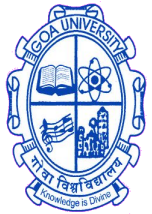


Goa Business School

MCA (Master of Computer Application)

Machine Learning LAB

Fifth Semester

Speech Emotion Recognition

Introduction

SER capitalizes on the fact that voice often reflects underlying emotion through tone and pitch. This is also the phenomenon that animals like dogs and horses employ to be able to understand human emotion. SER is tough because emotions are subjective and annotating audio is challenging.

Speech emotion recognition is one of the active researches in machine learning. There are already applications that use speech emotion recognition as its feature. Our purpose is to examine the difference in performance of model using multi-layer perceptron (MLP) and HuBert model with Mel-frequency cepstral coefficients (MFCCs) on Ravedess Dataset.

Additional we have also created our own dataset and tested it out on the model which performed best.

DataSets in Use

1. The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS).

2. Custom Dataset.

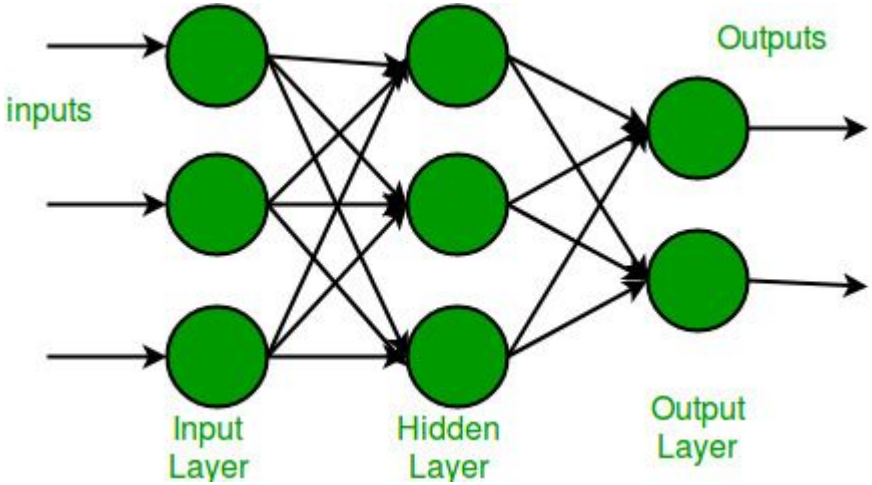
MFCC

- Preemphasis
- Windowing
- DFT (Discrete Fourier Transform)
- Mel-Filter Bank
- Applying Log
- IDFT
- Dynamic Features

Multi-layer Perceptron

MLP is fully connected dense layers, which transform any input dimension to the desired dimension.

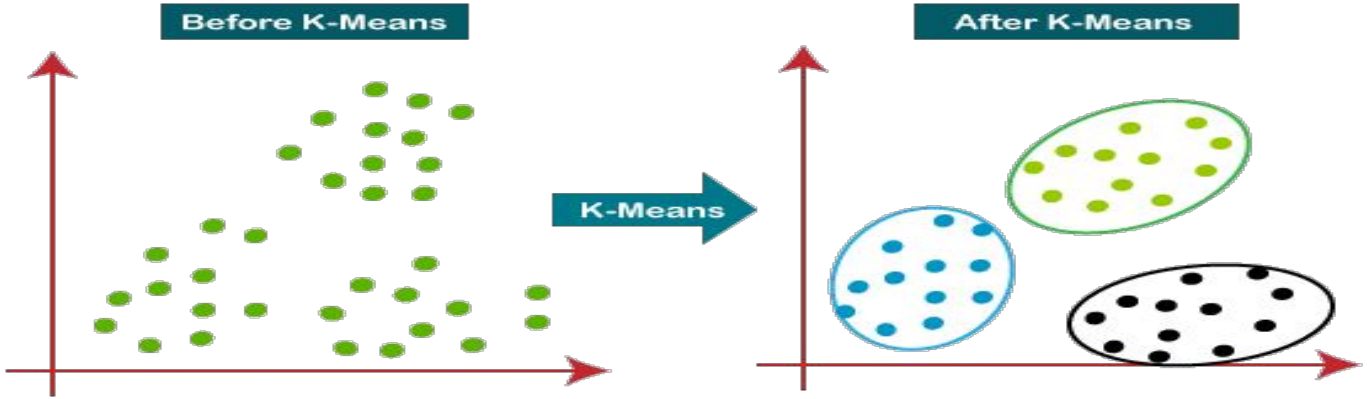
A multi-layer perception is a neural network that has multiple layers. To create a neural network we combine neurons together so that the outputs of some neurons are inputs of other neurons.



K-Means Clustering Algorithm

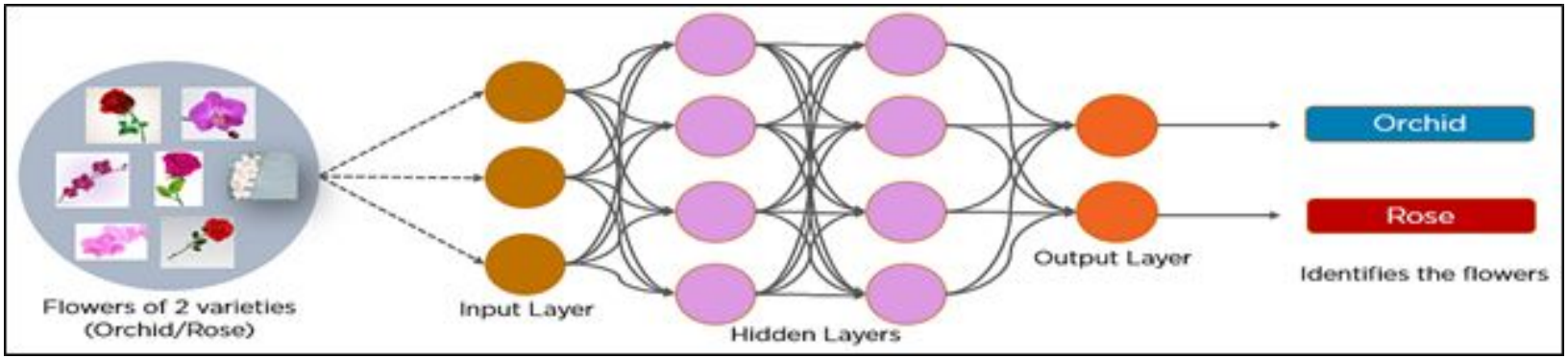
K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters.

