars to translating speech.

2.1 ML Definition

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. **[4]**

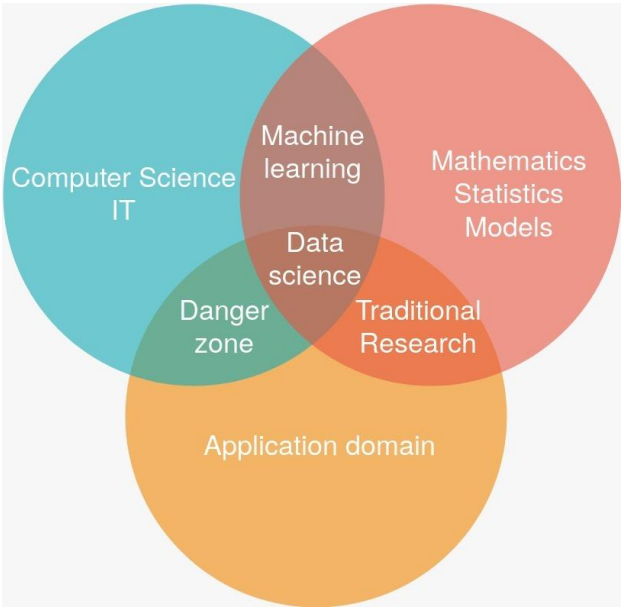


Figure 3:machine learning with statistics

2.2 ML life cycle

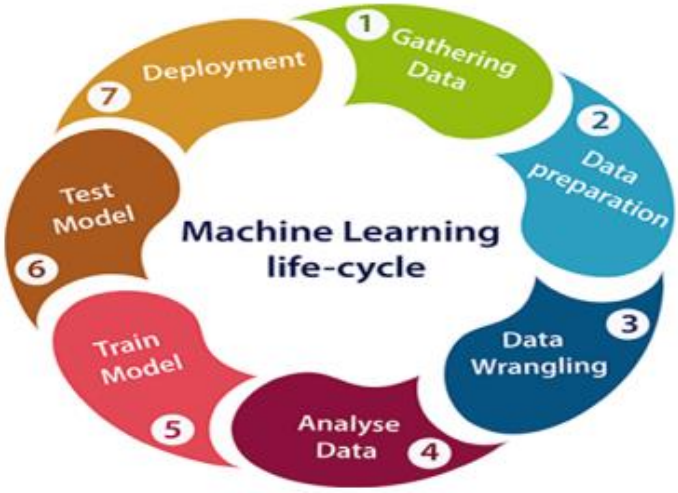


Figure 4: ML life cycle **[5]**

2.2.1 Gathering Data:

Data Gathering is the first step of the machine learning life cycle. The goal of this step is to identify and obtain all data-related problems. data can be collected from various sources such as files, database, internet, or mobile devices.

This step includes the below tasks:

- Identify various data sources
- Collect data
- Integrate the data obtained from different sources

After performing these tasks, we will get a coherent set of data which is called a dataset.

2.2.2 Data Preparation:

Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training. In this step, first, we put all data together, and then randomize the ordering of data. This step can be further divided into two processes:

Data exploration:

It is used to understand the nature of data that we must work with. We need to understand the characteristics, format, and quality of data. A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.

Data pre-processing:

Now the next step is pre-processing of data for its analysis.

2.2.3 Data Wrangling:

Data wrangling is the process of cleaning and converting raw data into a usable format. It is the process of cleaning the data, selecting the variable to use, and transforming the data in a proper format to make it more suitable for analysis in the next step. It is one of the most important steps of the complete process. Cleaning of data is required to address the quality issues. In real-world applications, collected data may have various issues, including:

- Missing Values
- Duplicate data
- Invalid data
- Noise

2.2.4 Data Analysis:

This step involves:

- Selection of analytical techniques
- Building models
- Review the result

The aim of this step is to build a machine learning model to analyse the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the machine learning techniques such as Classification, Regression, Cluster analysis, Association, etc. then build the model using prepared data, and evaluate the model.

2.2.5 Train Model:

In this step we train our model to improve its performance for better outcome of the problem. We use datasets to train the model using various machine learning algorithms. Training a model is required so that it can understand the various patterns, rules, and features

2.2.6 Test Model:

In this step, we check for the accuracy of our model by providing a test dataset to it. Testing the model determines the percentage accuracy of the model as per the requirement of project or problem.

2.2.7 Deployment:

The last step of the machine learning life cycle is deployment, where we deploy the model in the real-world system. If the above-prepared model is producing an accurate result as per our requirement with acceptable speed, then we deploy the model in the real system. But before deploying the project, we will check whether it is improving its performance using available data or not. The deployment phase is like making the final report for a project.

2.3 ML Uses

ML is used to build algorithms that can receive the input data and use statistical analysis to predict the output so there are limitless applications of machine learning. They are available in every form from simple to highly complex. The system can perform the following tasks by Machine Learning:

- Image Recognition.
- Speech Recognition.
- Medical diagnosis.
- Statistical Arbitrage.
- Learning associations.
- Classification.
- Prediction.
- Extraction.
- Regression.
- Financial Services.

2.4 ML Algorithm Types

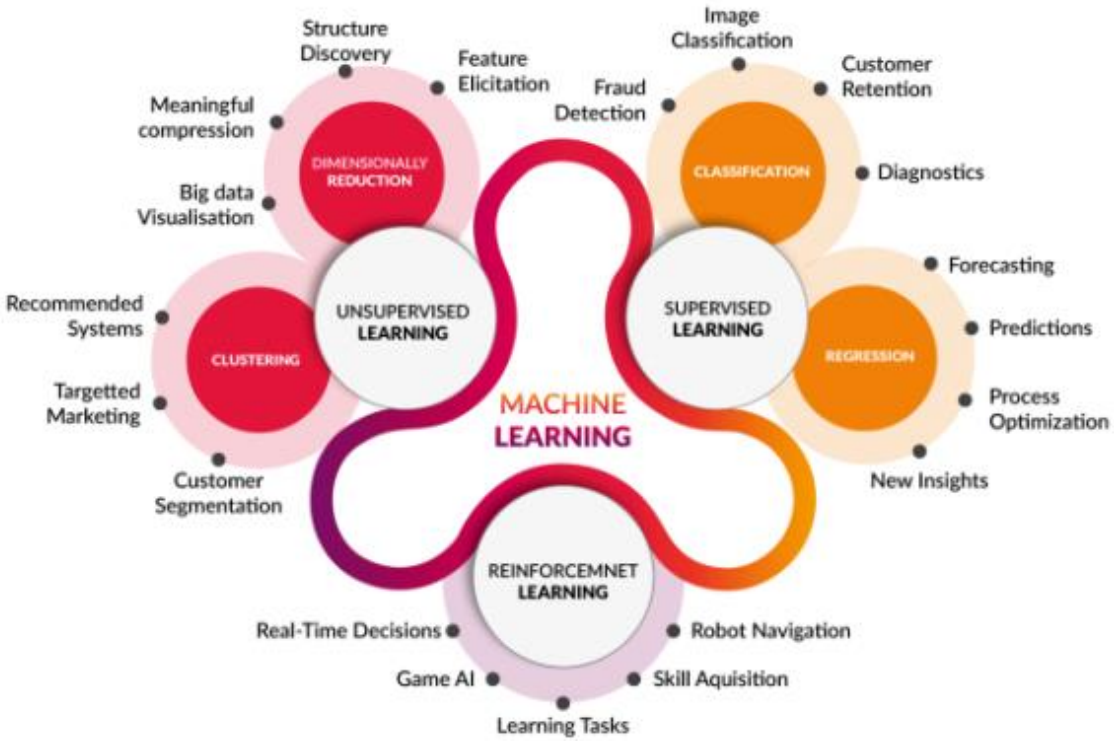


Figure 5: ML Algorithm Types **[6]**

2.4.1 Supervised Learning

Supervised learning is defined by its use of labelled datasets to train algorithms that to classify data or predict outcomes accurately. It uses a training set to teach models to yield the desired output. This training dataset includes inputs and correct outputs, which allow the model to learn over time.

The objective of a supervised learning model is to predict the correct label for newly presented input data a supervised learning algorithm can be written simply as:

$$Y = f(x)$$

The function used to connect input features to a predicted output is created by the machine learning model during training.

Types of Supervised Learning:

Supervised learning can be separated into two types of problems when data mining which are **classification and regression.**

- Classification:

Classification problem is when the output variable is a category, such as "red" or "blue" or "disease" and "no disease". **[7]**

Classification algorithm examples:

1. Binary Classification.
2. Logistic Regression.
3. k-Nearest Neighbors.
4. Decision Trees.
5. Support Vector Machine.
6. Naive Bayes.
7. Multi-Class Classification.
8. Plant species classification.
9. Face classification.

- Regression:

Regression analysis is the process of estimating the relationship between a dependent variable and independent variables.

Regression algorithms examples:

1. Linear regression.
2. Logistic regression.
3. Ridge regression.
4. Lasso regression.
5. Polynomial regression.

2.4.2 Unsupervised Learning

Unsupervised learning uses machine learning algorithms to analyze and cluster unlabeled 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, and image recognition. **[8]**

Examples of unsupervised learning:

- **Clustering:** it is a data mining technique which groups unlabelled data based on their similarities or differences. Clustering algorithms are used to process raw, unclassified data objects into groups represented by structures or patterns in the information. Clustering algorithms can be categorized into a few types, specifically exclusive, overlapping, hierarchical, and probabilistic.

Popular Clustering Types: **[9]**

1. **Distribution based methods:** It is a clustering model in which we will fit the data on the probability that it may belong to the same distribution. The grouping done may be normal or gaussian.

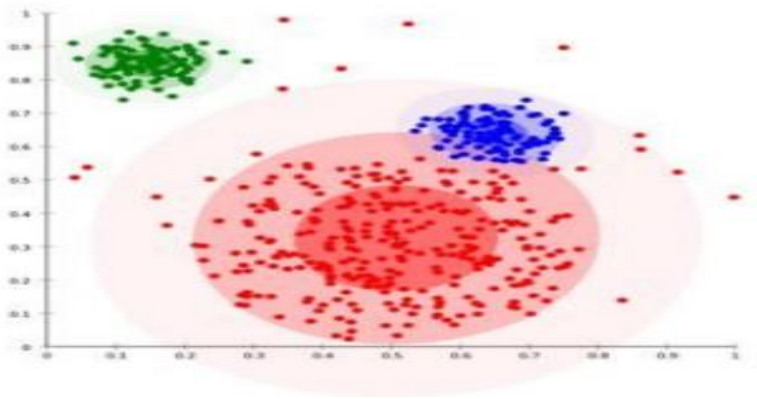


Figure 6: Distribution based methods Clustering

2. **Centroid based methods:** This is basically one of iterative clustering algorithms in which the clusters are formed by the closeness of data points to the centroid of clusters. K – means algorithm is one of the popular examples of this algorithm.

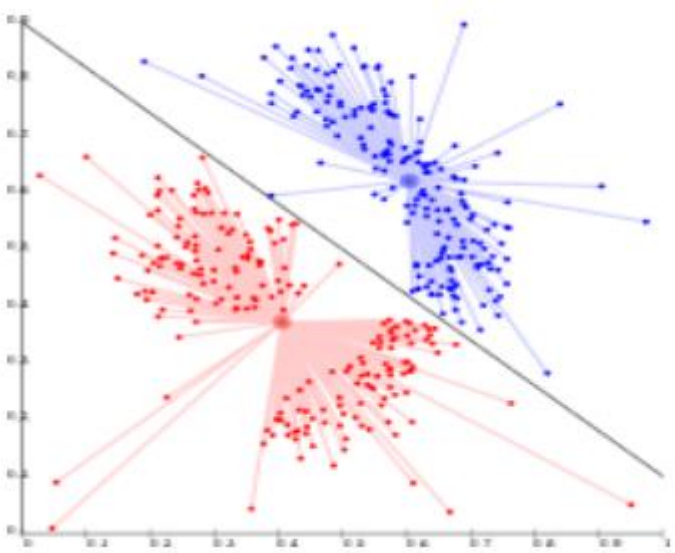


Figure 7: Centroid based methods Clustering

3. **Density Models:** In this clustering model there will be a search of data space for areas of varied density of data points in the data space. It isolates various density regions based on different densities present in the data space.

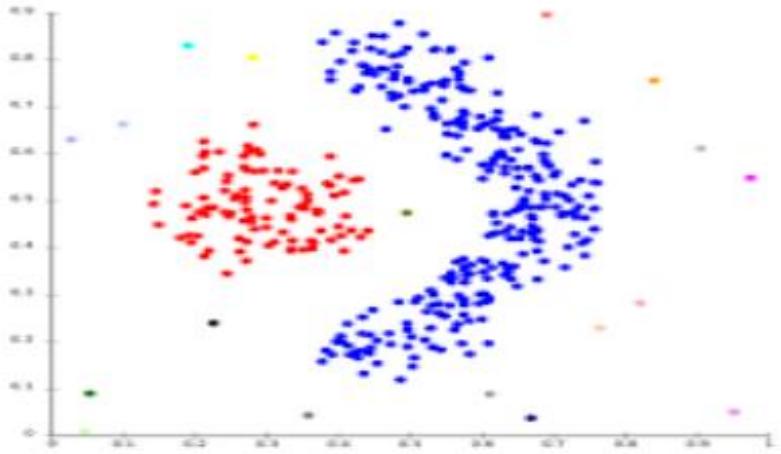


Figure 8: Density Models Clustering

4. **Connectivity Based Clustering**: which is known as Hierarchy Clustering which constructs trees of clusters of objects, in which any two clusters are disjoint, or one includes the other. The cluster of all objects is the root of the tree.

- **Association:** An association rule is an unsupervised learning method which is used for finding the relationships between variables in the large database. It determines the set of items that occur together in the dataset. **[8]**

Association rule learning can be divided into three types of algorithms: **[11]**

- Apriori.
- Eclat.
- F-P Growth Algorithm

2.4.3 Reinforcement Learning (RL)

Is an area of machine learning concerned with how intelligent agents ought to take actions in an environment in order to maximize the notion of cumulative reward, The goal of reinforcement learning is to pick the best-known action for any given state. **[10]**

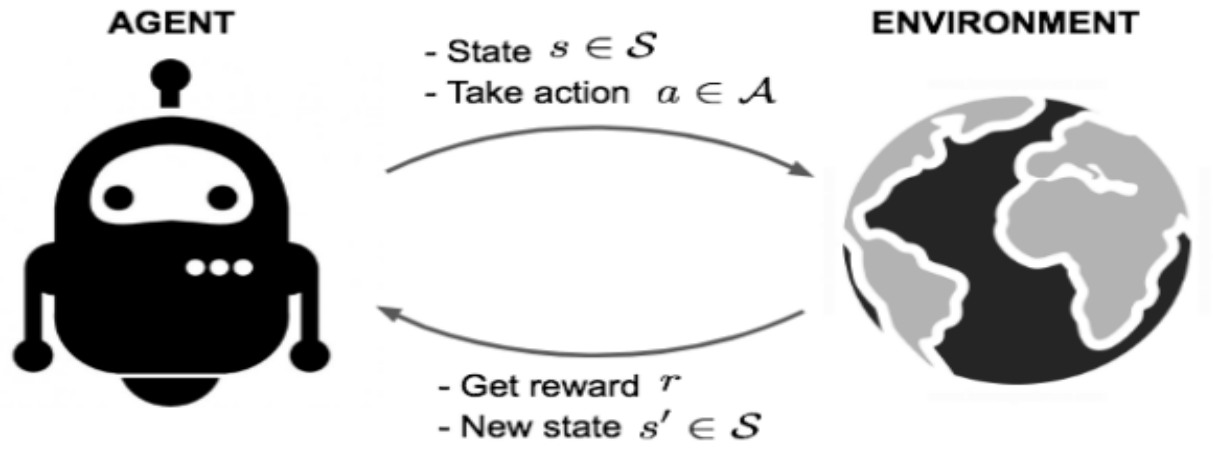


Figure 9: RL scenario

- **Agent:** It is an assumed entity which performs actions in an environment to gain some reward.
- **Environment (e):** A scenario that an agent has to face.
- **Reward (R):** An immediate return given to an agent when he or she performs specific action or task.
- **State (s):** State refers to the current situation returned by the environment.
- **Policy (π):** It is a strategy which is applied by the agent to decide the next action based on the current state.

Chapter Three

3 Deep learning

3.1 DL Definition

Deep learning is an artificial intelligence (AI) function that imitates the workings of the human brain in processing data and creating patterns for use in decision making. Deep learning is a subset of machine learning in artificial intelligence that has networks capable of learning unsupervised from data that is unstructured or unlabelled. Also known as deep neural learning or deep neural network. **[12]**

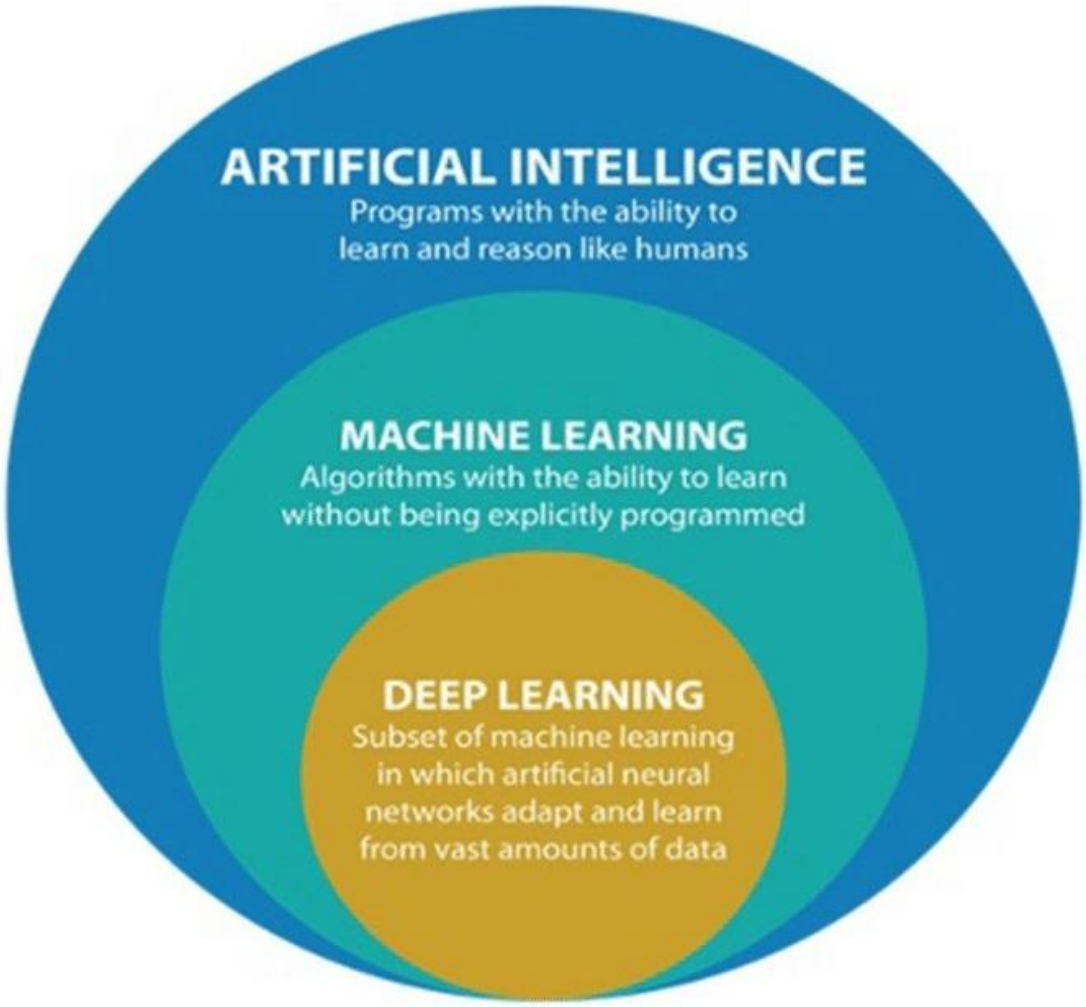


Figure 10: Understanding Deep Learning

3.2 ML vs DL

3.2.1 Data dependencies:

The most important difference between deep learning and traditional machine learning is its performance as the scale of data increases. When the data is small, deep learning algorithms don't perform that well. This is because deep learning algorithms need a large amount of data to understand it perfectly. On the other hand, traditional machine learning algorithms with their handcrafted rules prevail in this scenario. **[13]**

3.2.2 Hardware dependencies:

Deep learning algorithms heavily depend on high-end machines, contrary to traditional machine learning algorithms, which can work on low-end machines. This is because the requirements of deep learning algorithms include GPUs which are an integral part of its working. Deep learning algorithms inherently do a large amount of matrix multiplication operations. These operations can be efficiently optimized using a GPU because GPU is built for this purpose. **[13]**

3.2.3 Feature engineering:

Feature engineering is a process of putting domain knowledge into the creation of feature extractors to reduce the complexity of the data and make patterns more visible to learning algorithms to work. This process is difficult and expensive in terms of time and expertise. **[13]**

In Machine learning, most of the applied features need to be identified by an expert and then hand coded as per the domain and data type. For example, features can be pixel values, shape, textures, position, and orientation. The performance of most of the Machine Learning algorithms depends on how accurately the features are identified and extracted.

