GESTURE TO SPEECH USING ASL DATASET

PROJECT REPORT

TABLE OF CONTENTS

ABSTRACT

Speech disorders can affect the way a person creates sounds to form words. Certain voice disorders may also be considered speech disorders. The intention behind proposing this project is to help the speech impaired people to communicate their needs without the requirement of a translator. Usually when the speech disabled people try to communicate the requirements, the listener is also intended to have the knowledge of sign language. This project proposes to rule out that need with the help of emerging technologies. The idea of the project is to use Computer vision and Neural Networks to generate a CNN model on an image dataset to convert the gestures into words. This model can be chosen with the help of exploratory data analytics on the existing images. This is paired with a Mobile friendly Android Mobile Application that helps the end user interface with the Deep Learning model.

INTRODUCTION

The Indian Speech, Language and Hearing Association (ISHA) has launched a campaign during December, 2020 to highlight issues related to communication disorder. There are nearly 100 million people in India with communication impairment but public awareness about communication disorders and the role of audiologists and Speech Language Pathologists in alleviating them is very low, the release added. Sign languages use visual-manual modality to convey meaning. American Sign Language (ASL) is a natural language that serves as the predominant sign language of Deaf communities in the United States and most of Anglophone Canada. ASL is a complete and organized visual language that is expressed by facial expression as well as movements and motions with the

hands.ASL originated in the early 19th century in the American School for the Deaf (ASD) in West Hartford, Connecticut, from a situation of language contact. Since then, ASL use has been propagated widely by schools for the speech and hearing impaired community organizations.

Though ASL might seem straightforward to the specially impaired, the population unaware of the signs still remain significant. To bridge the communication gap between the specially impaired and common people and to instill confidence in the speech impaired people we propose this novel Smartphone Based Gesture to Speech converter. This project doesn't require any high end devices rather a humble smartphone which is used by most of the commoners is sufficient for the end user to utilize this facility.

LITERATURE SURVEY

There are many similar works presented by many authors. Few instances are:

S. Patel, U. Dhar, S. Gangwani, R. Lad, and P. Ahire proposed a system that uses an intrinsic mobile camera for gesture recognition and acquisition; gesture acquired is processed with the help of Algorithms like HSV model-(Skin Color Detection), LargeBlob Detection, Flood Fill and Contour Extraction. The system can recognize one-handed sign representation of the standard alphabets (A-Z) and numeric values (0-9) [1]. R. R. Itkarkar and A. V. Nandi proposed a system consisting of a camera attached to a computer that will take images of hand gestures. Image segmentation & feature extraction algorithm is used to recognize the hand gestures of the signer. According to recognized hand gestures, corresponding pre-recorded soundtracks will be played [2].S. Vigneshwaran, M. Shifa Fathima, V. Vijay Sagar, and R. Sree Arshika proposed a project non-vision-based technique will be used. Most of the dumb people are deaf also. So the normal people's voice can be converted into their sign language. In an emergency, messages will automatically be sent to their relation or friends[3]. P. Vijayalakshmi and M. Aarthi proposed a flex sensor-based gesture recognition module to recognize English alphabets and few words and a Text-to-Speech synthesizer based on HMM is built to convert the corresponding text [4]. L. Anusha and Y. U. Devi proposed a system consisting of MPU6050 for sensing gesture movement, Raspberry pi for processing, a three-button Keypad, and a speaker. It is implemented by using a trajectory recognition algorithm for recognizing alphabets. Raspberry pi generates voice output for the text in multiple languages using voice RSS and Microsoft translator [5]. H. S. Kala, S. R. S., S. Pal, U. S. K., and S. Chakma worked on a project that mainly focuses on removing the barrier of communication between the mute community and the people not familiar with the concept of sign language so that the messages that a dumb person is trying to relay is understandable to a person with no knowledge of sign language. The design of the device is based on embedded systems. Flex sensors and microcontrollers are the key components [6]. N. Harish and S. Poonguzhali have proposed a project using flex sensors and an accelerometer. The movements included during gesture representation are rotation, angle tilt, and direction changes. The flex sensor and the accelerometer are incorporated over fingers and wrist respectively to acquire their dynamics, these

sensors are fitted over the data glove. These voltage signals will then be processed by the microcontroller and sent to the voice module, where the words voice outputs are stored and playbacked equivalent to each word values to produce the appropriate voice words with the help of the speaker [7].