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset

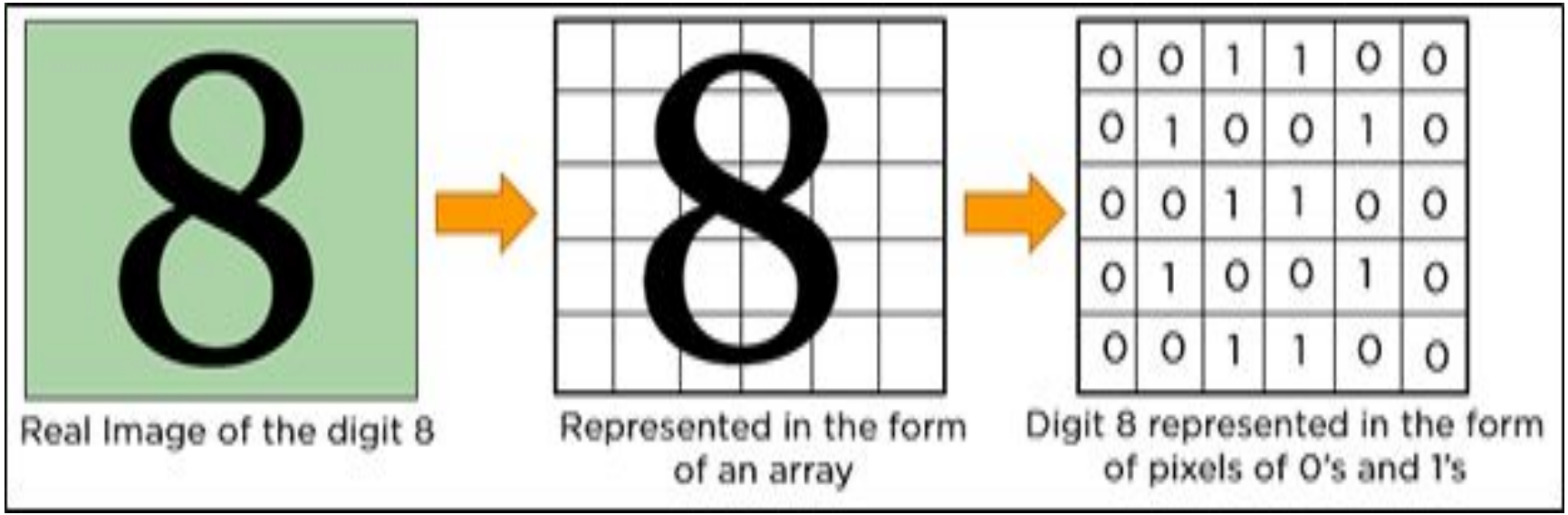


Convolutional Neural Network

- A convolutional neural network is a feed-forward neural network that is generally used to analyze visual images by processing data with grid-like topology. It's also known as a ConvNet.
- A convolutional neural network is used to detect and classify objects in an image.
- Below is a neural network that identifies two types of flowers: Orchid and Rose.



In CNN, every image is represented in the form of an array of pixel values.



Layers in a Convolutional Neural Network

A convolution neural network has multiple hidden layers that help in extracting information from an image.

The four important layers in CNN are:

1. Convolution layer
2. ReLU layer
3. Pooling layer
4. Fully connected layer

Convolution Layer

This is the first step in the process of extracting valuable features from an image. A convolution layer has several filters that perform the convolution operation. Every image is considered as a matrix of pixel values.

Consider the following 5x5 image whose pixel values are either 0 or 1. There's also a filter matrix with a dimension of 3x3. Slide the filter matrix over the image and compute the dot product to get the convolved feature matrix.

ReLU layer

ReLU stands for the rectified linear unit. Once the feature maps are extracted, the next step is to move them to a ReLU layer.

ReLU performs an element-wise operation and sets all the negative pixels to 0.

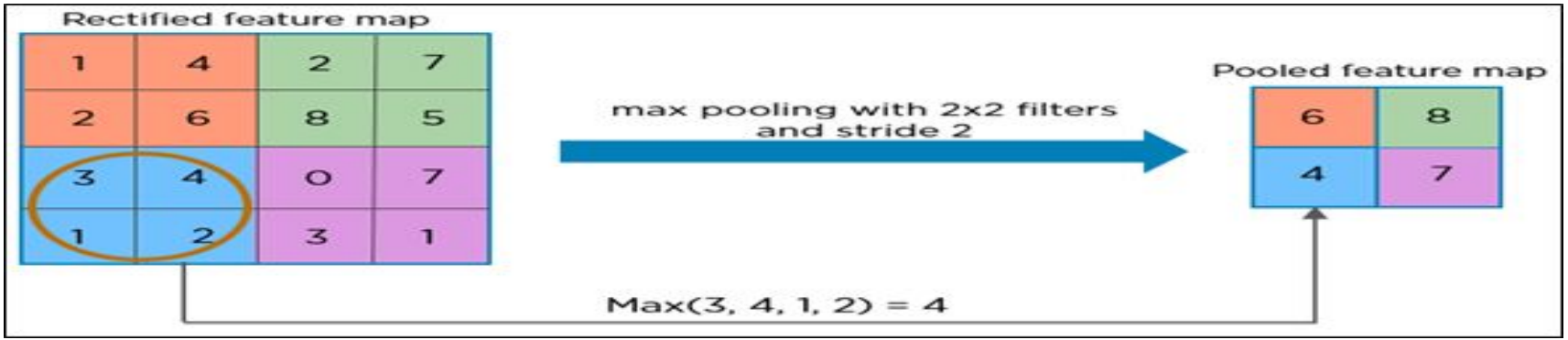
It introduces non-linearity to the network, and the generated output is a rectified feature map.