Deep learning algorithms try to learn high-level features from data. This is a very distinctive part of Deep Learning and a major step ahead of traditional Machine Learning.

Therefore, deep learning reduces the task of developing new feature extractor for every problem. Convolutional NN will try to learn low-level features such as edges and lines in early layers then parts of faces of people and then high-level representation of a face.

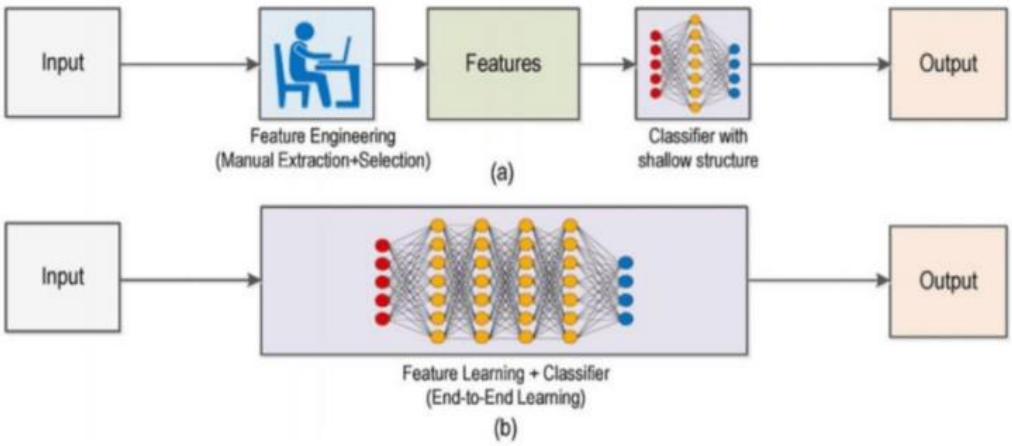


Figure 11:Feature Engineering.

3.2.4 Execution time:

a deep learning algorithm takes a long time to train. This is because there are so many parameters in a deep learning algorithm that training them takes longer than usual whereas machine learning comparatively takes much less time to train, ranging from a few seconds to a few hours.

This in turn is completely reversed on testing time. At test time, deep learning algorithms take much less time to run. Whereas, if you compare it with k-nearest neighbors (a type of machine learning algorithm), test time increases on increasing the size of data. Although this is not applicable on all machine learning algorithms, as some of them have small testing times too. **[14]**

3.3 Neural Network

The inspiration for deep learning is the way that the human brain filters the information. Its main motive is to simulate human-like decision making. Neurons in the brain pass the signals to perform the actions. artificial neurons connect in a neural network to perform tasks clustering, classification, or regression. **[14]**

3.3.1 Layers

Neurons are grouped into three different types of layers:

1. Input layer: It receives the input data from the observation.
2. Hidden layer: It performs mathematical computations on input data.
3. Output layer: It gives the desired result.

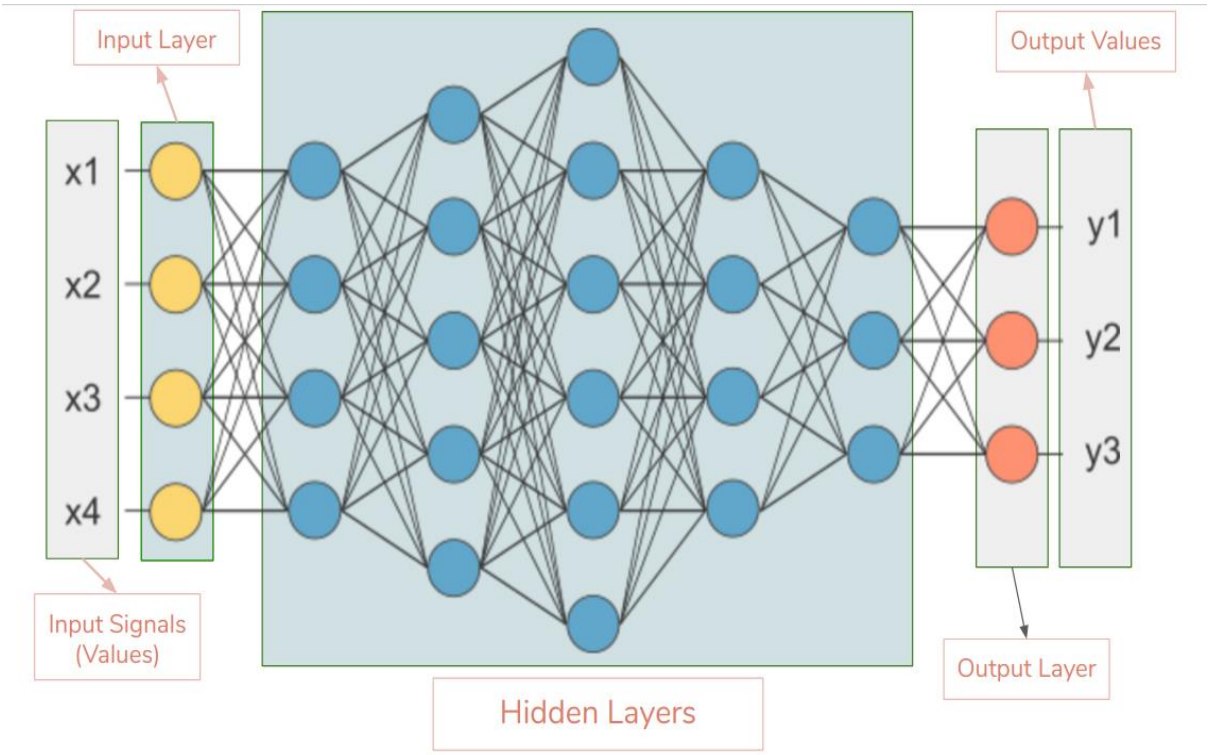


Figure 12: Types of Neurons Layers

3.3.2 Weight

The connection between neurons is called weight, which is the numerical value. The weight between neurons determines the learning ability of the neural network. **[14]**

3.3.3 Activation function

It is used for standardizing the output from the neuron, Activation functions are the mathematical equations that calculate the output of the neural network. It also helps to normalize the output in a range between 0 to 1 or -1 to 1. **[14]**

- Types of Activation Functions: [17]
  1. Linear Function:
  2. Sigmoid Function
  3. Tanh Function
  4. Rectified Linear Unit Function (ReLU):
  5. Softmax Activation Function:

In an ANN, the sigmoid function is a non-linear AF used primarily in feedforward neural networks. It is a differentiable real function, defined for real input values, and containing positive derivatives everywhere with a specific degree of smoothness. The sigmoid function appears in the output layer of the deep learning models and is used for predicting probability-based outputs.

3.4 DL Algorithm Types

3.4.1 Artificial Neural Network

An artificial neural network (ANN) is the piece of a computing system designed to simulate the way the human brain analyses and processes information. **[15]**

It is the foundation of artificial intelligence (AI) and solves problems that would prove impossible or difficult by human or statistical standards. ANNs have self-learning capabilities that enable them to produce better results as more data becomes available.

An ANN has hundreds, or thousands of artificial neurons called processing units, which are interconnected by nodes. These processing units are made up of input and output units. The input units receive various forms and structures of information based on an internal weighting system, and the neural network attempts to learn about the information presented to produce one output report. **[15]**

In ANN, Data is processed in two different propagations which are forward propagation and backpropagation.

1. **Forward Propagation:**

The input X provides the initial information that then propagates to the hidden units at each layer and finally produces the output Y. The architecture of the network entails determining its depth, width, and activation functions used on each layer. Depth is the number of hidden layers. Width is the number of units (nodes) on each hidden layer since we don't control neither input layer nor output layer dimensions. There are quite a few sets of activation functions such as Rectified Linear Unit, Sigmoid, Hyperbolic tangent, etc.

Forward propagation (or forward pass) refers to the calculation and storage of intermediate variables (including outputs) for a neural network in order from the input layer to the output layer**. [16]**

For the sake of simplicity, let us assume that the input example is $X \in R^d$ and that our hidden layer does not include a bias term. Here d the intermediate variable is: **[16]**

$$z = W^{(1)}x,$$

Where $W^{(1)} \in R^{h*d}$ is the weight parameter of the hidden layer. After running the intermediate variable $Z \in R^h$ through the activation h function ϕ we obtain our hidden activation vector of length h **[16]**

$$h = \emptyset(z)$$

The hidden variable h is also an intermediate variable. Assuming that the parameters of the output layer only possess a weight of $W^{(2)} \in R^{q*h}$,we can obtain an output layer variable with a vector of length O. **[17]**

$$o = W^{(2)}h$$

Assuming that the loss function is l and the example label is y, we can then calculate the loss term for a single data example **[16]**

$$L = l(o, y)$$

According to the definition of L2 regularization, given the hyperparameter λ, the regularization term is

$$s = \frac{\lambda}{2}(\|W^{(1)}\|_F^2 + \|W^{(2)}\|_F^2)$$

where the Frobenius norm of the matrix is simply the L2 norm applied after flattening the matrix into a vector. Finally, the model's regularized loss on a given data example is: **[16]**

$$J = L + s$$

We refer to J as the objective function.

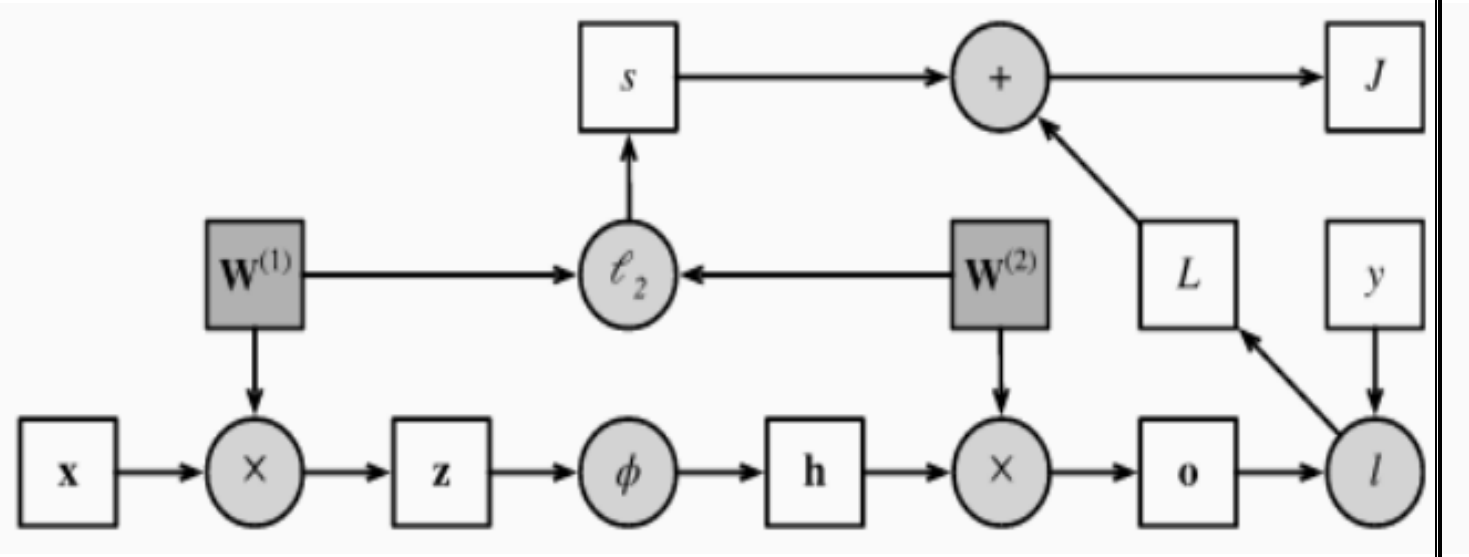


Figure 13: Computational graph of forward propagation

2. Backpropagation:

Backpropagation refers to the method of calculating the gradient of neural network parameters. In short, the method traverses the network in reverse order, from the output to the input layer, according to the chain rule from calculus. The algorithm stores any intermediate variables (partial derivatives) required while calculating the gradient with respect to some parameters. Assume that we have functions Y=f(X) and Z=g(Y), in which the input and the output X, Y, Z are tensors of arbitrary shapes. By using the chain rule, we can compute the derivative of Z with respect to X via

$$\frac{\partial Z}{\partial X} = prod(\frac{\partial Z}{\partial Y}, \frac{\partial Y}{\partial X})$$

Here we use the prod operator to multiply its arguments after the necessary operations, such as transposition and swapping input positions, have been carried out. For vectors, this is straightforward: it is simply matrix-matrix multiplication. For higher dimensional tensors, we use the appropriate counterpart. The operator prod hides all the notation overhead. Recall that the parameters of the simple network with one hidden layer, whose computational graph is in **Figure 9**, are $W^{(1)}\ and\ W^{(2)}$. The objective of backpropagation is to calculate the gradients $\frac{\partial J}{\partial W^{(1)}}\ and\ \frac{\partial Y}{\partial W^{(2)}}$. To accomplish this, we apply the chain rule and calculate, in turn, the gradient of each intermediate variable and parameter. The order of calculations are reversed relative to those performed in forward propagation, since we need to start with the outcome of the computational graph and work our way towards the parameters. The first step is to calculate the gradients of the objective function J=L+s with respect to the loss term L and the regularization term s. **[16]**

$$\frac{\partial J}{\partial L} = 1\ and\ \frac{\partial J}{\partial s} = 1$$

Next, we compute the gradient of the objective function with respect to variable of the output layer o according to the chain rule: **[17]**

$$\frac{\partial J}{\partial \mathbf{o}} = \text{prod}\left(\frac{\partial J}{\partial L}, \frac{\partial L}{\partial \mathbf{o}}\right) = \frac{\partial L}{\partial \mathbf{o}} \in \mathbb{R}^q.$$

Next, we calculate the gradients of the regularization term with respect to both parameters: **[16]**

Now we are able to calculate the gradient $\frac{\partial J}{\partial W^{(2)}} \in R^{q*h}$ of the model parameters closest to the output layer. Using the chain rule yields: **[16]**

$$\frac{\partial J}{\partial \mathbf{W}^{(1)}} = \text{prod}\left(\frac{\partial J}{\partial \mathbf{z}}, \frac{\partial \mathbf{z}}{\partial \mathbf{W}^{(1)}}\right) + \text{prod}\left(\frac{\partial J}{\partial s}, \frac{\partial s}{\partial \mathbf{W}^{(1)}}\right) = \frac{\partial J}{\partial \mathbf{z}}\mathbf{x}^\top + \lambda \mathbf{W}^{(1)}.$$

To obtain the gradient with respect to $W^{(1)}$ we need to continue back propagation along the output layer to the hidden layer. The gradient with respect to the hidden layer's outputs $\frac{\partial J}{\partial h} \in R^h$ is given by

$$\frac{\partial J}{\partial \mathbf{h}} = \text{prod}\left(\frac{\partial J}{\partial \mathbf{o}}, \frac{\partial \mathbf{o}}{\partial \mathbf{h}}\right) = {\mathbf{W}^{(2)}}^\top \frac{\partial J}{\partial \mathbf{o}}.$$

Since the activation function ϕ applies elementwise, calculating the gradient $\frac{\partial J}{\partial z} \in R^h$ of the intermediate variable z requires that we use the element wise multiplication operator, which we denote by

$$\frac{\partial J}{\partial \mathbf{z}} = \text{prod}\left(\frac{\partial J}{\partial \mathbf{h}}, \frac{\partial \mathbf{h}}{\partial \mathbf{z}}\right) = \frac{\partial J}{\partial \mathbf{h}} \odot \phi'(\mathbf{z}).$$

$$\frac{\partial J}{\partial \mathbf{W}^{(1)}} = \text{prod}\left(\frac{\partial J}{\partial \mathbf{z}}, \frac{\partial \mathbf{z}}{\partial \mathbf{W}^{(1)}}\right) + \text{prod}\left(\frac{\partial J}{\partial s}, \frac{\partial s}{\partial \mathbf{W}^{(1)}}\right) = \frac{\partial J}{\partial \mathbf{z}}\mathbf{x}^\top + \lambda \mathbf{W}^{(1)}.$$

Finally, we can obtain the gradient $\frac{\partial J}{\partial W^{(1)}} \in R^{h*d}$ of the model parameters closest to the input layer. According to the chain rule, we get

3.4.2 Convolutional Neural Network

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. **[18]**

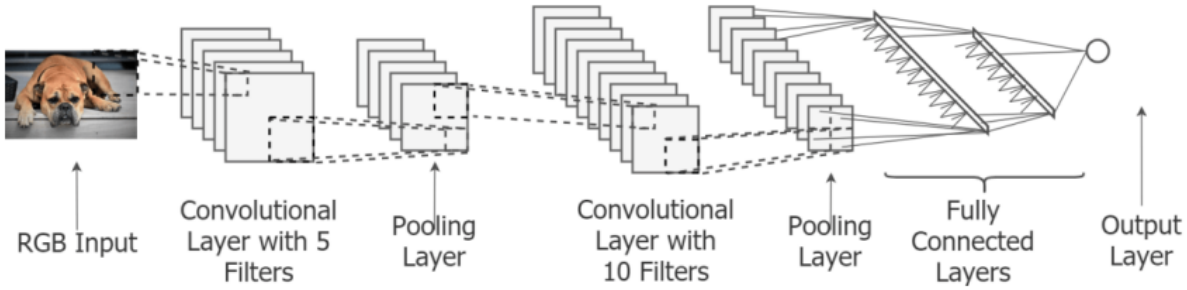


Figure 14: CNN Architecture

Convolutional neural network (CNN), a class of artificial neural networks that has become dominant in various computer vision tasks, is attracting interest across a variety of domains, including radiology. CNN is designed to learn spatial hierarchies of features automatically and adaptively through backpropagation by using multiple building blocks, such as convolution layers, pooling layers, and fully connected layers.