The original image is scanned with multiple convolutions and ReLU layers for locating the features.

Pooling Layer

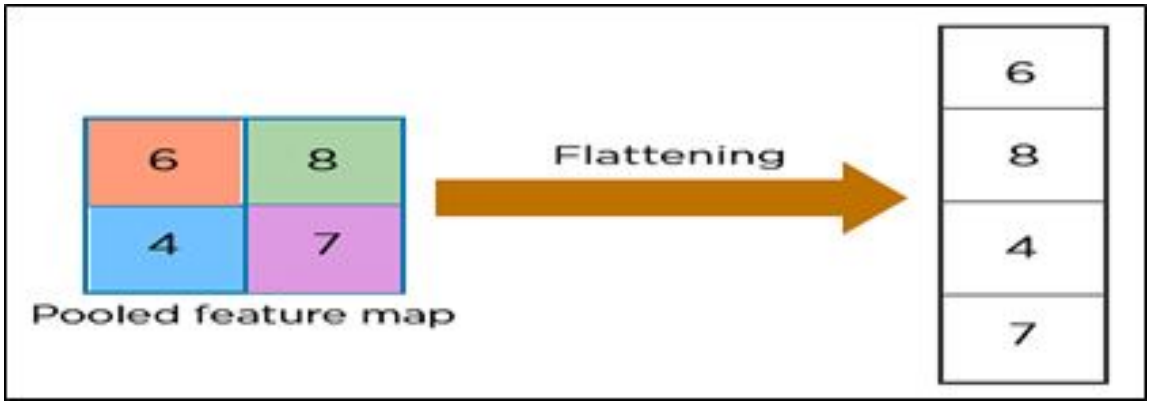
Pooling is a down-sampling operation that reduces the dimensionality of the feature map. The rectified feature map now goes through a pooling layer to generate a pooled feature map.

The pooling layer uses various filters to identify different parts of the image like edges, corners, body, feathers, eyes, and beak.

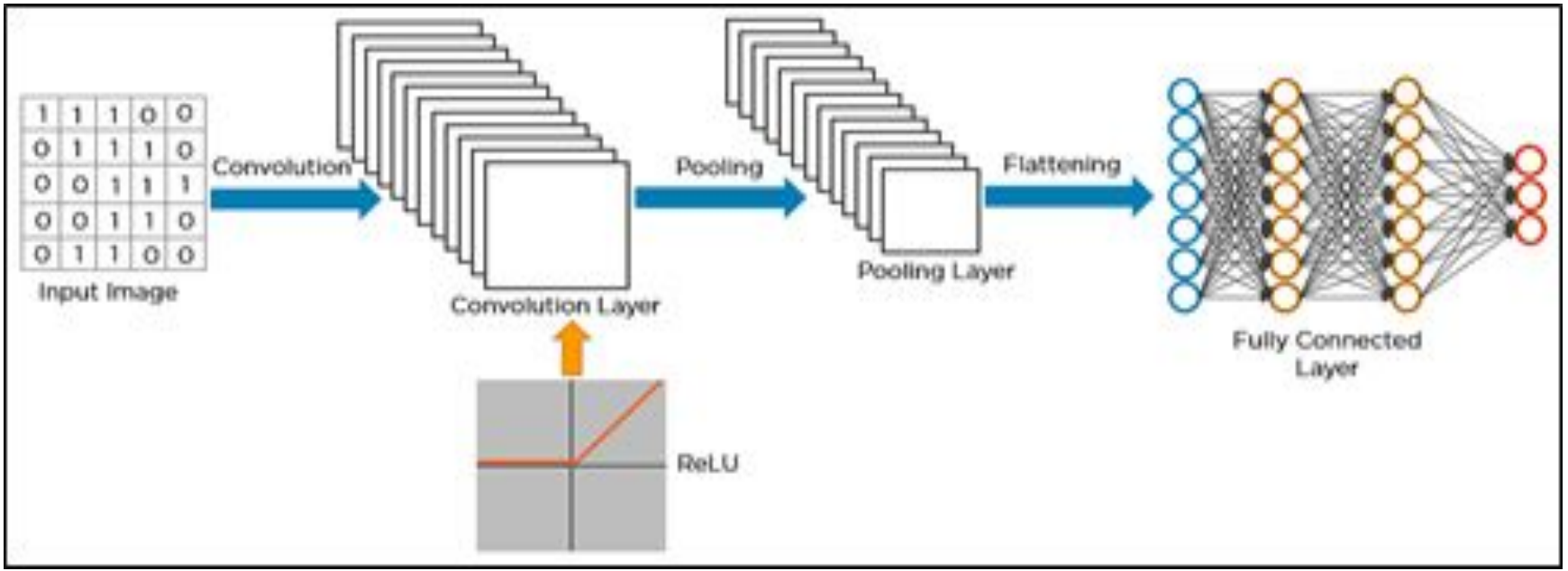


Flattening

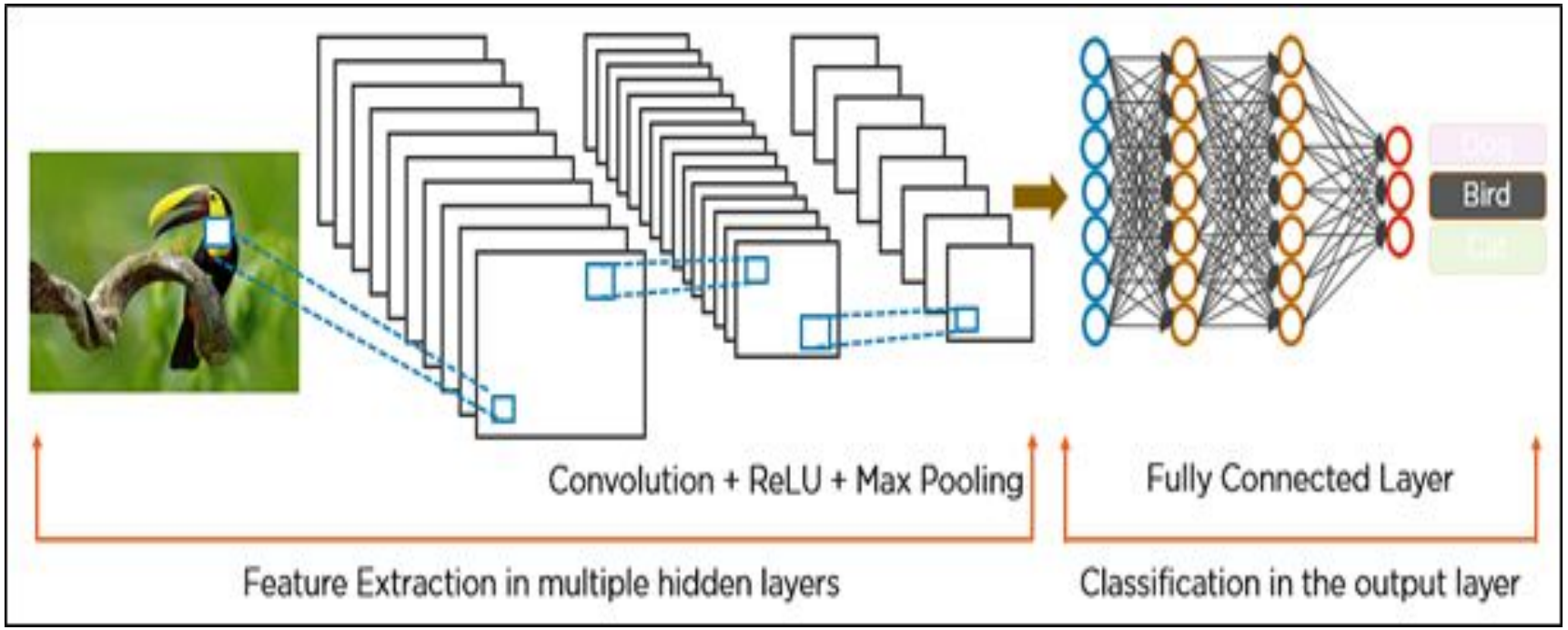
The next step in the process is called flattening. Flattening is used to convert all the resultant 2-Dimensional arrays from pooled feature maps into a single long continuous linear vector.



The flattened matrix is fed as input to the fully connected layer to classify the image.



Here’s how exactly CNN recognizes a bird:



The Transformer Model

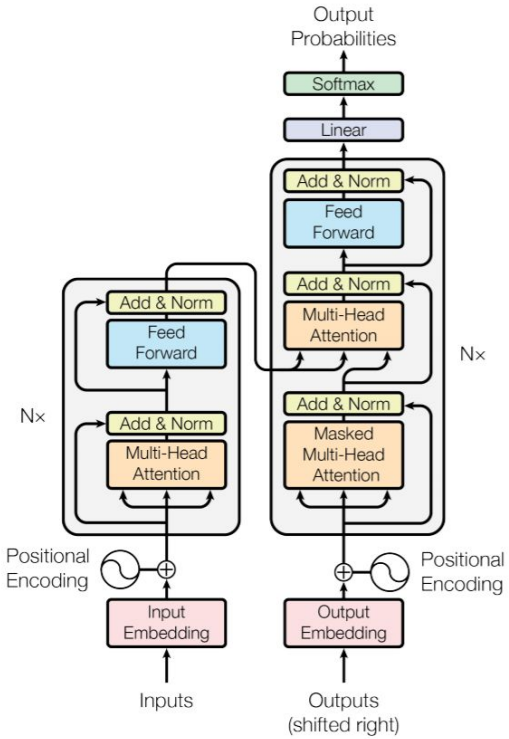
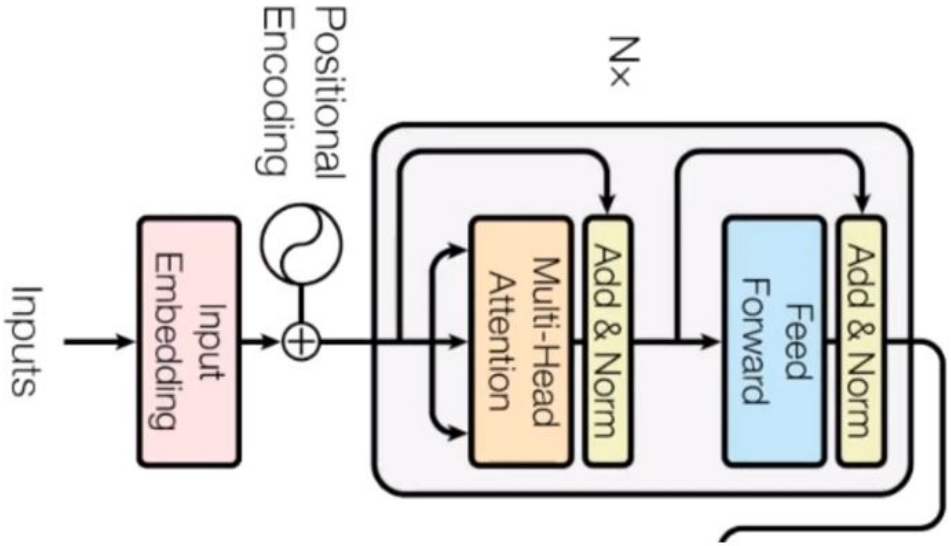


Figure 1: The Transformer - model architecture.