CNN Layers:

1. **Convolution Layer:** Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter or kernel. Convolution is a mathematical operation on two objects to produce an outcome that expresses how the shape of one is modified by the other.

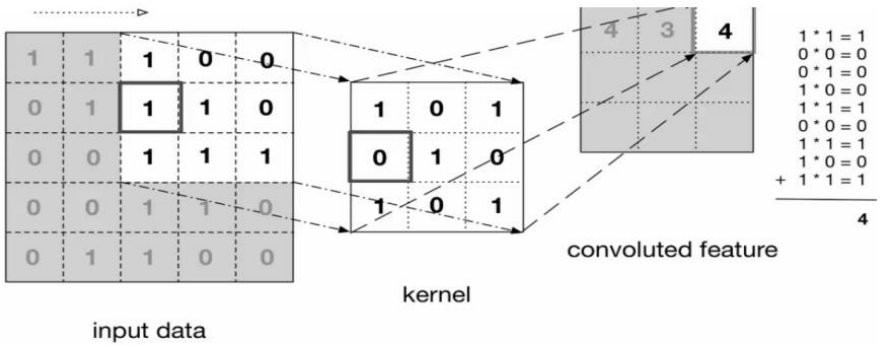


Figure 15: Convolution Layer

CNN uses filters (Kernels) to detect specific features in an image in order to get useful features.

Filter (Kernel) is a set of weights in a matrix applied on an image or a matrix to obtain the required features. We can obtain different output using different filters and here are some types of filters

Figure 16: Types of filters

- **Strides:** Stride is the number of pixels shifted over the input matrix. When the stride is 1 then we move the filters to 1 pixel at a time. When the stride is 2 then we move the filters to 2 pixels at a time and so on. **[19]**
- **Padding:** Sometimes the filter does not fit the input image. We have two options:
    - Pad the picture with zeros (zero-padding) so that it fits
    - Drop the part of the image where the filter did not fit. This is called valid padding which keeps only valid part of the image. **[19]**

2. **Pooling Layer:** Pooling layers section would reduce the number of parameters when the images are too large. Spatial pooling is also called subsampling or down sampling which reduces the dimensionality of each map but retains important information. Spatial pooling can be of different types:
    - Max Pooling
    - Average Pooling
    - Sum Pooling

Max pooling takes the largest element from the rectified feature map. Taking the largest element could also take the average pooling. Sum of all elements in the feature map call as sum pooling. **[19]**

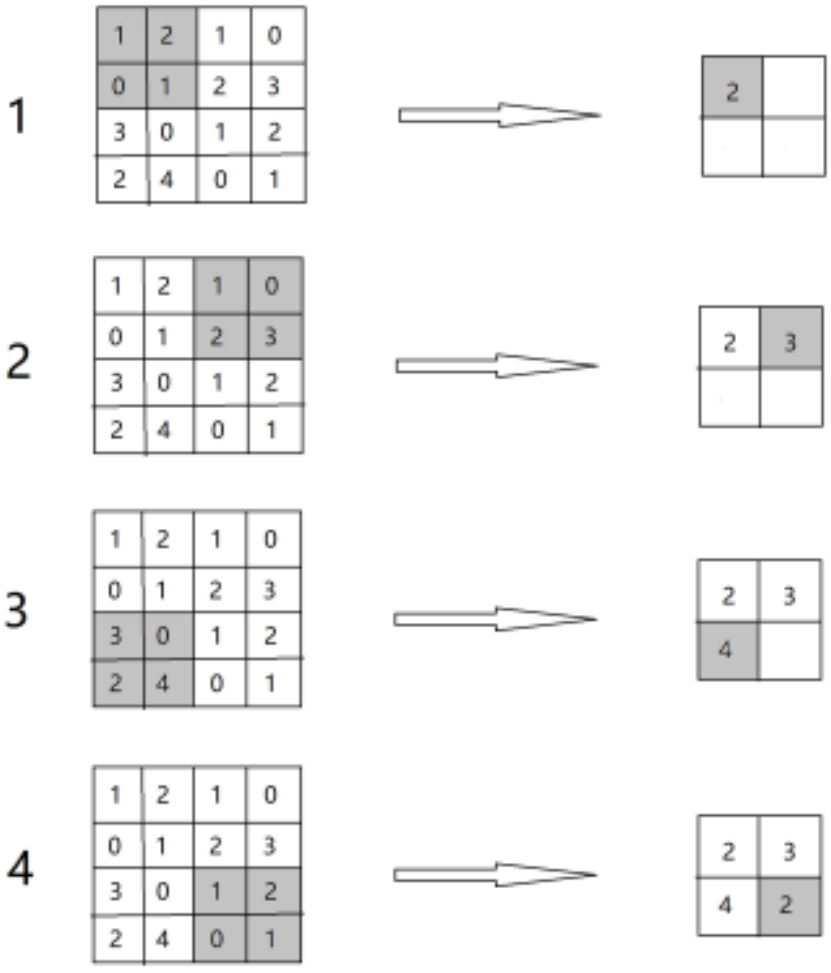


Figure 17: Example of Pooling (Max pooling)

3. **Flatten layer:** Flattening is converting the data into a 1-dimensional array for inputting it to the next layer. We flatten the output of the convolutional layers to create a single long feature vector. **[20]**

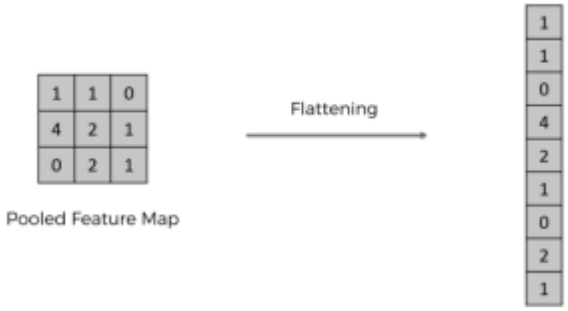


Figure 18: Flatten layer

4. **Fully Connected Layer:** which is feedforward neural networks, neurons in a fully connected layer have full connections to all activations in the previous layer, as it is in regular ANN, and their activation can be computed via a matrix multiplication followed by a bias offset.

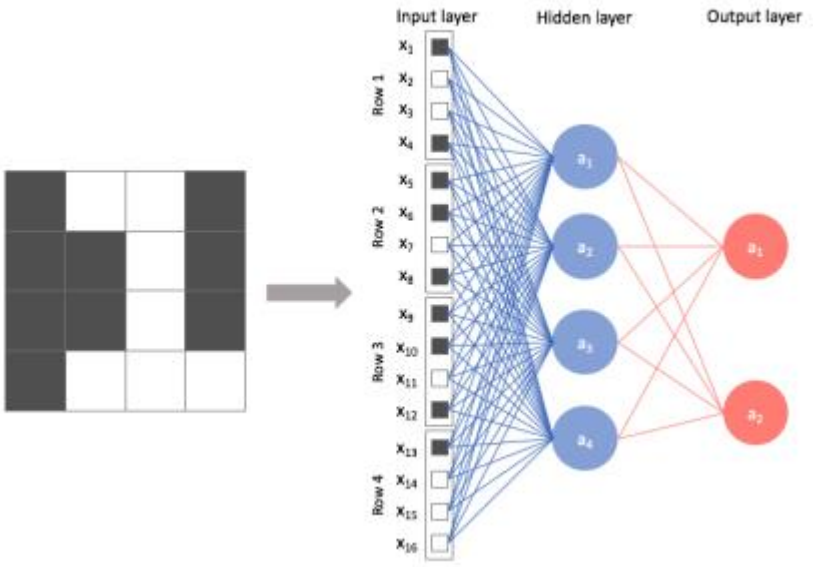


Figure 19: Fully connected layer

The problem with CNN is that the objects of interest might have different spatial locations within the image and different aspect ratios. And this leads us to get a huge number of regions, and this could computationally blow up. Therefore, algorithms like R-CNN have been developed to find these occurrences and find them fast.

3.4.3 Region-Based Convolutional Neural Network:

R-CNN is short for "Region-based Convolutional Neural Networks". The main idea is composed of two steps. First, using selective search, it identifies a manageable number of bounding-box object region candidates ("region of interest" or "RoI"). And then it extracts CNN features from each region independently for classification.

The CNN acts as a feature extractor and the output dense layer consists of the features extracted from the image and the extracted features are fed into an SVM to classify the presence of the object within that candidate region proposal.

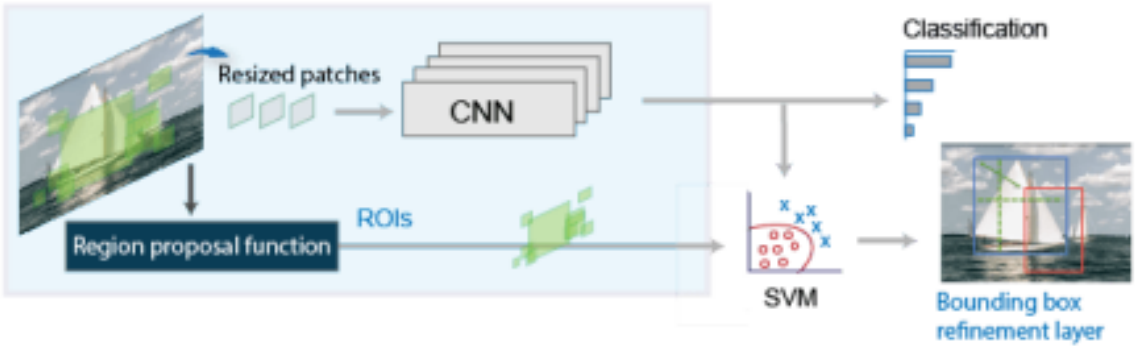


Figure 20: R-CNN Architecture **[23]**

1. **Model Workflow: [21]**

- Pre-train a CNN network on image classification tasks.
- Propose category-independent regions of interest by selective search. Those regions may contain target objects and they are of different sizes. ○ Selective search is a common algorithm to provide region proposals that potentially contain objects. It is built on top of the image segmentation output and uses region-based characteristics. Region candidates are warped to have a fixed size as required by CNN.
- Continue fine-tuning the CNN on warped proposal regions for K + 1 classes.
- Given every image region, one forward propagation through the CNN generates a feature vector. This feature vector is then consumed by a binary SVM trained for each class independently. The positive samples are proposed regions with IoU (intersection over union) overlap threshold >= 0.3, and negative samples are irrelevant.

To reduce the localization errors, a regression model is trained to correct the predicted detection window on the bounding box correction offset using CNN features.

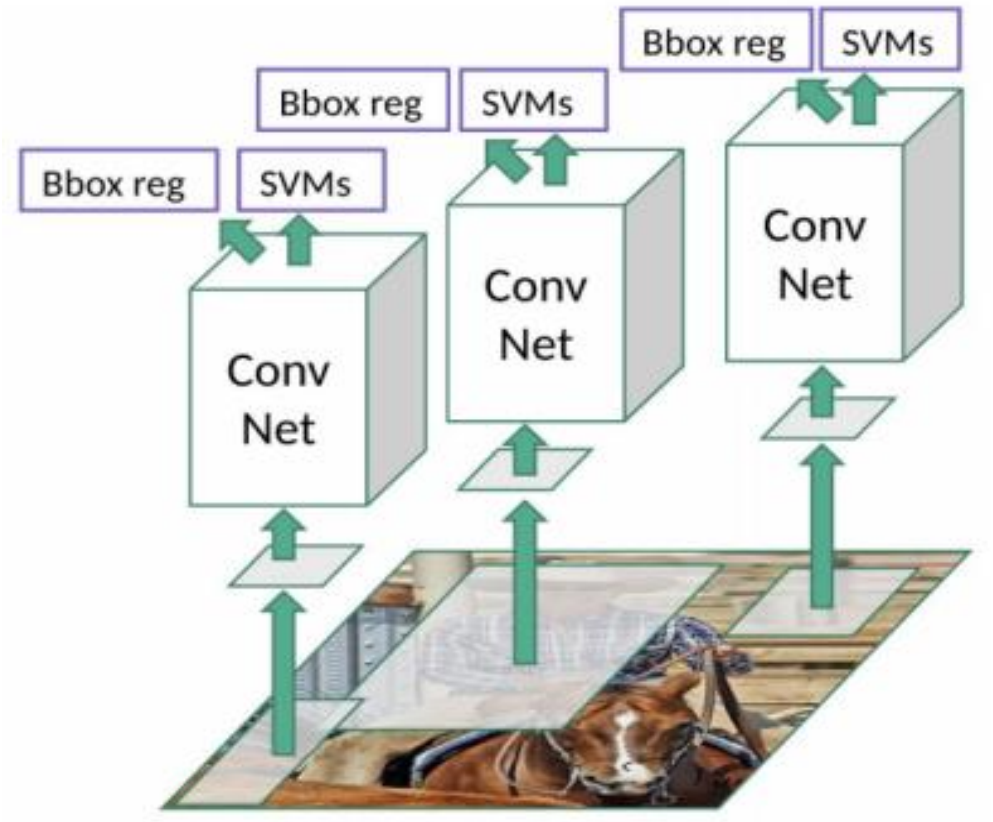


Figure 21: R-CNN Model **[24]**

2. Bounding Box Regression:

Given a predicted bounding box coordinate p=(px,py,pw,ph) (center coordinate, width, height) and its corresponding ground truth box coordinates g=(gx,gy,gw,gh) the regressor is configured to learn scale-invariant transformation between two centres and log-scale transformation between widths and heights. All the transformation functions take p as input.

$$\hat{g}_x = p_w d_x(\mathbf{p}) + p_x$$
$$\hat{g}_y = p_h d_y(\mathbf{p}) + p_y$$
$$\hat{g}_w = p_w \exp(d_w(\mathbf{p}))$$
$$\hat{g}_h = p_h \exp(d_h(\mathbf{p}))$$

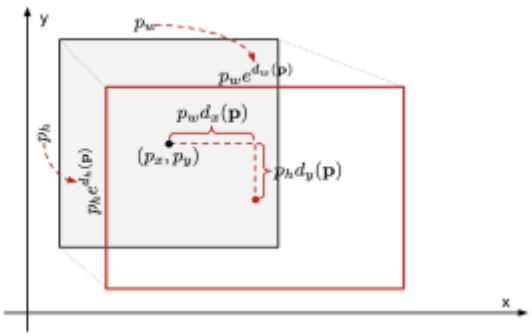


Figure 22: Bounding Box **[22]**

An obvious benefit of applying such transformation is that all the bounding box correction functions di (p) where i∈{x,y,w,h} can take any value between [-∞, +∞].

3. **Disadvantage:**
    - One of the most common defects of R-CNN is speed bottleneck as the model is expensive and slow.
    - The main performance bottleneck of an R-CNN lies in the independent CNN forward propagation for each region proposal without sharing computation.

3.4.4 Fast R-CNN

To make R-CNN faster, Girshick improved procedure by unifying three independent models into one jointly trained framework and increasing shared computation results , One of the major improvements of the fast R-CNN from the R-CNN is that the CNN forward propagation is only performed on the entire image and the region proposals share this feature matrix, Then the same feature matrix is branched out to be used for learning the object classifier and the bounding-box regressor.

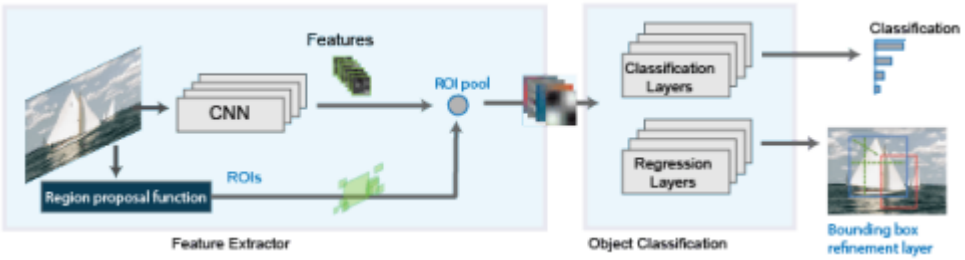


Figure 23: Fast R-CNN Architecture **[23]**

RoI Pooling

It is a type of max pooling to convert features in the projected region of the image of any size, h x w, into a small, fixed window, H x W. The input region is divided into H x W grids, approximately every sub window of size h/H x w/W. Then apply max pooling in each grid.

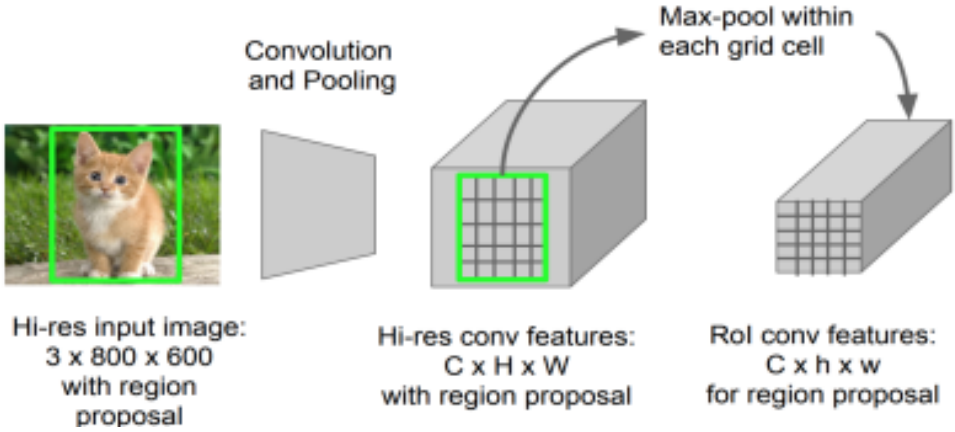


Figure 24: ROI Pooling **[22]**

1. **Model Workflow [22]**

- First, pre-train a convolutional neural network on image classification tasks.
- Propose regions by selective search (~2k candidates per image). Alter the pre-trained CNN:
- Replace the last max pooling layer of the pre-trained CNN with a ROI Pooling layer. The RoI pooling layer outputs fixed-length feature vectors of region proposals. Sharing the CNN computation makes a lot of sense, as many region proposals of the same images are highly overlapped.
- Replace the last fully connected layer and the last softmax layer (K classes) with a fully connected layer and softmax over K + 1 classes.
- Finally, the model branches into two output layers:
- A softmax estimator of K + 1 classes (same as in R-CNN, +1 is the "background" class), outputting a discrete probability distribution per RoI.
- A bounding-box regression model which predicts offsets relative to the original RoI for each of K classes.
- Fast R-CNN is much faster in both training and testing time.

2. **Disadvantage:**

The improvement is not dramatic because the regional proposals are generated separately by another model and that is very expensive.

3.4.5 Faster R-CNN

An intuitive speedup solution is to integrate the region proposal algorithm into the CNN model. Faster R-CNN Ren is doing exactly this: construct a single, unified model composed of RPN (region proposal network) and fast R-CNN with shared convolutional feature layers.