The Encoder Block

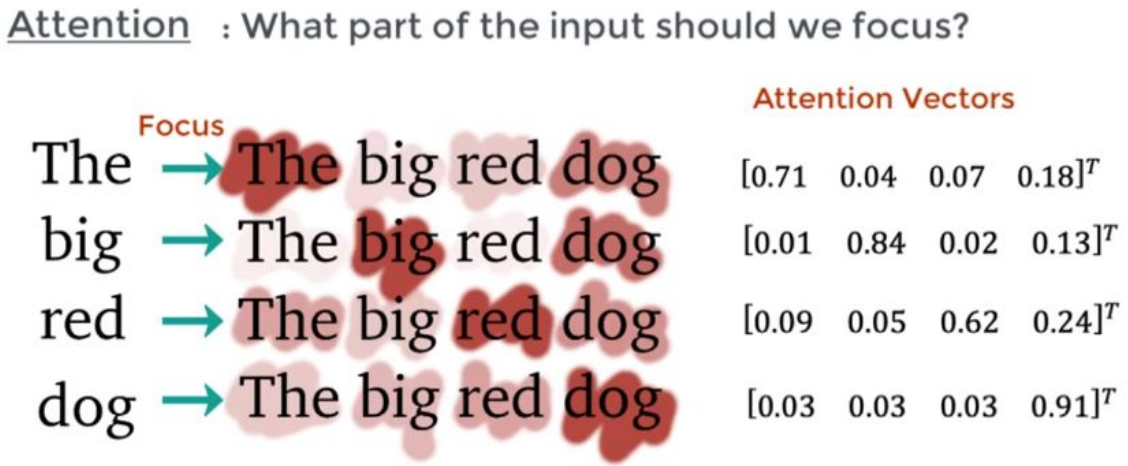


It’s a fact that computers don’t understand words, it works on numbers, vectors or matrices. So, we need to convert our words to a vector. So, here the concept of Embedding Space comes. It’s like an open space or dictionary where words of similar meanings are grouped together or are present close to each other in that space. This space is termed as embedding space, and here every word, according to its meaning, is mapped and assigned with a particular value. So, here we convert our words to vectors.

But one other issue we will face is that every word in different sentences has different meanings. So, to solve this issue, we take the help of Positional Encoders. It is a vector that gives context according to the position of the word in a sentence.

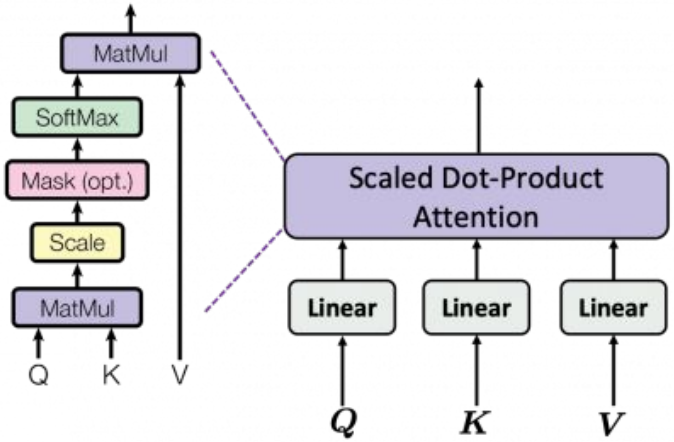
Now the main essence of the transformer comes in, “Self Attention”.

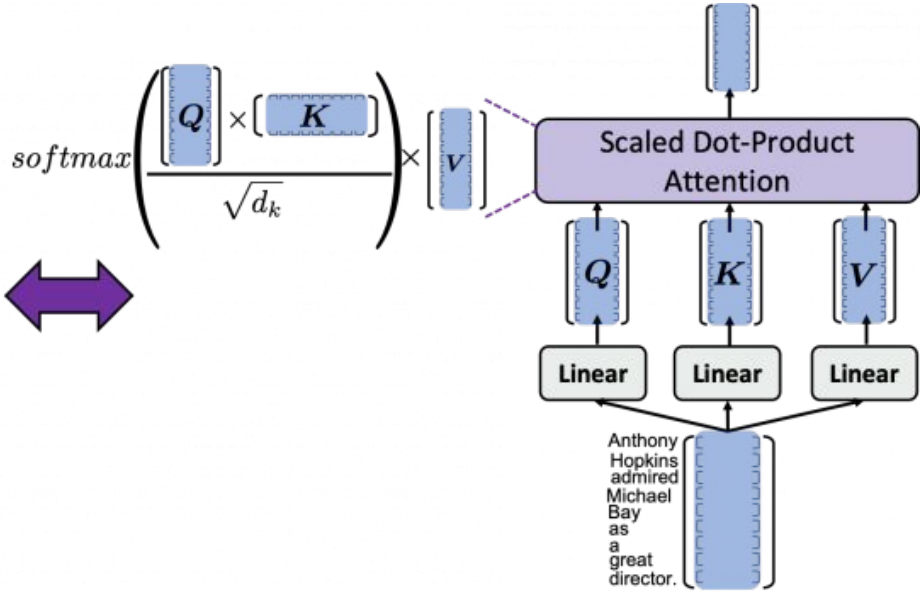
It focuses on how relevant a particular word is with respect to other words in that sentence. It is represented as an attention vector. For every word, we can have an attention vector generated, which captures the contextual relationship between words in that sentence



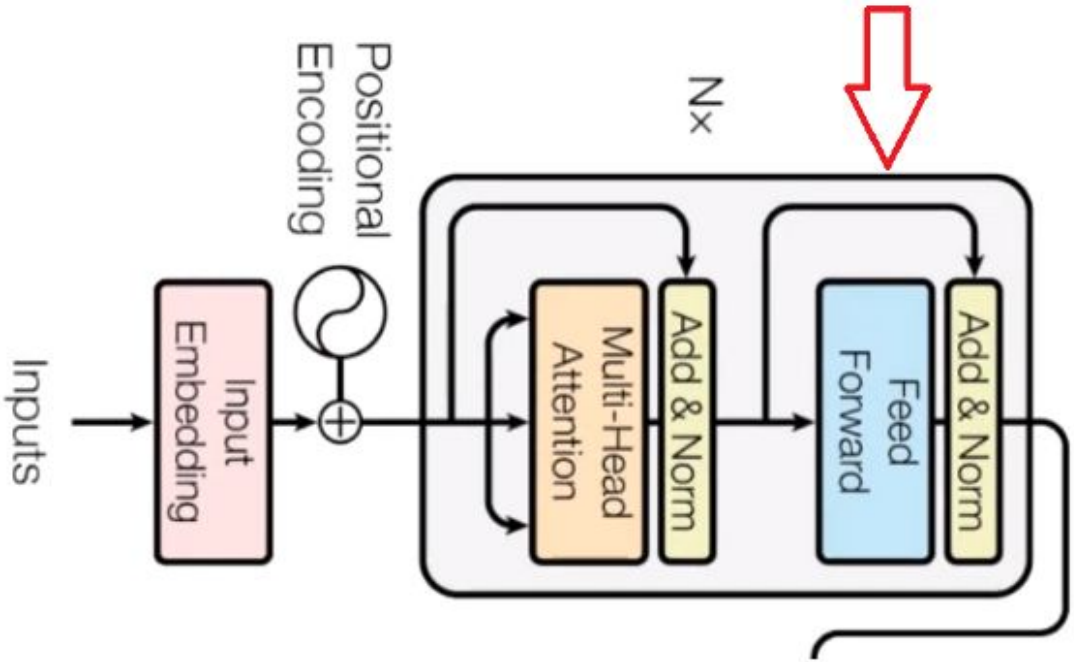
The only problem it faces is that for every word it weighs its value much higher on itself in the sentence, be we are inclined towards its interaction with other words of that sentence. So, we determine multiple attention vectors per word and take a weighted average, to compute the final attention vector of every word.