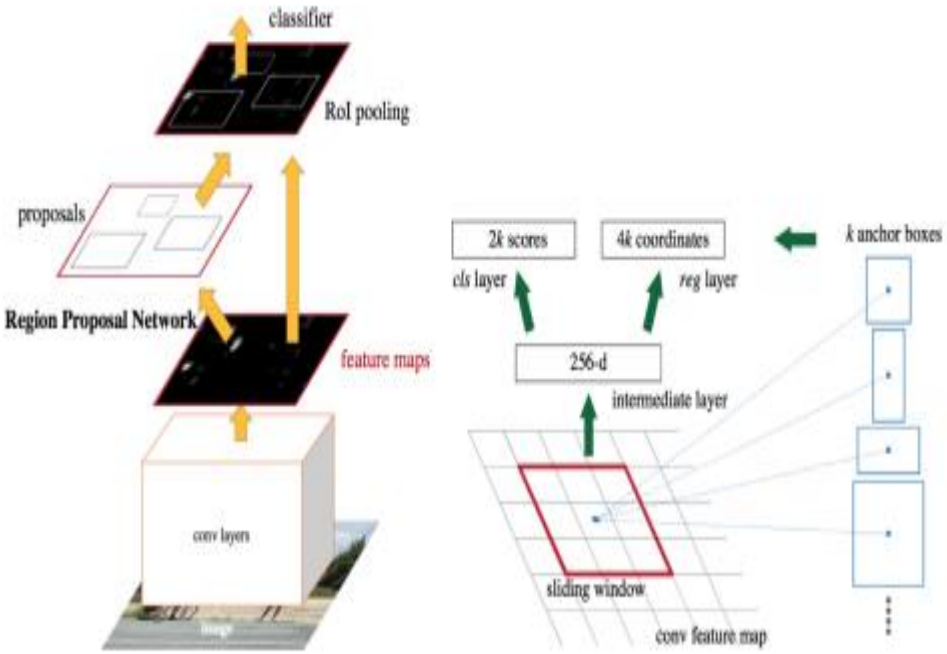


Figure 25: Faster R-CNN architecture. **[23]**

1. **Model Workflow [24]**

- Pre-train a CNN network on image classification tasks.
- Fine-tune the RPN (region proposal network) end-to-end for the region proposal task, which is initialized by the pre-train image classifier. Positive samples have IoU (intersection-over-union) > 0.7, while negative samples have IoU < 0.3.
- Slide a small n x n spatial window over the conv feature map of the entire image.

- At the center of each sliding window, we predict multiple regions of various scales and ratios simultaneously. An anchor is a combination of (sliding window center, scale, ratio). For example, 3 scales + 3 ratios => k=9 anchors at each sliding position.
- Train a Fast R-CNN object detection model using the proposals generated by the current RPN
- Then use the Fast R-CNN network to initialize RPN training. While keeping the shared convolutional layers, only fine-tune the RPN-specific layers.
- Finally fine-tune the unique layers of Fast R-CNN.

3.4.6 RNN

The AI of this project has to make a prediction depending on a sequence of frames. In this case, each frame has a position of the landmarks of the hands and the combination of the frames of a video create a sign. Recurrent neural networks are capable of solving this type of problems.

RNN is a type of neural network, which can process sequential data with variable length. They apply the same function over a sequence recurrently. Unlike regular networks, where the state only depends on the current input (and network weights), RNN also depend on the previous states.

RNN can be defined as a recurrence relation:

$$st = f(st-1, xt) \qquad (3.1)$$

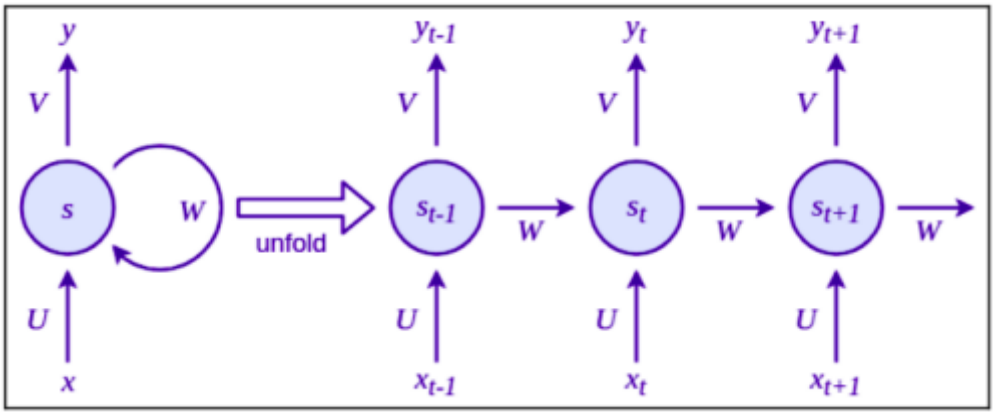
Where f is a differential function, st is a vector of values called internal network state (at step t), and xt is the network input at step t. As shown in equation 3.1, the internal network state at a given moment st depends on the previous internal network state st−1. At the same time, st−1 depends on st−2 and so on. Therefore, the current state depends and all the previous ones.

Once the internal wights of the RNN are added, the output is as shown in equation 3.3.

$$st = f(st-1 * W + xt * U) \qquad (3.2)$$

$$yt = st * V \qquad (3.3)$$

Where W and U transform the previous state st−1 and the input xt respectively, V transforms the current state st , and yt is the output of the RNN. In the figure below we can see the typical architecture of a RNN and its weights:



Convolutional neural networks are great for a 1 to 1 relation What CNNs cannot do is accept a sequence of vectors.

That's where Recurrent Neural Networks (RNNs) are used, RNNs allow us to understand the context of a video frame, relative to the frames that came before it. They do this by passing the output of one training step to the input of the next training step, along with the new frame

Each of the neural network's weights receives a new value proportional to the partial derivative of an error function with respect to the current weight in each iteration of training. In some cases, the proportion changes created by earlier states will be very small compared to resent states, which removes the early states weight in the RNN.

To solve this, the RNN of this project uses LSTM (Long Short-Term Memory) units. An LSTM unit is composed of a cell with an input gate, an output gate and a forget gate. This cell contains the temporal state which can handle long-term dependencies. Depending of the LSTM function, the LSTM cell can erase the temporal state and allow a new one to be stored or keep the same state that had before. Therefore, while the normal RNN states may have forgotten old dependencies, LSTM cells still hold them.

Long-Short-Term memory (LSTM).

Long short-term memory (LSTM) is an artificial neural network used in the fields of artificial intelligence and deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. Such a recurrent neural network can process not only single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected handwriting recognition speech recognition machine translation robot control] video games and healthcare. LSTM has become the most cited neural network of the 20th century.

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate, The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell.

LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the vanishing gradient problem that can be encountered when training traditional RNNs. Relative insensitivity to gap length is an advantage of LSTM over RNNs, hidden Markov models and other sequence learning methods in numerous applications.

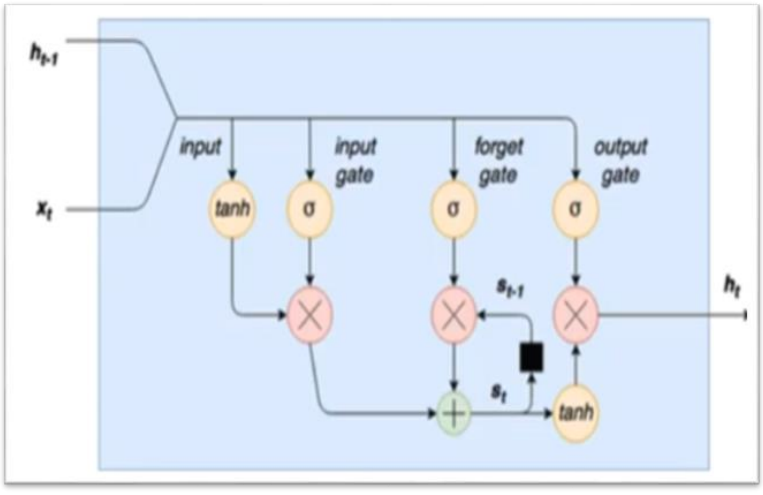
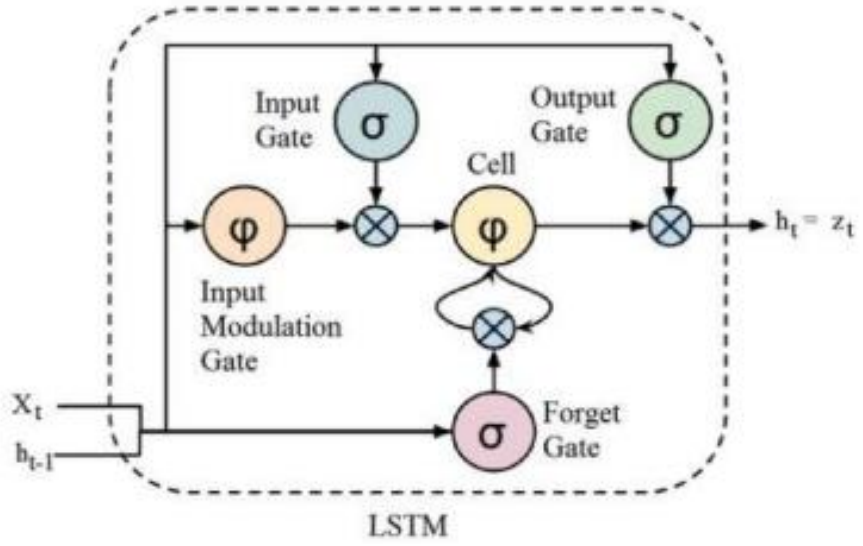


Figure 26: LSTM Archticture

Convolutional neural networks are great for a 1 to 1 relation; given an image of a sign, it generates fixed-size label, like the class of the sign in the image. However, What CNNs cannot do is accept a sequence of vectors. That's where Recurrent Neural Networks (RNNs) are used. RNNs allow us to understand the context of a video frame, relative to the frames that came before it. They do this by passing the output of one training step to the input of the next training step, along with the new frame. We're using a special type of RNN here, called an LSTM, that allows our network to learn long-term dependencies.



Basic LSTM cell The Mediapipe-LSTM architecture involves using a pre-trained Mediapipe for feature extraction from input data along with LSTMs for sequence prediction It is helpful to think of this architecture as defining two sub-models: the MediaPipe for feature extraction and the LSTM Model for interpreting the features across time steps. We used to methods; The first method we followed is pre-training the inception v3 model on our data. The second method is to pass the predicted labels from the Inception Mediapipe to an LSTM. After extracting the bottleneck features, we used a network consisting of single layer of 512 LSTM units, followed by a fully connected layer with Softmax activation. Finally, a regression layer is applied. For minimizing the loss function, We used Adaptive Moment Estimation (ADAM) which is a stochastic optimization algorithm that's been designed for training deep neural networks, and we trained the data for 1000 epochs.

Finally, in this chapter we took a background about machine learning and deep learning, so in **9** we will talk about the solutions we used in Blind-Deaf System.

Chapter Four

4 MediaPipe

4.1 MediaPipe definition and uses

Until now there were some technology limitations Nowadays, thanks to Google MediaPipe we can track the hands from a computer or mobile phone camera in real time. MediaPipe is an open-source framework released by Google in August 2019 that includes real time artificial vision technologies such as object tracking, face detection and multi-hand tracking.

Taking advantage of MediaPipe, the selected approach is to track the hands and its fingers from a rassperypi4 camera and then detect gestures with a LSTM in real time. The hand detection depends on MediaPipe and therefore, this research is focused on building the AI to do the gesture recognition without the possibility of improving the hand tracking and find out if MediaPipe is a tool good enough to recognize sign language. The main differences regarding previous studies, is the use of only one camera from a computer or mobile phone available nowadays and the use of a computer application capable of recognizing signs in real time.

MediaPipe is a Framework for building machine learning pipelines for processing time-series data like video, audio, etc. This cross-platform Framework works in Desktop/Server, Android, iOS, and embedded devices like Raspberry Pi and Jetson Nano. **[37]**

Uses of MediaPipe

Every Youtube video we watch is processed with machine learning models using MediaPipe. Google has not hired thousands of employees to watch every video people upload, because thousands of people are not enough to look after and check each published video, the amount of data Google gets daily is not easy for humans to check. Machine Learning models are developed to make our life easier, so tasks that are hard for us to complete, machine learning and deep learning models help us to do in less amount of time, on the other hand, we can save money by not hiring employees.

Yes, Google has machine learning/deep learning models to see if the videos match their policies and the content is not having copy-right issues.

Basically, MediaPipe is a framework for Computer Vision and Deep Learning that builds perception pipelines. For now, you just need to know, perception pipelines are some sort of audio, video, or time-series data that catch the process in pipelining zone. **[37]**

4.2 What is possible with mediaPipe

There are several AI problems that can be done by MediaPipe. Here some are mentioned: **[37]**

- Object Tracking
- Box Tracking
- Face Mesh
- Hair Segmentation
- Live Hand Tracking
- Face detection
- Holistic tracking and more.

4.3 Methodology

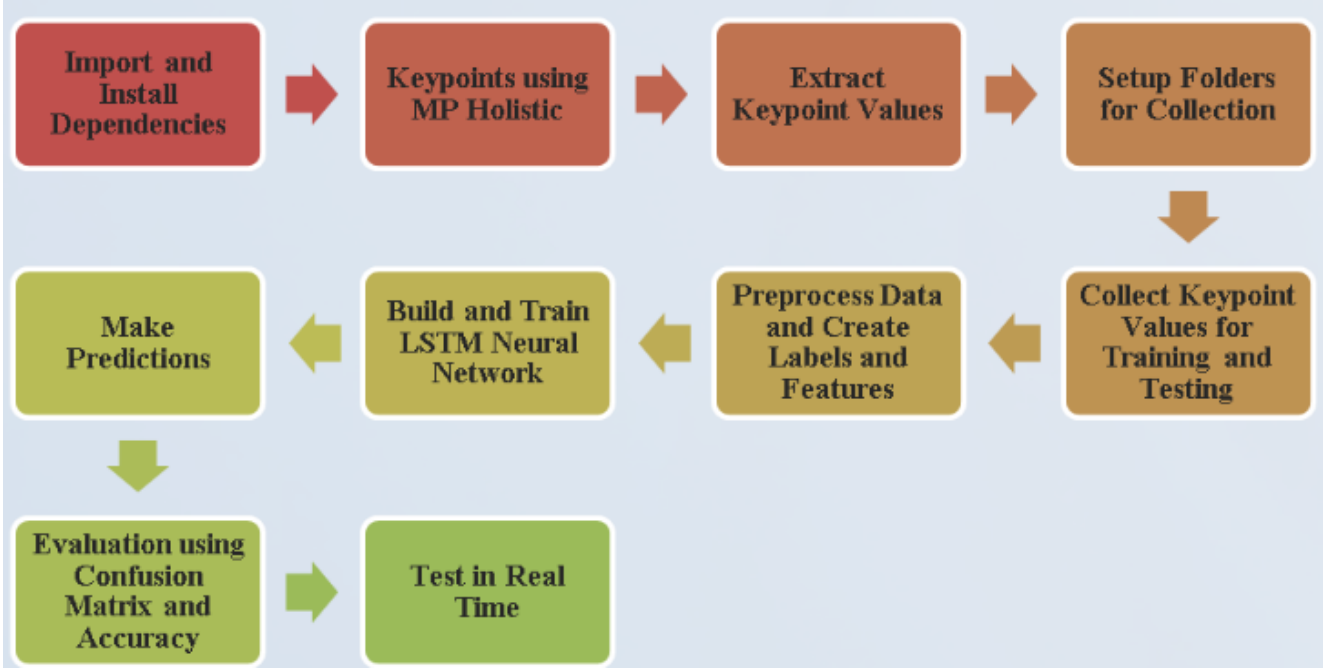


Figure 27: Methodology of mode

4.4 MediaPipe solution

4.4.1 MediaPipe Face Mesh

MediaPipe Face Mesh is a solution that estimates 468 3D face landmarks in real-time even on mobile devices. It employs machine learning (ML) to infer the 3D facial surface, requiring only a single camera input without the need for a dedicated depth sensor. Utilizing lightweight model architectures together with GPU acceleration throughout the pipeline, the solution delivers real-time performance critical for live experiences.

Additionally, the solution is bundled with the Face Transform module that bridges the gap between the face landmark estimation and useful real-time augmented reality (AR) applications. It establishes a metric 3D space and uses the face landmark screen positions to estimate a face transform within that space. The face transform data consists of common 3D primitives, including a face pose transformation matrix and a triangular face mesh. Under the hood, a lightweight statistical analysis method called Procrustes Analysis is employed to drive a robust, performant and portable logic. The analysis runs on CPU and has a minimal speed/memory footprint on top of the ML model inference. [36]

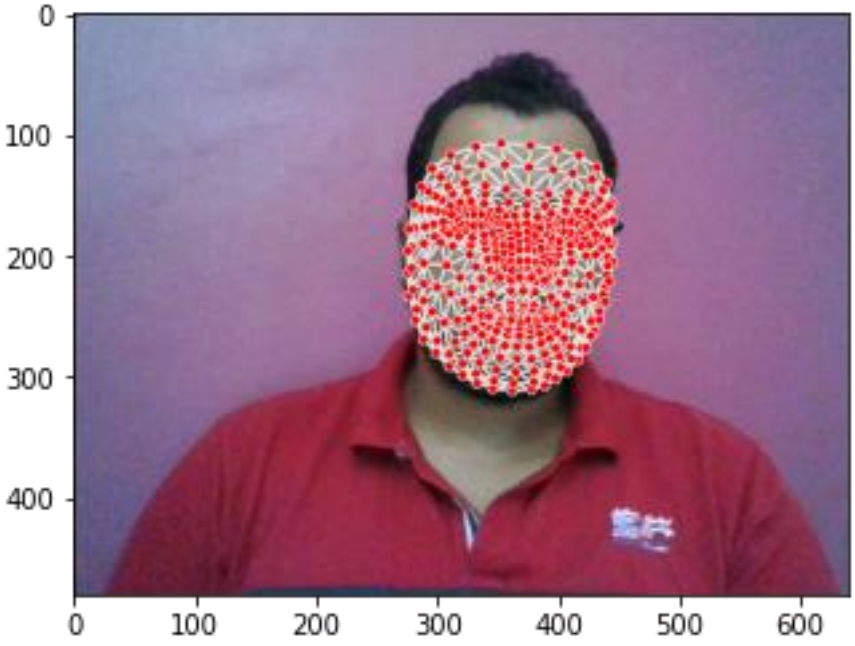


Figure 28 face landmarks

4.4.2 MediaPipe Hands

The ability to perceive the shape and motion of hands can be a vital component in improving the user experience across a variety of technological domains and platforms. For example, it can form the basis for sign language understanding and hand gesture control and can also enable the overlay of digital content and information on top of the physical world in augmented reality. While coming naturally to people, robust real-time hand perception is a decidedly challenging computer vision task, as hands often occlude themselves or each other (e.g., finger/palm occlusions and handshakes) and lack high contrast patterns.

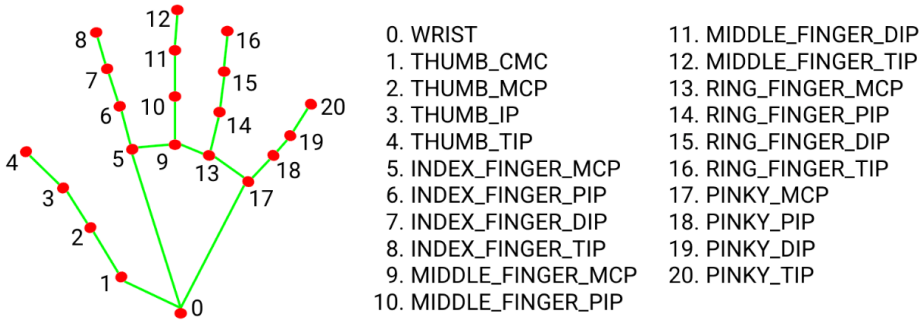


Figure 29: hand_landmarks

MediaPipe Hands is a high-fidelity hand and finger tracking solution. It employs machine learning (ML) to infer 21 3D landmarks of a hand from just a single frame. Whereas current state-of-the-art approaches rely primarily on powerful desktop environments for inference, our method achieves real-time performance on a mobile phone, and even scales to multiple hands. We hope that providing this hand perception functionality to the wider research and development community will result in an emergence of creative use cases, stimulating new applications and new research avenues.

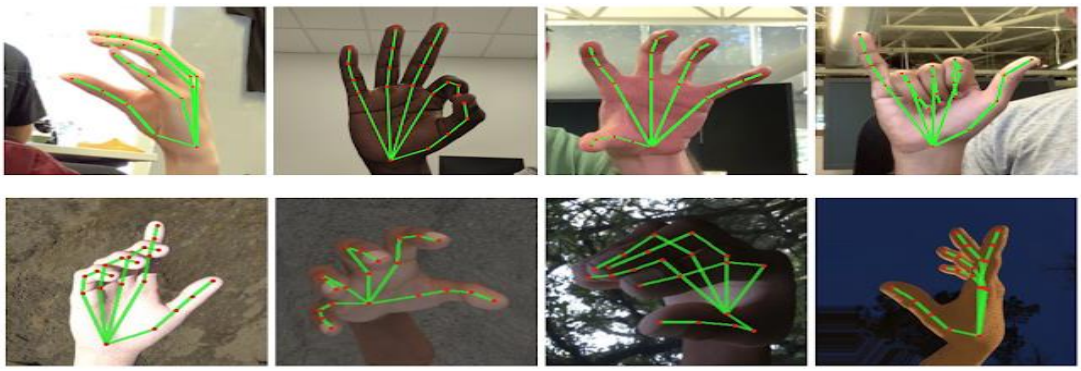
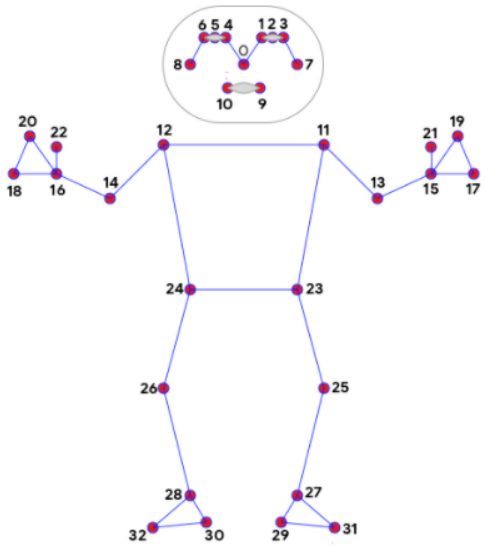


Figure 30: hand crop

4.4.3 MediaPipe Pose

Pose estimation means finding a person's or an object's key points. A person's key points are elbow, knee, wrist, etc so MediaPipe can be used for training the ML model to learn the key points and further use the knowledge for specific tasks, this actually can be useful for action recognition.

Human pose estimation from video plays a critical role in various applications such as quantifying physical exercises, sign language recognition, and full-body gesture control. For example, it can form the basis for yoga, dance, and fitness applications. It can also enable the overlay of digital content and information on top of the physical world in augmented reality.



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right_knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

Figure 31: pose landmarks

4.4.4 MediaPipe Holistic

The MediaPipe Holistic pipeline integrates separate models for

MediaPipe Pose, MediaPipe Face Mesh and MediaPipe Face Mesh is a solution that estimates 468 3D face landmarks in real-time even on mobile devices. It employs machine learning (ML) to infer the 3D facial surface, requiring only a single camera input without the need for a dedicated depth sensor. Utilizing lightweight model architectures together with GPU acceleration throughout the pipeline, the solution delivers real-time performance critical for live experiences. components, each of which are optimized for their domain. However, because of their different specializations, the input to one component is not well-suited for the others.

The pose estimation model, for example, takes a lower, fixed resolution video frame (256x256) as input. But if one were to crop the hand and face regions from that image to pass to their respective models, the image resolution would be too low for accurate articulation. Therefore, we designed MediaPipe Holistic as a multi-stage pipeline, which treats the different regions using a region appropriate image resolution.

First, we estimate the human pose (top of Figure 32) with pose detector and subsequent landmark model. Then, using the inferred pose landmarks we derive three regions of interest (ROI) crops for each hand (2x) and the face, and employ a re-crop model to improve the ROI. We then crop the full-resolution input frame to these ROIs and apply task-specific face and hand models to estimate their corresponding landmarks. Finally, we merge all landmarks with those of the pose model to yield the full 540+ landmarks.

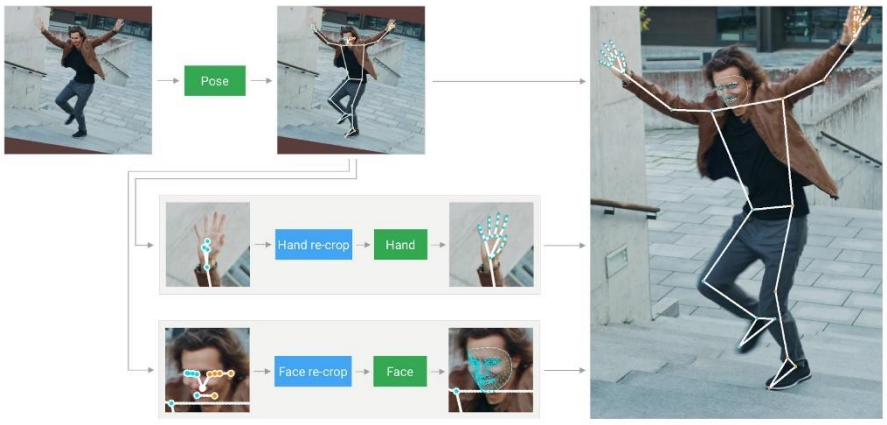


Figure 32: MediaPipe Holistic Pipeline Overview

4.4.5 MediaPipe Selfie segmentation

MediaPipe Selfie Segmentation segments the prominent humans in the scene. It can run in real-time on both smartphones and laptops. The intended use cases include selfie effects and video conferencing, where the person is close (< 2m) to the camera.

4.5 Synchronization and Performance Optimization

MediaPipe supports multimodal graphs. To speed up the processing, different calculators run in separate threads. For performance optimization, many built-in calculators come with options for GPU acceleration. Working with time series data must be in proper synchronization; otherwise, the system will break. The graph ensures this so that flow is handled correctly according to the timestamps of packets. The Framework handles synchronization, context sharing, and inter-operations with CPU calculators.

4.6 Dependency

MediaPipe depends on OpenCV for video and FFMPEG for audio data handling. It also has other dependencies like OpenGL/Metal, TensorFlow, etc.

Chapter Five

5 Previous Solutions

5.1 Traditional Solutions

5.1.1 Hearing Aids:

History: The first electric hearing aid was invented in 1898 by Miller Reese Hutchison. His design used an electric current to amplify weak signals. In 1913, the world was introduced to the first commercially manufactured hearing aids. These devices were cumbersome and not very portable. In the 1920s vacuum-tube hearing aids were produced; these tubes were able to turn speech into electric signals and then the signal itself was amplified. **[25]**

Figure 33 : Hearing Aids

What is hearing Aids?

A hearing aid is a battery-powered electronic device designed to improve your hearing. Small enough to wear in or behind your ear, they make some sounds louder. They may help you hear better when it's quiet and when it's noisy. It works by using:

- A microphone picks up sound around you.
- An amplifier makes the sound louder.
- A receiver sends these amplified sounds into your ear. **[26]**

Disadvantages of Hearing Aids:

1. Hearing aids require an adjustment period that may take several months. Follow up visits with the licensed hearing aid dispenser may be needed to take full advantage of the hearing aids.
2. Hearing Aids have batteries with various battery life so the deaf person must change batteries regularly.
3. These small hearing aids aren't suitable for people with severe, more advanced hearing loss.
4. Hearing aids can be expensive. **[27,28]**

5.1.2 cochlear implant:

History: The first cochlear implant was performed by Professor Kurt Burien in Vienna on December 16, 1977. A few months later a 48-year-old patient denoted MC-1 in was implanted in Melbourne in 1978.5 Since that implantation, advances in medical technology have led to continual miniaturization of the components.

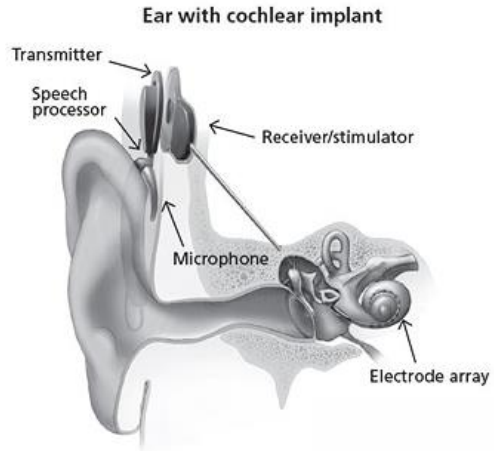


Figure 34 cochlear implant

What is a cochlear implant?

A cochlear implant is a small, complex electronic device that can help to provide a sense of sound to a person who is profoundly deaf or severely hard-of-hearing.

The implant consists of an external portion that sits behind the ear and a second portion that is surgically placed under the skin (see figure). **[31]**

An implant has the following parts:

1. A microphone, which picks up sound from the environment.
2. A speech processor, which selects and arranges sounds picked up by the microphone.
3. A transmitter and receiver/stimulator, which receive signals from the speech processor and convert them into electric impulses.
4. An electrode array, which is a group of electrodes that collects the impulses from the stimulator and sends them to different regions of the auditory nerve.
5. An implant does not restore normal hearing. Instead, it can give a deaf person a useful representation of sounds in the environment and help him or her to understand speech.

The Difference between cochlear implant and Hearing Aids:

A hearing aid is also a medical device for hearing loss. But unlike a cochlear implant, it doesn't transmit sound signals via electrodes.

Instead, hearing aids use a microphone, amplifier, and speaker to make sounds louder. This can help you hear things better.

Also, hearing aids aren't surgically implanted. They're worn inside or behind the ear.

Hearing aids are typically ideal if you have mild to moderate hearing loss. The device's level of amplification depends on your degree of hearing loss.

Certain hearing aids may help severe hearing loss, but sometimes they still won't benefit speech understanding. In this case, a cochlear implant might be the better choice. **[32]**

Disadvantage of cochlear implant:

1. Losing remaining natural hearing in the ear with the implant.
2. Regularly recharging batteries or using new ones.
3. Cochlear implants require ongoing maintenance.
4. It can be prohibitively expensive.
5. Damage to the implant during sports activity or accidents.

5.2 Modern Solutions:

5.2.1 Hardware gloves

The basic idea dates to the 1980s, when researchers started exploring how humans could interact with computers using gestures. In 1983, a Bell Labs engineer named Gary Grimes invented a glove for data entry using the 26 manual gestures of the American Manual Alphabet, used by speakers of American Sign Language. But the first glove intended to make interactions between deaf and non-deaf people easier was announced in 1988 by the Stanford University researchers James Kramer and Larry Leifer. It was called the "talking glove".

1. **Wearable-tech gloves:**

The system includes a pair of gloves with thin, stretchable sensors that run the length of each of the five fingers. These sensors, made from electrically conducting yarns, pick up hand motions and finger placements that stand for individual letters, numbers, words and phrases. The device then turns the finger movements into electrical signals, which are sent to a dollar-coin–sized circuit board worn on the wrist. The board transmits those signals wirelessly to a smartphone that translates them into spoken words at the rate of about a one word per second.

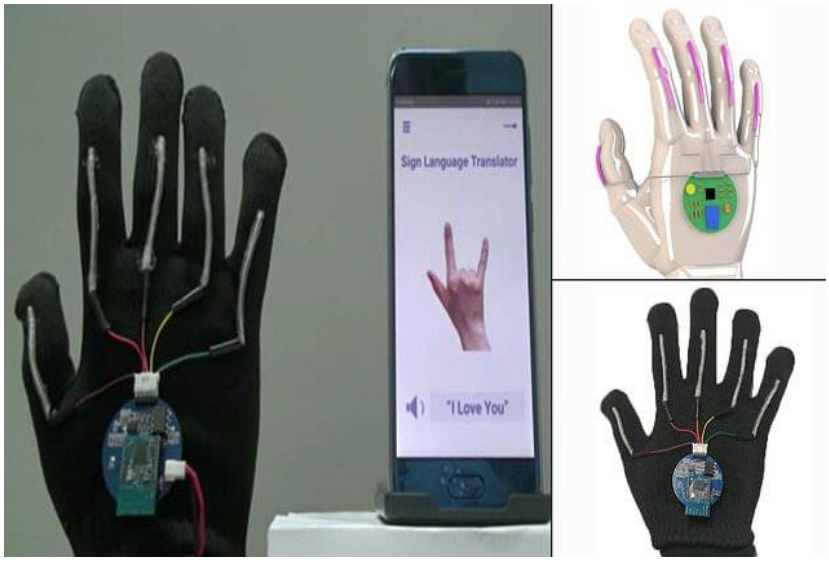


Figure 35: Wearable-tech glove

2. SignAloud Gloves:

"SignAloud," is a pair of gloves that can recognize hand gestures that correspond to words and phrases in American Sign Language. Each glove contains sensors that record hand position and movement and send data wirelessly via Bluetooth to a central computer. The computer looks at the gesture data through various sequential statistical regressions, like a neural network. If the data match a gesture, then the associated word or phrase is spoken through a speaker. **[29]**

Disadvantages of Hardware Gloves:

1. Hardware gloves would not have access to facial expressions which is an important feature in sign language.
2. Hardware costs a lot of money so this solution is so expensive.
3. It is not comfortable to deaf people to wear gloves, especially on summer days.

5.2.2 7DeepASL

DeepASL, uses a camera device to capture hand motions, then feeds the data through a deep learning algorithm, which matches it to signs of ASL. Unlike many previous devices, DeepASL can translate whole sentences rather than single words, and doesn't require users to pause between signs. DeepASL can help people who are deaf and hard of hearing by serving as a real-time translator. It could be especially useful in emergency situations.

5.2.3 WeCapable website

WeCapable Tool easily converts English text into sign language symbols. This tool will be very useful for both teaching and learning American sign language.

Translation of text to sign language is also given as a task during sign language study sessions. This tool can easily produce the correct answers and because the visual stays on screen, students can follow the hand movements at their own pace.

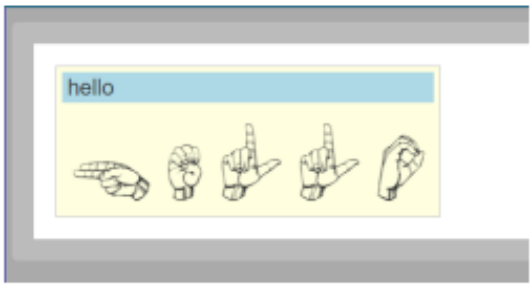


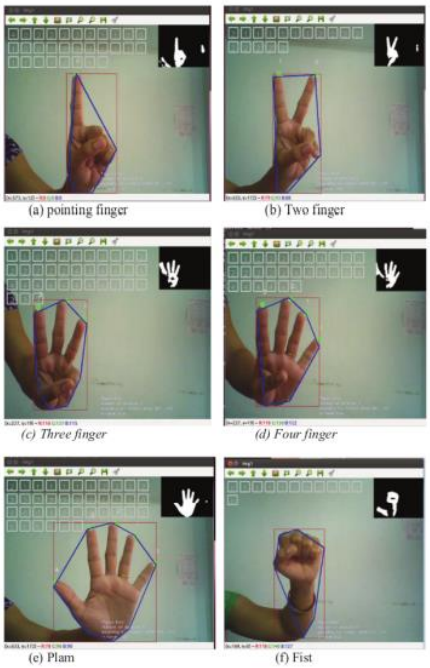
Figure 36: WeCapable website

5.3 Contour detection approach

Contour detection has probably been the approach most commonly used in the past since it is one of the first technologies that gave the possibility of object detection. Moreover, before MediaPipe, if one was supposed to rely on open-source technology, this could only be accomplished with OpenCV. This is a programming library developed by Intel and it focuses on real time computer vision.

The contour detection generates a series of problems. First of all, it is not possible to detect fingers that are behind other fingers. Second, if we have forms like a fist, we can only detect the form of the fist, but not the position of each finger. In addition, the processing time of each video frame is slower. As it can be seen in Gurav, only clear and steady signs can be detected and according to the author, the maximum speed for these simple signs is 30 fps (note: this is with some specific computer properties and the use of only one hand). Furthermore, this only works with a plain contrasted background and with the hand kept close to the camera.

Another paper that studies the possibility of real time hand detection with OpenCV is Mittal. that when the half top part of the body appears, the detection becomes less accurate and although the paper claims to be able to detect multiple hands in an image, it doesn't detect its orientation or fingers, only its position in the image.



(a) pointing finger (b) Two finger

(c) Three finger (d) Four finger

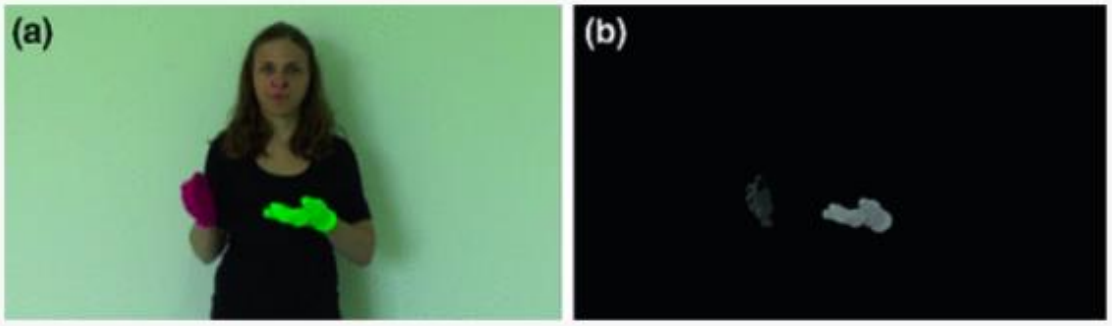
(e) Plam (f) Fist

5.4 RNN combined with another AI approach

This is a very similar approach to the one used in this paper because of the use of a RNN. RNNs are not enough to recognize sign language, they only serve for the time aspect. Therefore, a space recognition AI is still required. For example, MediaPipe is the technology that provides the space tracking of the hands for this research using CNNs.

The combination of space and time recognition allows the understanding of movement by a machine, which is what is needed to recognize sign language.

In Masood, the authors apply CNN to detect the position of the hands and RNN to detect patterns of the signs through time. In this paper, the authors accomplish the recognition of 46 signs with an accuracy of 95.2 percent. The results are very good but are only possible with some approaches that eliminate the possibility of real time recognition: In order to detect the hands, the authors remove all the background so only the hands appear in the image. The quality of the camera used by the authors is not specified.