How Attention is calculated

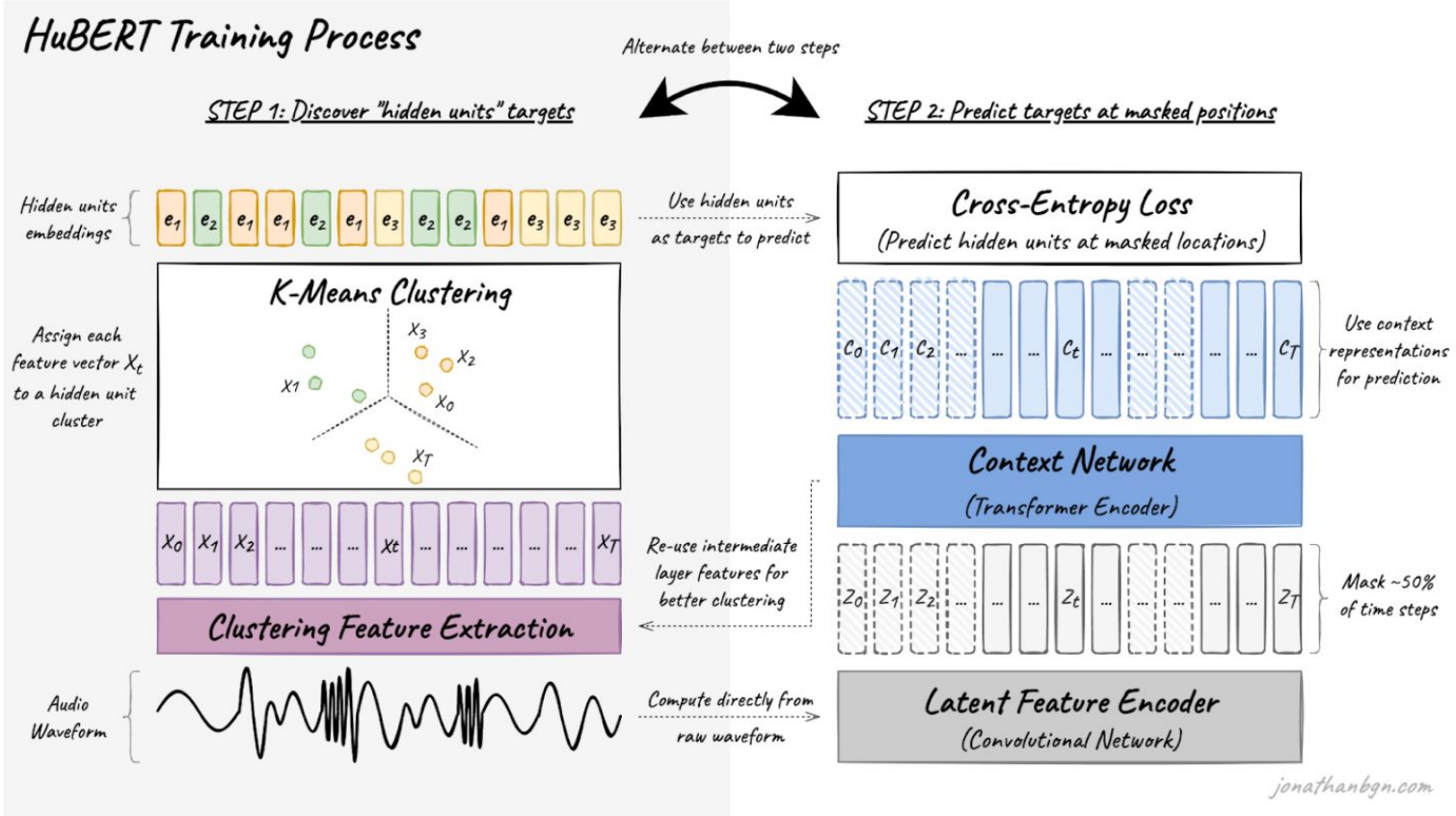




Feed Forward layer



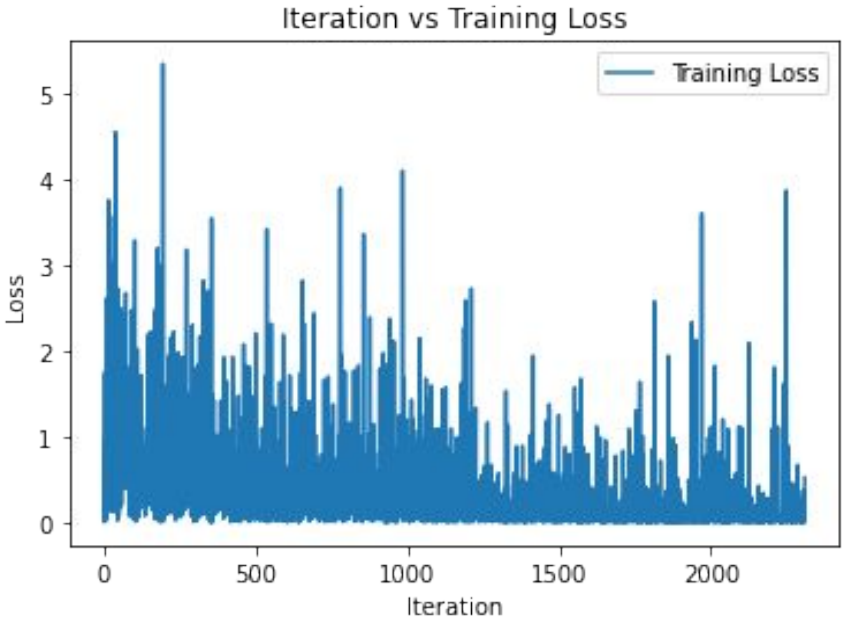
The HuBERT Model

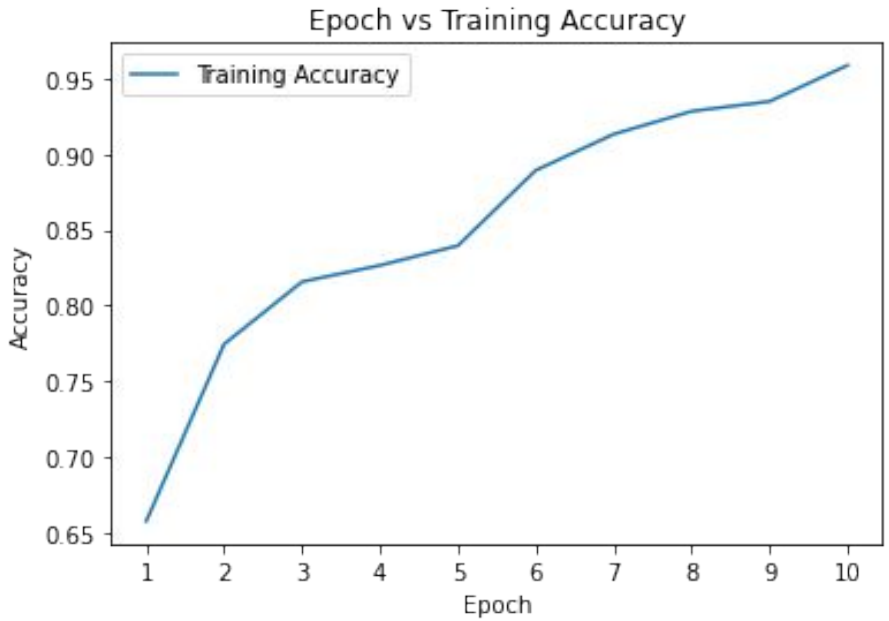


Libraries

- Librosa
- SoundFile
- Pandas
- NumPy
- Glob
- train_test_split
- Wav2VecFeatureExtractor
- MLPClassifier
- Matplotlib
- Seaborn
- HubertForSequence Classification

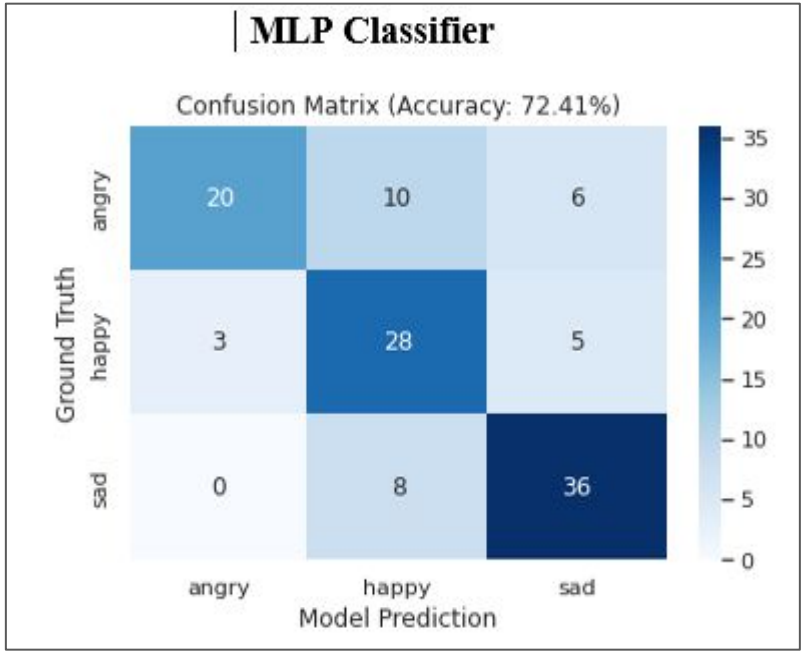
AdamW Optimizer

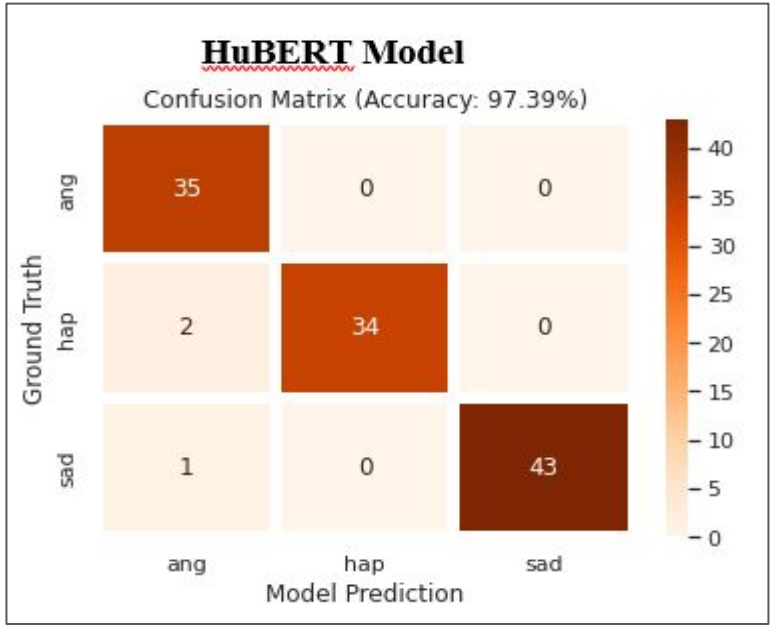




Comparison (MLP Classifier and HuBERT Model)

Dataset: **The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)**





Classification Report: Dataset (RAVDESS)

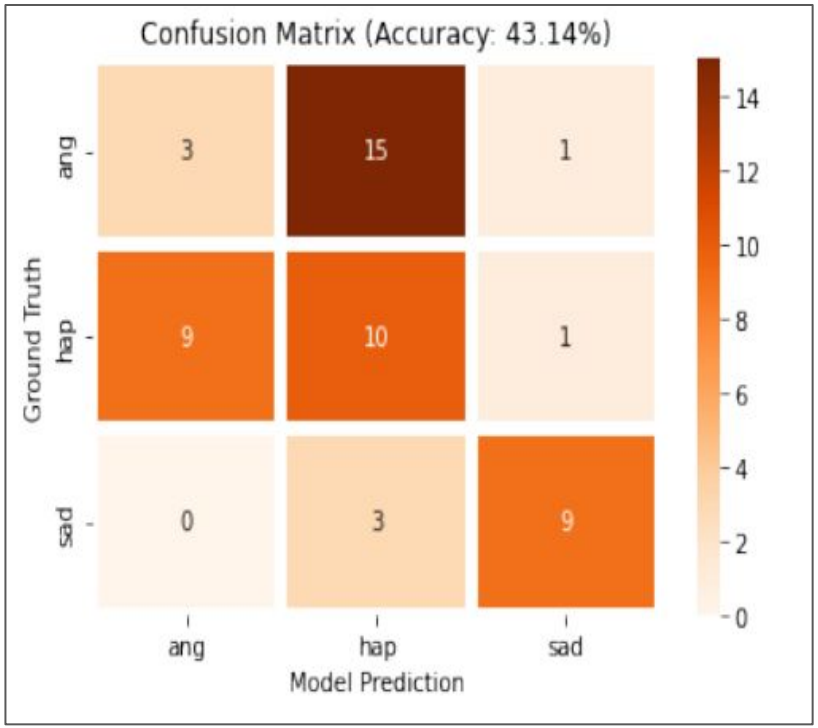
MLPClassifier

	precision	recall	f1-score	support
angry	0.87	0.56	0.68	36
happy	0.61	0.78	0.68	36
sad	0.77	0.82	0.79	44
accuracy			0.72	116
macro avg	0.75	0.72	0.72	116
weighted avg	0.75	0.72	0.72	116

HuBERT Model

	precision	recall	f1-score	support
ang	0.92	1.00	0.96	35
hap	1.00	0.94	0.97	36
sad	1.00	0.98	0.99	44
accuracy			0.97	115
macro avg	0.97	0.97	0.97	115
weighted avg	0.98	0.97	0.97	115

Our Dataset Confusion Matrix & Classification Report



Classification Report

	precision	recall	f1-score	support
ang	0.25	0.16	0.19	19
hap	0.36	0.50	0.42	20
sad	0.82	0.75	0.78	12
accuracy			0.43	51
macro avg	0.48	0.47	0.46	51
weighted avg	0.43	0.43	0.42	51

Challenges

- Understanding the feature extraction techniques
- Understanding how MFCC works. It is important to know how to use it and what values it requires to get the desired features.
- Selecting the correct ML model. There are a lot of models available. Choosing the correct model and more importantly, one that’s not outdated was a difficult task.
- Understanding the various ML models available.
- Creating our own dataset Finding audio samples to make our own dataset.
- Processing the audio was also a new challenge as we had to learn to remove noise, isolate music from speech, and other audio processing, etc.
- Learning about the libraries. Each of the library that is used has various functions / modules. Learning each of those modules and the various parameters they expect as well as what type of data it returns.