Another interesting paper is Liu from the University of Science and Technology of China, which studies different approaches of LSTM (see LSTM in chapter 3) with RNNs. After comparing LSTMs with methods used by other papers, the authors claim that an LSTM based model works better than Hidden Markov Models (HMM).

In below table can be seen the comparison made by the authors with a large database. Both LSTM approaches show better results than the other models tried in other papers. Another interesting paper is Liu [15] from the University of Science and Technology of China, which studies different approaches of LSTM with RNNs. After comparing LSTMs with methods used by other papers, the authors claim that an LSTM based model works better than Hidden Markov Models (HMM). In table can be seen the comparison made by the authors with a large database. Both LSTM approaches show better results than the other models tried in other papers.

Method	Features	Accuracy
Normal HMM	Normal skeleton joints	0.332
CM_VoM [5]	MLS	0.576
TM_HMM [14]	Shape Context	0.673
LSTM_fc1	Normal skeleton joints	0.856
LSTM_fc2	Normal skeleton joints	0.862

5.5 Glove approach

Previous to this paper, hand tracking with gloves has been studied by Toti Moragas, Aniol Sol´e and the author of this paper, Antonio Dom`enech. This approach consists of putting a glove with sensors to the person that is signing in order to recognize the movement of the hands and display on a screen what the user is saying.

Each finger of the gloves has a flex-sensor which detects if the finger is stretched or bent, and a gyroscope on each hand to detect its orientation.

For each sign, the values of all sensors are stored in a database in order to later train a classification model with machine learning. For a little database of approximately 200 examples, the results in below table were obtained depending on the classification model. Only signs without movement are recognized in this research, the addition of movement requires of a larger database.

Classification Model	Average accuracy
K Nearest Neighbours	71.11 %
Centroids	38.27 %
Gaussian Naïve Bayes	66.30 %
Linear Discriminant Analysis	58.89 %
Decision trees	70.12 %
Bagging with K Nearest Neighbours	72.22 %
Random Forest	77.40 %
AdaBoost	8.89 %
Gradient Boosting	70.74 %
Support Vector Machines	74.69 %

The glove approach has been studied many times before with different technologies. It's actually one of the first approaches studied since it doesn't rely on computer vision which is computationally demanding.

Chapter Six

6 Mobile Application

The mobile Application helps deaf people understand blind or normal people by converting speech into video with sign language developed with Java Programming Language.

6.1 Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA . On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for all Android devices
- Apply Changes to push code and resource changes to your running app without restarting your app
- Code templates and GitHub integration to help you build common app features and import sample code
- Extensive testing tools and frameworks
- Lint tools to catch performance, usability, version compatibility, and other problems
- C++ and NDK support

6.2 What Is Java:

Java is a popular programming language. It was created in 1995 and is *operated by Oracle*. Java runs on more than 3 billion computers. It is used for **mobile app development** (especially Android apps), the development of web applications, web servers and application servers, **game development**, and database association. **Java** is the common option for building high-performance mobile applications nowadays.

As per the result of the recent State of The Developer Nation report by *Slashdata*, the total number of Java developers is currently slightly over 8 million, with about 0.5 million new coders per year being part of the Java community. For example, dedicated **mobile app developers in Dallas** have a mastery range from inheritance Java programming language to present day full stack undertaking app development

6.3 How Is Java Beneficial for Android App Development?

6.3.1 Object Oriented Programming

The primary benefit of using Java for android development is that it provides the concepts of OOPS (Object Oriented Programming) and is more proficient because they are extensible, scalable and adaptable.

A rich library of default design patterns and other best practices comes with it. It is more adaptable to the growth of mobile apps because it is open source. This permits you to make modular projects and reusable code. When building up Java android applications, web and app developers can use the source codes to modify the application coding according to their prerequisite.

6.3.2 Open -source Programming Language

This amazing programming language also provides a great collection of open-source libraries that essentially reduce the total cost of creating applications and speed up the procedure. Java is a high-level programming language, which means it resembles human language quite closely. High-level languages must be translated using assembler or translators, unlike low-level languages that mimic machine code. This simplifies Java android application development, and making it much easier to write, read, and maintain a language.

It has been reported that the Java for android development has dominated the most relevant part of the market by offering Java developers 26,269 opportunities to work on.

6.3.3 Powerful Development Tools

Java app development also comes with a collection of excellent programming tools, making it much easier for app developers to work on it and construct an application that suits your needs perfectly.

Java app development comes with few most popular development tools, including Eclipse, Netbeans, and many more. Such powerful tools play a crucial role in making Java programming language the first choice of Java android app development companies.

The robust tool sets not only assist you in coding, but also give you the ability to influence debugging, which is vital for real-world mobile application development.

6.3.4 Community Support to Developers

Android developers can acquire genuine bits of knowledge of this programming language from the similar developers in their community and are ready to grow their network broadly. Its constructive developer group shares the insights and associated data to help beginners enhance their Java app development skills.

In fact, regardless of whether you need help to fix a problem or to hold a conversation on the same note, the group of the expert and experienced developers are always happy to hear from you.

6.3.5 Independent and compatible platform

Java is a platform-independent programming language for multiple operating systems, and therefore Java is responsible for creating ideal android apps for broad parts of the developer community. With its independent nature, Java app development has gained popularity since 1990 as a mobile application development platform. And other supporting factors have recently rendered it a widely valued technology.

In other developing languages, a platform-independent feature is not popular. Thus, since Java app development has opened doors for many new technologies, it has accomplished the tagline "*Write Once Run Anywhere*" in a real sense.

6.3.6 Easy and learn language

The main reason for choosing Java app development is that it has a steep learning curve for building mobile applications. In the case of most professional programs, it is difficult to get this programming language productive in a short period of time.

Like Basic English language, Java has familiar English punctuation with least excellent characters. Generics, for example, have angle brackets that make coding easy to read and comprehend.

Moreover, Java is free to get started with, you don't have to spend pennies to create Java based mobile apps. You can outsource mobile app development to build Java android apps at the lowest cost.

.

6.3.7 Builds Robust and Secure Mobile Applications

Security is an integral part of any mobile app design. The compiler, interpreter, and runtime environment – all have been developed in the Java programming language keeping security in mind.

Robust means stability and reliability. Java puts a lot of focus on testing for early potential errors, as Java compilers can identify several issues that could occur during first execution time in other languages. Due to the robustness, ease of use, cross-platform development capabilities, and security features, Java app development has become the first choice for the provision of Internet solutions worldwide.

6.3.8 Low Investment

The return on investment is what we need, and it is possible with the executed mobile app's success rate. With a cost-effective **Java mobile app development agency** which supports all your requirements, building high-performance apps on low investment is possible.

For your mobile application development requirements, you should hire the best Android app developers, who can build well-suited app for your business.

6.3.9 The Bottom Line

For all these reasons, we may conclude that Java programming is the best choice for mobile application development. You can go with the best **Java application development company** for building secure, robust, and feature-rich mobile apps.

Chapter Seven

7 Hardware

Without hardware, there would be no way of running the essential software that makes such components so useful. Software is defined as the virtual programs that run on computers; that is, operating system, internet browser, word-processing documents, etc.

So, we decided using a hardware component (A Raspberry Pi) with Raspberry Pi Camera then upload our software on it to make interfacing easier

7.1 What is a Raspberry Pi

The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games.

What's more, the Raspberry Pi has the ability to interact with the outside world and has been used in a wide array of digital maker projects, from music machines and parent detectors to weather stations and tweeting birdhouses with infra-red cameras.

7.2 Why Raspberry Pi

Many companies offer Raspberry Pi–focused peripherals that make it much easier to tackle otherwise impossible projects because you don't have to waste time figuring out how to do basic things like adding a touchscreen. You just buy the thing, plug it into the Pi, and get to making something cool. The official Raspberry Pi camera module, for example, lets you add a camera to the Pi and use it as a point-and-shoot or a surveillance camera. So, if you've ever wanted to build something for your own home because it doesn't exist, there's a reasonable chance that a Raspberry Pi and a couple of accessories can do the job as will it has low cost and small size.

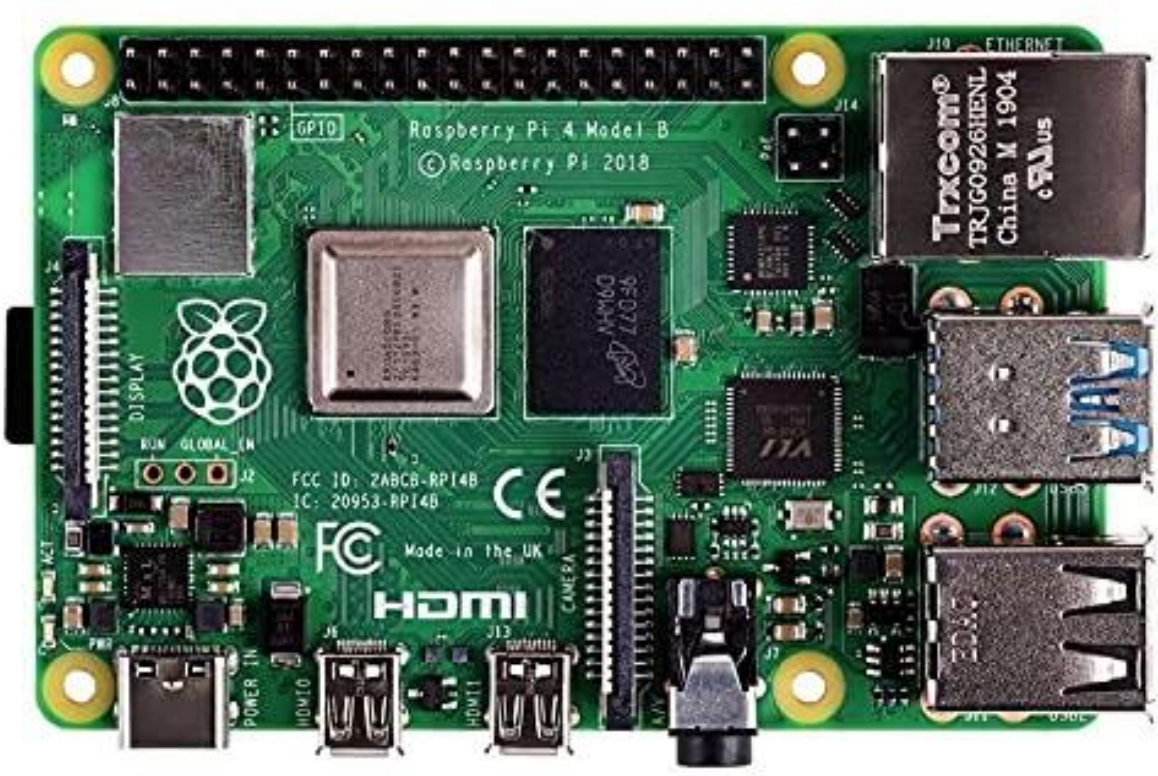


Figure 37: Raspberry Pi

7.3 Which Raspberry Pi should you choose?

There are a number of different models and versions of the Raspberry Pi computer. But which one is best for your project?

Table 1: summary of the main models available

MODEL	SPECIFICATION
PI 4	**CPU**: 1.5GHz 64-bit quad-core ARMv8 **RAM**: up to 8GB, depending on model **USB**: 2 USB 2 ports, 2 USB 3 ports **Connectivity**: Ethernet, Wi-Fi, Bluetooth
PI 3	**CPU**: 1.2GHz 64-bit quad-core ARMv8 **RAM**: 1GB **USB**: 4 ports **Connectivity**: Ethernet, Wi-Fi, Bluetooth
PI 2	**CPU**: 900MHz 32-bit quad-core ARMv8* **RAM**: 1GB **USB**: 4 ports **Connectivity**: Ethernet
PI 1	**CPU**: 700MHz 32-bit single-core ARMv6 **RAM**: 512MB **USB**: 4 ports **Connectivity**: Ethernet
PI ZERO	**CPU**: 1GHz 32-bit single-core ARMv6 **RAM**: 512MB **USB**: 1 micro-USB OTG port **Connectivity**: None

7.4 Raspberry Pi Specification

Raspberry Pi specifications differ from model to another. In this section we will focus on general

- **USB ports** — these are used to connect a mouse and keyboard. You can also connect other components, such as a USB drive.
- **SD card slot** — you can slot the SD card in here. This is where the operating system software and your files are stored.
- **Ethernet port** — this is used to connect Raspberry Pi to a network with a cable. Raspberry Pi can also connect to a network via wireless LAN.
- **Audio jack** — you can connect headphones or speakers here.
- **HDMI port** — this is where you connect the monitor (or projector) that you are using to display the output from the Raspberry Pi. If your monitor has speakers, you can also use them to hear sound.
- **Micro USB power connector** — this is where you connect a power supply. You should always do this last, after you have connected all your other components.
- **GPIO ports** — these allow you to connect electronic components such as LEDs and buttons to Raspberry Pi.

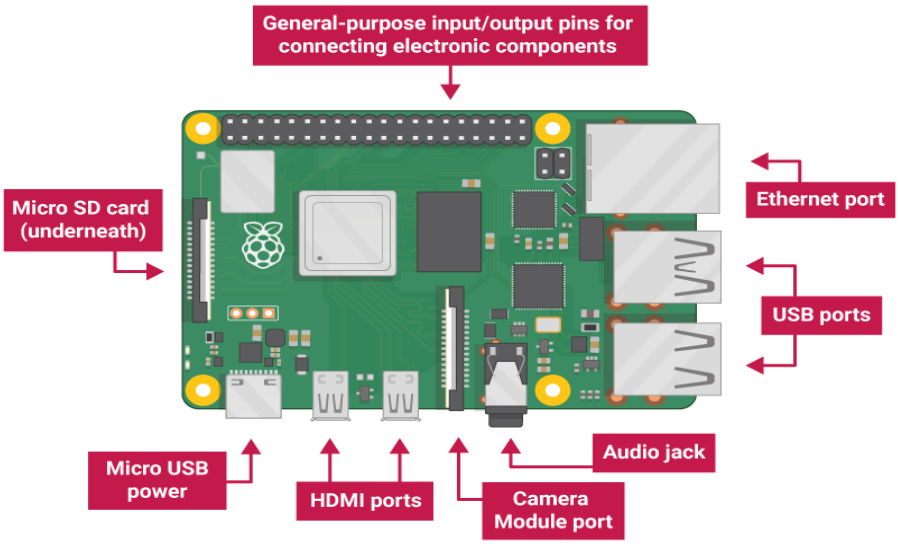


Figure 38 Raspberry Pi Specifications

7.5 Raspberry Pi Camera Module

There are two versions of the Camera Module:

- **The standard version**, which is designed to take pictures in normal light.
- **The NoIR version,** which doesn't have an infrared filter, so you can use it together with an infrared light source to take pictures in the dark.

7.5.1 Raspberry Pi NoIR Camera Module

The Raspberry Pi NoIR Camera Module v2 is a high quality 8-megapixel Sony IMX219 image sensor custom designed add-on board for Raspberry Pi, featuring a fixed focus lens. It's capable of 3280 x 2464-pixel static images, and supports 1080p30, 720p60 and 640x480p60/90 video. It attaches to Pi by way of one of the small sockets on the board upper surface and uses the dedicated CSi interface, designed especially for interfacing to cameras. The board itself is tiny, at around 25mm x 23mm x 9mm. It also weighs just over 3g, making it perfect for mobile or other applications where size and weight are important. It connects to Raspberry Pi by way of a short ribbon cable. The high-quality Sony sensor itself has a native resolution of 8 megapixel and has a fixed focus lens on-board. In terms of still images, the camera is capable of 3280 x 2464-pixel static images, and supports 1080p30, 720p60 and 640x480p90 video.

7.5.2 Why Raspberry Pi Camera

The Pi camera board does not use a USB port and is directly interfaced to the Pi. So, it provides better performance than a webcam in terms of the frame rate and resolution. We can directly use the pi camera module in Python to work on images and videos.

7.6 Get started with Raspberry Pi

To start using a Raspberry Pi you will need some hardware components and a Raspberry Pi OS installed on RPI.

7.6.1 What you will need

- A Raspberry Pi computer with an SD card or micro-SD card.
- A monitor with a cable (and, if needed, an HDMI adaptor).
- A USB keyboard and mouse.
- A power supply.
- Raspberry Pi OS installed using the Raspberry Pi Imager.

7.6.2 Raspberry Pi Imager

Raspberry Pi needs an operating system to work. This is it. Raspberry Pi OS (previously called Raspbian) is our official supported operating system.

Raspberry Pi Imager is the quick and easy way to install Raspberry Pi OS and other operating systems to a microSD card, ready to use with your Raspberry Pi.

7.6.3 Connections

Start finding the USB connector end of your mouse's cable and connect the mouse to a USB port on your Raspberry Pi (it doesn't matter which port you use), Connect the keyboard in the same way. Use a cable to connect the screen to the Raspberry Pi's HDMI port — use an adapter if necessary. If your screen has speakers, your Raspberry Pi can play sound through these. Or you could connect headphones or speakers to the audio port. Check the slot on the underside of your Raspberry Pi to see whether an SD card is inside. If no SD card is there, then insert an SD card with Raspbian installed. Your Raspberry Pi then boots up into a graphical desktop.

Chapter Eight

8 Proposed Methods

8.1 System Overview

As said, everyone has the right to communicate with everyone. Here Blind-Deaf system comes the make communication between Blinds and Deaf easier. System will contain two parts

1. Glasses for Blind person
2. Mobile App for Deaf person

- First way of communication (Glasses):

As Blind person want to understand Deaf person that communicate with sign language here role of glasses comes. Glasses contain (camera, microcontroller, headphone), it will translate sign language into speech that Blind can hear it

- Second way of communication (Mobile Application):

As Deaf person want to understand Blind person that communicate with voice here role of Mobile App comes.it will translate voice into Video with sign language that Deaf can see.

In this chapter the two ways of communication will be discussed in detail.

8.2 System Diagrams

8.2.1 System Block Diagram

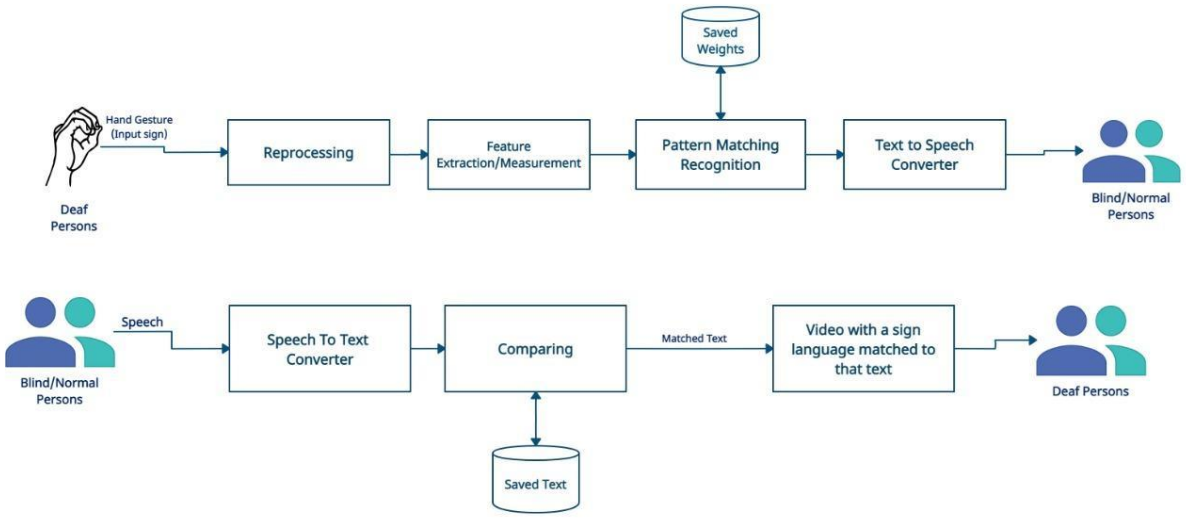


Figure 39: System Block Diagram.

8.2.2 System Class Diagram

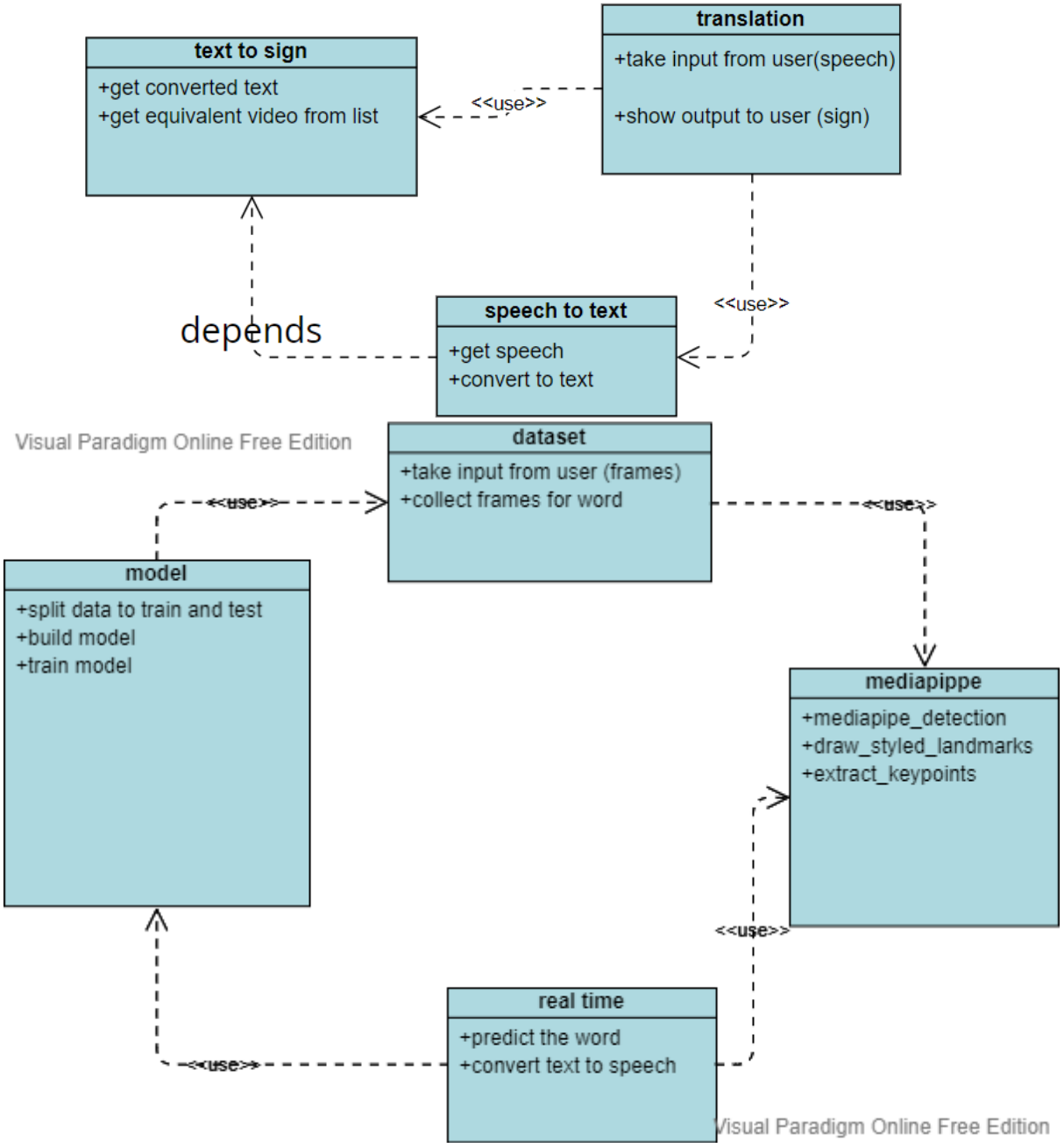


Figure 40: System Class diagram.

8.2.3 System Sequence Diagram

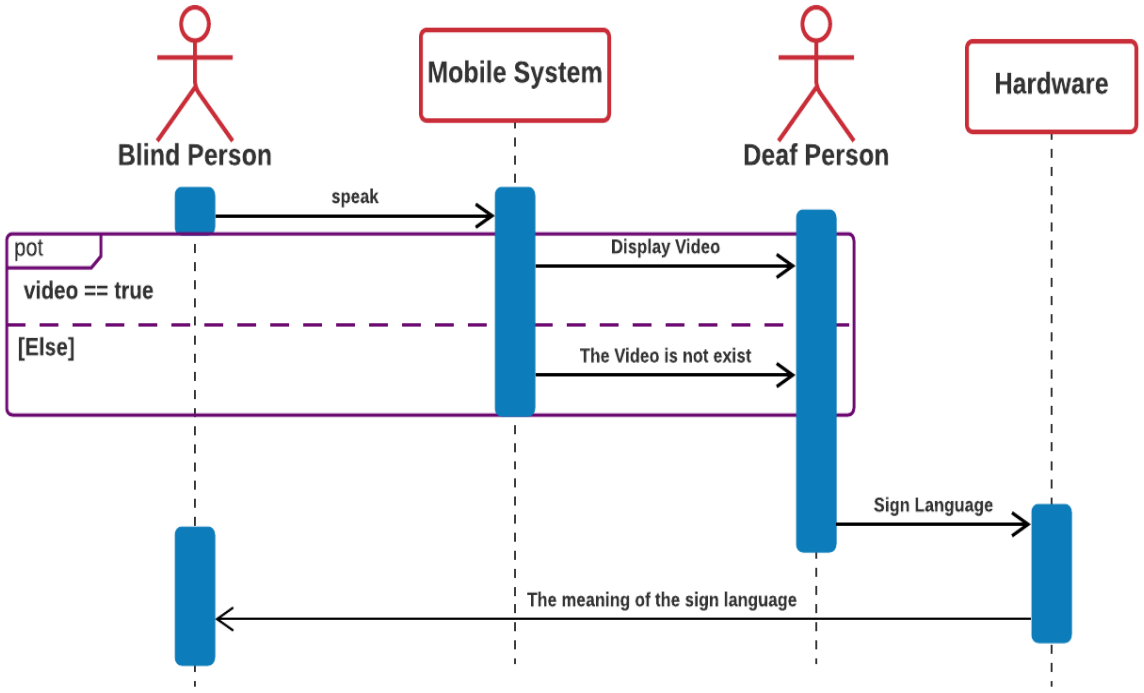


Figure 41: System Sequence Diagram.

8.2.4 System Use Case Diagram

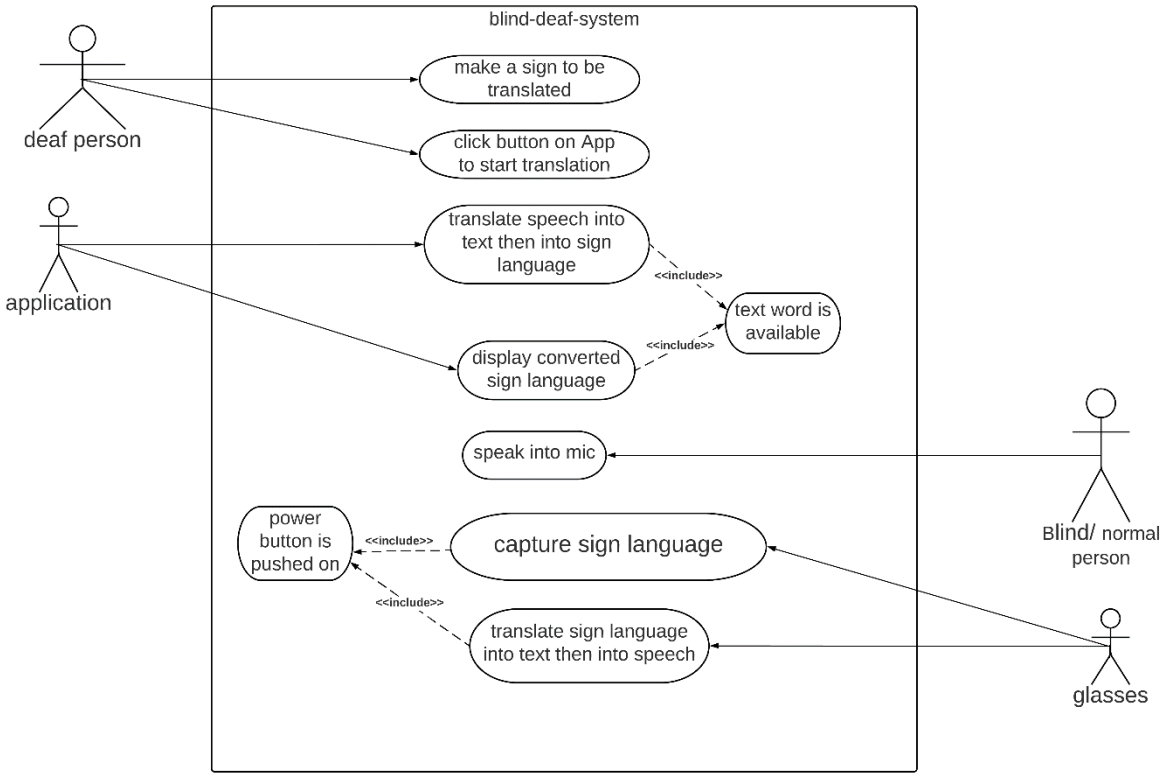


Figure 42: System Use Case Diagram.

8.3 ASL To Text | speech

8.3.1 Computer vision phase 1

object detection

Object detection is a computer vision technique that allows us to identify and locate objects in an image or video.

Object detection draws bounding boxes around these detected objects, which allow us to locate where said objects are in.

Deep learning-based object detection models typically have two parts. An encoder takes an image as input and runs it through a series of blocks and layers that learn to extract statistical features used to locate and label objects. Outputs from the encoder are then passed to a decoder, which predicts bounding boxes and labels for each object. **[34]**

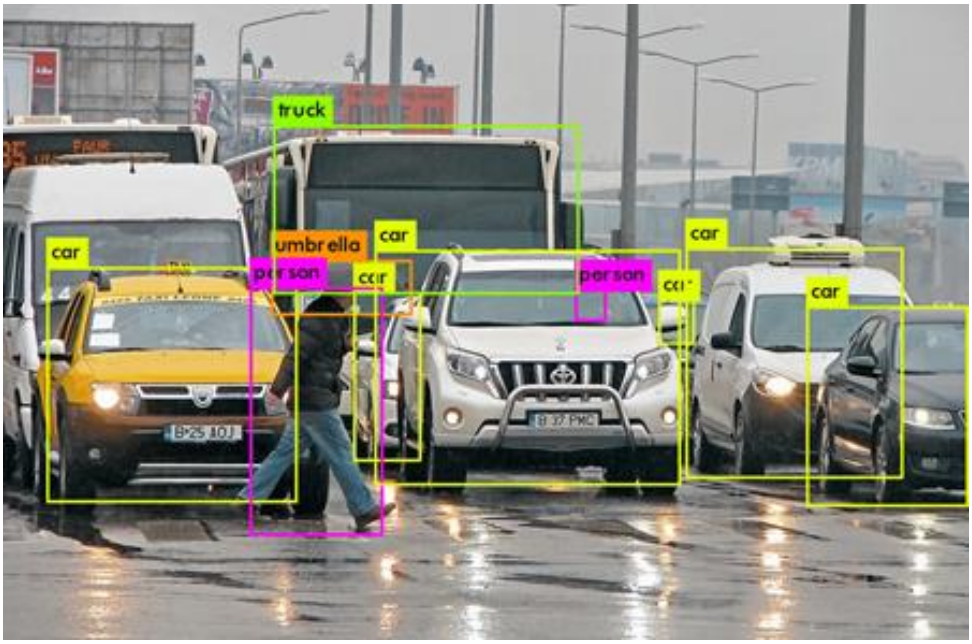


Figure 43: Object Detection

1. **Model**

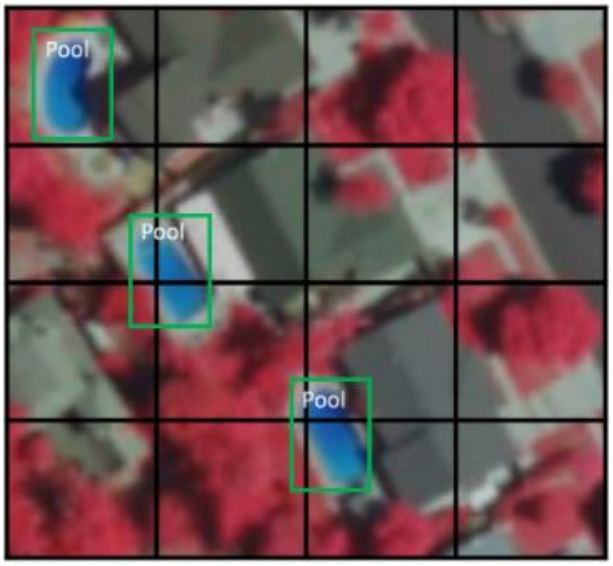
How objected detection SSD model work?

The computer vision algorithm we used is **Single shot detectors (SSDs)**

Single shot detectors (SSDs) rely on a set of predetermined regions. It uses a fully convolutional approach in which the network can find all objects within an image in one pass (hence ‘single-shot’).

SSD has two components: a backbone model and SSD head. Backbone model usually is a pre-trained image classification network as a feature extractor. The SSD head is just one or more convolutional layers added to this backbone and the outputs are interpreted as the bounding boxes and classes of the objects. **[34]**

SSD divides the image using a grid and have each grid cell be responsible for detecting objects in that region of the image. Detection objects simply means predicting the class and location of an object within that region. If no object is present, we consider it as the background class and the location is ignored.



The SSD approach is based on a feed-forward convolutional network that produces a fixed-size collection of bounding boxes and scores for the presence of object class instances in those boxes, followed by a non-maximum suppression step to produce the final detections. **[33]**

2. implementation and Results

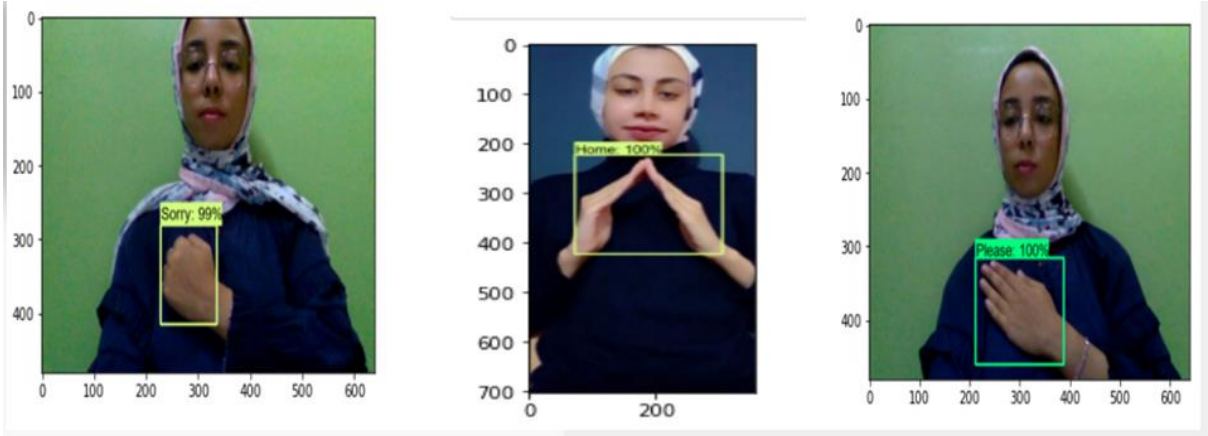


Figure 44: SSD with Object Detection.

3. Limitation:

1. It needs Too much data set for the training process.
2. Very slow in Training process.
3. Slow translation of each sign
4. bad detection of sign language for the similar signs

8.3.2 Computer vision phase 2

MediaPipe

The Google MediaPipe technology provides detailed real time finger tracking with multiple hands, face detection and multiple others detection models. In the images below it is shown how the hand detection works. **[35]**

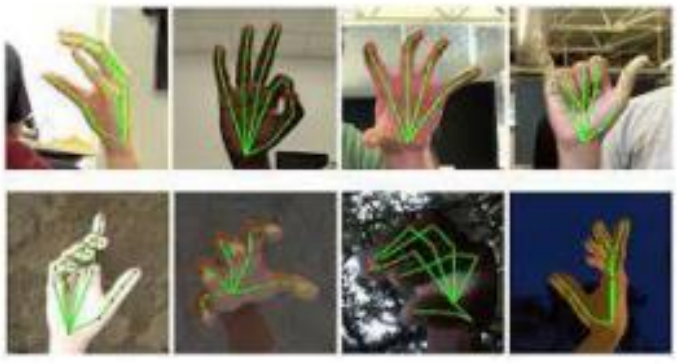


Figure 45: MediaPipe Hand Detection

MediaPipe solutions:

1. pose detection
2. hands detection
3. face detection
4. holistic solution

pose of the body consists of 33 landmarks, face consists of 468 landmark and each hand contain 21 landmarks.

How MediaPipe Work?

1. We estimate the human pose with mediapipe pose detector and extract landmarks.
2. using the inferred pose landmarks, we derive three regions of interest (ROI) crops for each hand and the face,
3. re-crop model to improve the ROI.
4. We then crop the full-resolution input frame to these ROIs and apply task-specific face and hand models to estimate their corresponding landmarks.
5. Finally, we merge all landmarks with those of the pose model to yield the full 540+ landmarks.

1. **model**

RNN: Convolutional neural networks are great for a 1 to 1 relation What CNNs cannot do is accept a sequence of vectors.

That's where Recurrent Neural Networks (RNNs) are used, RNNs allow us to understand the context of a video frame, relative to the frames that came before it. They do this by passing the output of one training step to the input of the next training step, along with the new frame.

LSTM (long short-term memory): RNN contains a problem that if number of layers is large vanishing gradient problem happens, so they solve this in LSTM

LSTM is a refined artificial recurrent neural network (RNN) architecture, unlike standard feedforward neural networks, LSTM has feedback connections, but also entire sequences of data (such as speech or video).

2. Implementation and Results

Figure 46: Media-pipe with LSTM Detection

WHY this approach (MediaPipe + LSTM)?

long-time of research, we have reached that it is the best approach because of

1. Less Data required
2. Faster to train
3. Faster in detection

8.4 Speech | Text to ASL

8.4.1 Mobile Application

Speech to text uses speech recognition technology to identify patterns in sound waves and match them to the phonemes of speech to translate them into text.

Why speech to text?

- Ease of communication.
- Time saved with increased efficiency: can transcribe even lengthy passages of text in minutes or even seconds.
- Speech to text is cheap and available.

Design of mobile Application

There are 6 Activities:

- Splash activity

Figure 47: Splash Activity

- the Home Page

home page (Main activity) has 2 buttons, the first button for the Arabic language and second button for English language.

Figure 48: Main Activity.

- the second page

The second page has button and textbox. In this activity the speech converts into text then into video with sign language after Displaying the video the Application will return automatically to this Activity. If the word is not existed, the application will display toast say that the word is not exit.

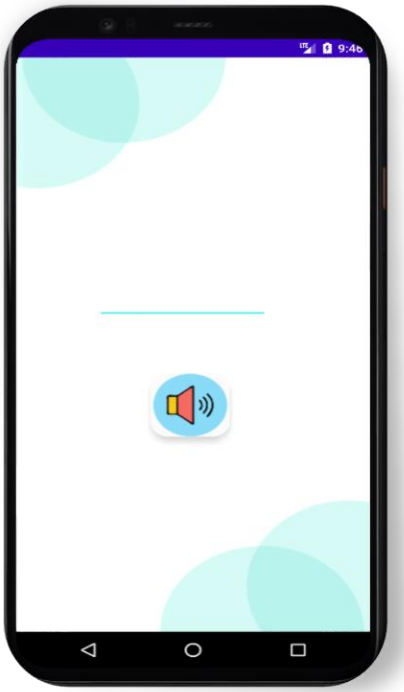


Figure 49: Speech to Text Activity

The application may display multiple videos in case the user say sentence; number of videos depend on number of words.

To Display more than video

1. create multiple objects from Video View and give them the same id.
2. create action listener for each object.

```
object.setOnCompletionListener(new MediaPlayer.OnCompletionListener
() {
    @Override
    public void onCompletion(MediaPlayer mediaPlayer) {

    }
});
```

Figure 50:Action Listener Cod

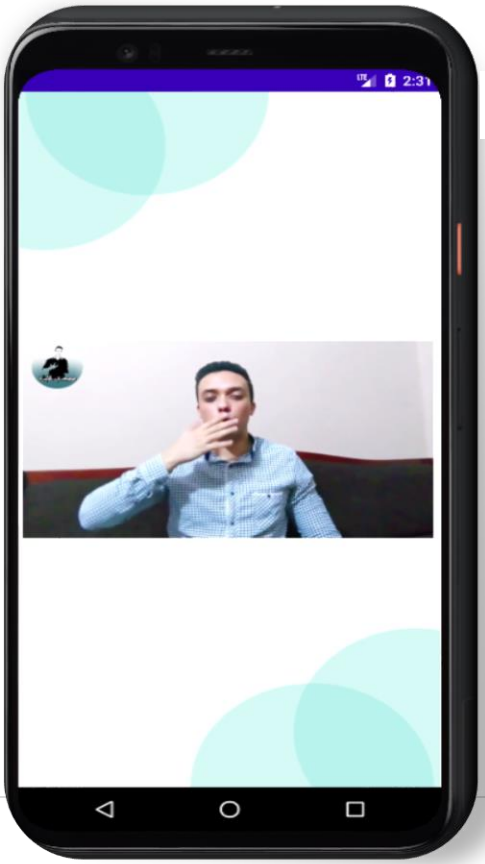
Chapter Nine

9 Results and Discussion

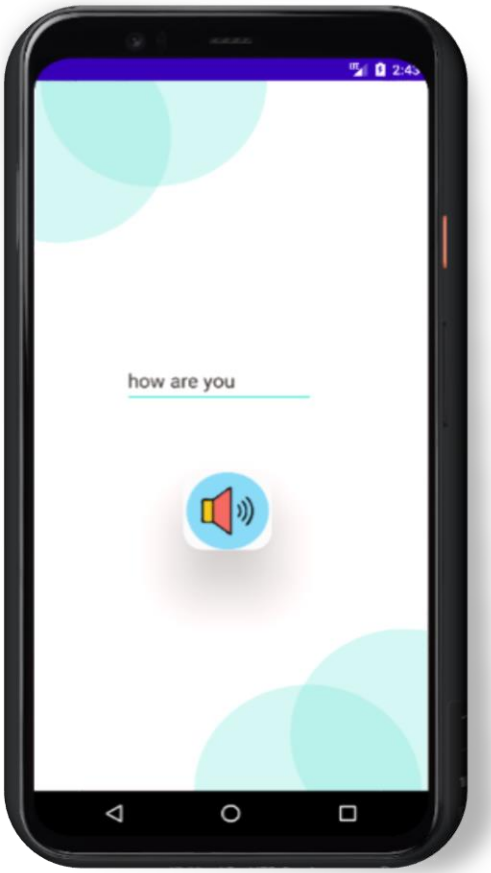
9.1 speech | text to ASL Results

9.1.1 Arabic Translation Results:

Figure 51:Arabic Translation Results



9.1.2 English Translation Results:



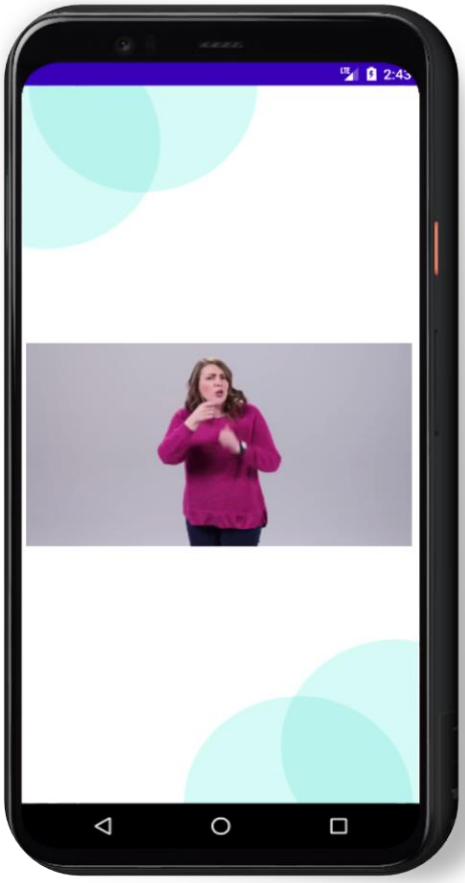


Figure 52: English Translation Results:

9.2 ASL to text | Speech Results


hello bathroom thanks hungry water


hungry thanks bathroom hello bathroo

9.2.1 Models Result

Table 2: results of different models we tried

Model id	Number of words	Epoch number	Number of videos for each word	Accuracy
1	5	300	50	92.5%
2	6	500	30	83.333%
3	6	450	30	83.333%
4	7	300	30	92%
5	7	700	30	87%
6	**11**	**1000**	**60**	**96%**
7	**11**	**1200**	**60**	**92%**
8	**11**	**800**	**30**	**88.2%**
9	5	800	30	89%
10	5	1600	30	96%

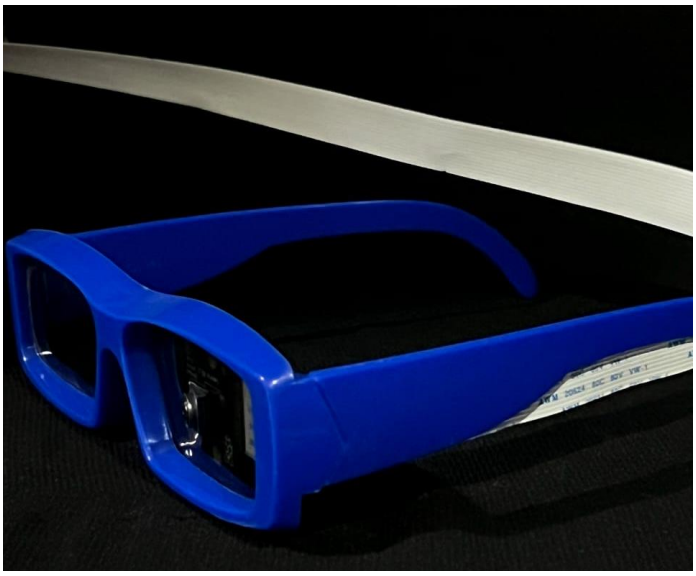
Table 3: Model number_7

Word	Right Times	Wrong Times
Hello	6	1
thanks	6	0
Yes	3	3
No	1	5
Me	5	1
Forget	4	3
repeat	1	3
Hungry	6	0
I love you	3	3
deaf	2	4
Water	4	0

Table 4: Model number_8

Word	Right Times	Wrong Times
Hello	3	0
thanks	3	0
Yes	1	2
No	0	3
Me	0	3
Forget	3	0
repeat	2	1
Hungry	3	0
I love you	2	1
deaf	1	0
Water	3	0

9.3 Hardware prototype



9.4 The used Tools

Table 5:used tools

Raspberrypi:	**Definition:** Raspberry Pi is the name of a series of single-board computers made by the Raspberry Pi Foundation, a UK charity that aims to educate people in computing and create easier access to computing education. **Features:** • low cost • credit-card sized computer • USB ports • HDMI port • Ethernet port • Linux operating system **Usage:** The main part in our hardware, we used in real time processing. **Version:** Latest version raspberry pi model B (4G RAM**)**
PiCamera:	**Definition:** Pi Camera module is a camera which can be used to take pictures and high-definition video. Raspberry Pi Board has CSI (Camera Serial Interface) interface to which we can attach PiCamera module directly. This Pi Camera module can attach to the Raspberry Pi's CSI port using 15-pin ribbon cable. **Features:** • Resolution – 8 MP • HD Video recording – 1080p @30fps, 720p @60fps, 960p @45fps and so on. • It Can capture wide, still (motionless) images of resolution 2592x1944 pixels • CSI Interface enabled. **Usage:** Connected with raspberry pi to record the sign language. **Version:** Version 2

Python Libraries:	**Definition:** A Python library is a collection of related modules. It contains bundles of code that can be used repeatedly in different programs. It makes Python Programming simpler and convenient for the programmer. As we don't need to write the same code again and again for different programs. **Used libraries:** • OpenCv • TensorFlow keras • MediaPipe • Pyttsx3 • Pygame
Anaconda:	**Definition:** Anaconda is a distribution of the Python and R programming languages for scientific computing that aims to simplify package management and deployment. **Usage:** • An open-source package and environment management system called conda, which makes it easy to: • Install/update packages and • Create/load environments. • Support multiple useful IDEs like Spyder and Jupyter. • Support machine/Deep learning libraries like TensorFlow, sklearn and keras. **Version:** Our used version is 2.1.1.

Android Studio:	**Definition:** Android studio is the official integrated development environment (IDE) for Google's android operating system. **Features:** • Instant App Run. • Visual Layout Editor. • Intelligence Code Editor. • Addition of New Activity as a Code Template. • Help to Build Up App for All devices. • Help to Connect with Firebase. **Usage:** It is used as an interface for translating. **Version:** We used the latest version 4.2.

9.5 Environment

9.5.1 For raspberry pi

1. Debian buster 32bit
2. Sd card class 10 128 GB
3. 4GB RAM

9.5.2 For mobile app

1. Windows 10 os
2. 16 GB RAM
3. Intel core-i7

9.6 Github link

https://github.com/Omar-AlQashlan/Blind-Deaf-System

Chapter Ten

10 Conclusion and Future Work

10.1 Conclusion

As we said before Blind-Deaf System (BDS) facilitates communication between Blind and deaf people or Normal and Deaf people.

This system is two-way communication system and consists of two parts:

1. **Glasses (ASL to Text | speech).**

As Blind/normal person want to understand Deaf person that communicate with sign language here role of glasses comes. Glasses contain (camera, microcontroller, headphones), it will translate sign language into speech that Blind can hear it.

2. **Mobile app (speech | Text to ASL).**

As Deaf person want to understand Blind/normal person that communicate with voice here role of Mobile App comes.it will translate voice into Video with sign language that Deaf can see.

10.2 Future Work

It will be interesting to continue what we have started in this work by collecting more signs from ESL to widen our dataset of sentences for the process of continuous classification.

Also, one of the most important steps in the future to increase the accuracy of the model.

Chapter Eleven

11 References

1. Kelly Mercer, 'Deaf History Month: Assistive Technology', April 8,2015[Online].Available:https://101mobility.com/blog/deaf-history-month-assistive-technology/
2. WHO ,' Deafness and hearing loss', 1 April 2021 [Online].Available:https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss
3. WHO,'Deafness and hearing loss Rate',1 April 2021[Online].Available:https://www.who.int/pbd/deafness/estimates/en/
4. IBM Cloud Education ,'Machine Learning ' , 15 July 2020[Online] .Available: https://www.ibm.com/cloud/learn/machine-learning
5. Developed by JavaTpoint, 'Machine learning Life cycle', 2011-2021[Online].Available: https://www.javatpoint.com/machine-leaning-life-cycle?fbclid=IwAR2QTGOYMprZxZY9-AiGxr4gt2YXWI
6. 7wData,'Types of Machine Learning Algorithms',June 5, 2020[Online].Available:https://7wdata.be/visualization/types-of-machine-learning-algorithms-2/
7. Jason Brownlee ,'4 Types of Classification Tasks in Machine Learning', August 19, 2020[Online].Available: https://machinelearningmastery.com/types-of-classification-in-machin e-learning/
8. Developed by JavaTpoint,'Unsupervised Machine Learning', 2011-2021,[Online].Available:https://www.javatpoint.com/unsupervise d-machine-learning?fbclid=IwAR2CqCwMWpdpa8HeAAyukqH2PX4nhMPchzU31H-tegNIEjSgp5IHcyZEj_U
9. Surya Priy,'Different Types of Clustering Algorithm',08 Jun, 2021 [Online].Available:https://www.geeksforgeeks.org/different-types-clustering-algorithm/
10. Wikipedia,'Reinforcement learning ',13 July 2021[Online].Available: https://en.wikipedia.org/wiki/Reinforcement_learning
11. JavaTpoint,'Association Rule Learning',2011-2021 [Online].Available: https://www.javatpoint.com/association-rule-learning
12. MARSHALL%20HARGRAVE,'Deep%20Learning',May%2017,%202021[Online].%20Available:https://www.investopedia.com/terms/d/deep-learning.asp

13. FAIZAN SHAIKH,'Deep Learning vs. Machine Learning',April 8, 2017[Online].Available:https://www.analyticsvidhya.com/blog/2017/04/comparison-between-deep-learning-machine-learning/?fbclid=IwAR2vFG4GBuduSRd7IlVl81AWC3l6xvvIHv26mWeMjCy_wwlFsgGNovOKkbk

14. DataFlair,'How Deep Learning Works with Different Neuron Layers ',2021[Online].Available:https://data-flair.training/blogs/how-deep-learning-works/

15. JAKE FRANKENFIELD,'Artificial Neural Network (ANN)',Aug 28, 2020[Online].Available:https://www.investopedia.com/terms/a/artificial-neural-networks-ann.asp

16. Aston Zhang,Zack C. Lipton,Mu Li,Alex J. Smola,'Dive into DeepLearning',May2019[Online].Available:https://d2l.ai/chapter_multilayer-perceptrons/backprop.html

17. Hamza Mahmood,'Activation Functions in Neural Networks',Dec 31,2018[Online].Available:https://towardsdatascience.com/activation-functions-in-neural-networks-83ff7f46a6bd

18. Sumit Saha,'A Comprehensive Guide to Convolutional Neural Networks',Dec15,2018[Online].Available:https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

19. Prabhu,'Understanding of Convolutional Neural Network (CNN) — Deep Learning',Mar 4,2018[Online].Available: https://medium.com/@RaghavPrabhu/understanding-of-convolutional neural-network-cnn-deep-learning-99760835f148

20. Jiwon Jeong,'The Most Intuitive and Easiest Guide for Convolutional Neural Network',Jan 24,2019[Online].Available: https://towardsdatascience.com/the-most-intuitive-and-easiest-guide-f or-convolutional-neural-network-3607be47480

21. Lilian Weng,'Object Detection for Dummies Part 3: R-CNN Family ',Dec 31, 2017[Online].Available: https://lilianweng.github.io/lil-log/2017/12/31/object-recognition-for-d ummies-part-3.html

22. The MathWorks, Inc,'Getting Started with R-CNN, Fast R-CNN, and Faster R-CNN',1994-2021[Online].Available: https://www.mathworks.com/help/vision/ug/getting-started-with-r-cnn -fast-r-cnn-and-faster-r-cnn.html

23. Aston Zhang,Zack C. Lipton,Mu Li,Alex J. Smola,'Dive into Deep Learning',May2019[Online].Available:https://d2l.ai/chapter_computer -vision/rcnn.html?fbclid=IwAR1Vm-p_5XmcPel7j2Tpt6KRluOlTKilLPsHs-dK5olTi5GazYTRyefpvCQ

24. Rohith Gandhi,'R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms',Jul 9 ,2018[Online].Available: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-obje ct-detection-algorithms-36d53571365e

25. Hearing Systems developer,'The History of Hearing Aids',February 19,2019[Online].Available:https://hearingsystemsinc.com/the-history-of-hearing-aids/

26. WebMD developer,'Hearing Aid Basics', 2005 - 2021[Online]. Available:https://www.webmd.com/healthy-aging/hearing-aids#1

27. FAD developers,'Benefits of hearing aids',January16, 2018[Online]. Available:https://www.fda.gov/medical-devices/hearing-aids/benefits-and-safety-issue

28. Debbie Clason,'Hearing aids',February 15, 2018[Online]. Available:https://www.healthyhearing.com/report/52837-The-pros-and -cons-of-small-hearing-aids

29. University of Washington,'SignALoud',April 12, 2016[Online].Available:https://www.washington.edu/news/2016/04/12 /uw-undergraduate-team-wins-10000-lemelson-mit-student-prize-for-g loves-that-translate-sign-language/

30. Matthew Chin,'Wearable-tech glove',June 29, 2020[Online]. Available:Wearable-tech glove translates sign language into speech in real time

31. NIHdeveloper,'Cochlear Implants',February 2016[Online]. Available:https://www.nidcd.nih.gov/health/cochlear-implants?fbclid= IwAR23yyK0ahUqKaQtoc2rSVideMkfDR7oErKpCaLGCloyhXjGqD 09LNoP9qw

32. Kirsten Nunez,'Cochlear Implant',February 27, 2020[Online].Available:https://www.healthline.com/health/cochlear-i mplant#surgical-procedure

33. Dec 2015 · Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg · Available:https://paperswithcode.com/paper/ssd-single-shot-multibox-detector

34.https://www.fritz.ai/object-detection/#:~:text=detection%20%E2%80%93%20the%20basics-,What%20is%20object%20detection%3F,move%20through)%20a%20given%20scene.

35.https://upcommons.upc.edu/bitstream/handle/2117/343984/ASL%20recognition%20in%20real%20time%20with%20RNN%20-%20Antonio%20Dom%C3%A8nech.pdf?sequence=1&isAllowed=y

36. MediaPipe
37. MediaPipe Python Tutorial [Install + Real-Time Hand Tracking Example] (omdena.com)
38. Word-level Deep Sign Language Recognition from Video: A New Large-scale Dataset and Methods Comparison.
39. Efficient sign language recognition system and dataset creation method based on deep learning and image processing.
40. American Sign Language Recognition using Deep Learning and Computer Vision. (2018 IEEE International Conference on Big Data (Big Data))
41. An Efficient Sign Language Translator Device Using Convolutional Neural Network and Customized ROI Segmentation (2019 2nd International Conference on Communication Engineering and Technology)
42. Research of a Sign Language Translation System Based on Deep Learning
43. Real-time Sign Language Fingerspelling Recognition using Convolutional Neural Networks from Depth map