METHODOLOGY

DATASET GENERATION

Kaggle is a machine Learning repository. The repository has an ASL image dataset consisting of 3000 images per an alphabet for training the model. We chose only 400 images per alphabet and additionally added 100 images of our own self generated images to customize the training to suit different background scenes and different lighting effects. That will sum up to 14000 images for training in total. This is the suitable number of images for CNN model, without overfitting the model.

Fig. Sample of the dataset generated

A python script that utilizes the laptop's front camera is employed to take the snap shots of the gestures. These are stored in separate folders according to their respective classes. We have chosen 28 classes in total. All the 26 alphabets along with "space" and "nothing" classes to define a break between the words and detect if no gesture is made is trained to predict the exact sentence.

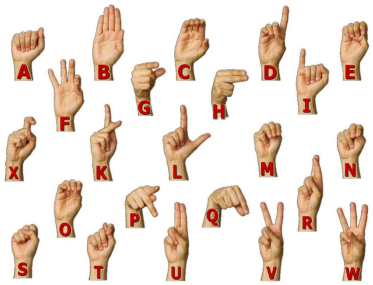
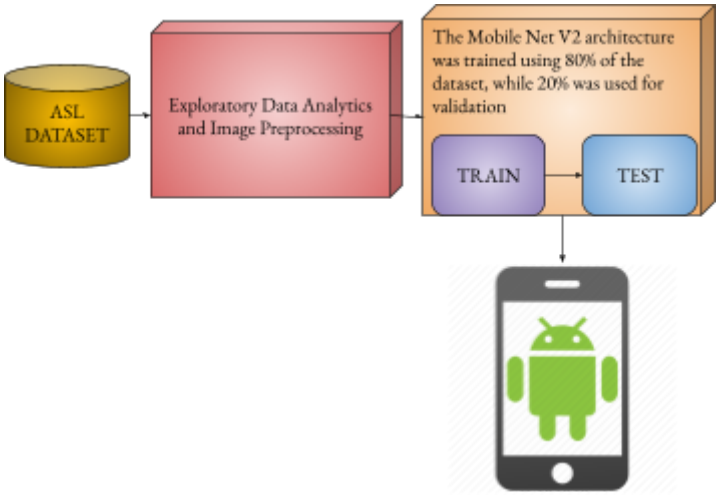


Fig. Symbols in ASL dataset

BLOCK DIAGRAM



The working of this prototype is straightforward. The dataset was explored using various techniques that are further discussed in the document. This EDA extracted useful features from the images and gave a skeleton on the images to be dealt with. This image is passed through various Deep Learning Models to find the best model. Mobile Net V2 was the preferred architecture since it has a lightweight architecture suitable for deployment into a Mobile Phone. The model performed with

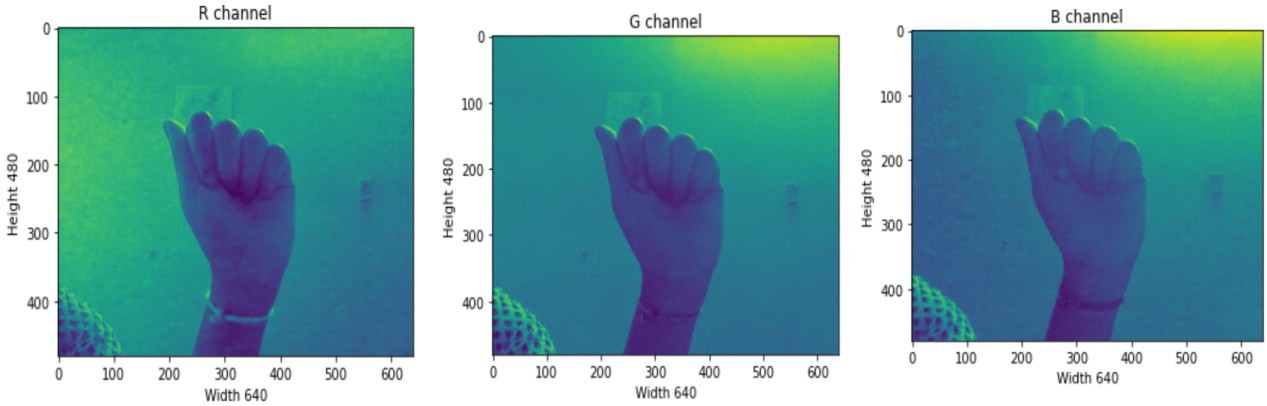
This is deployed into a Mobile Application. The Android application was developed using Java in Android Studio Platform.

EXPLORATORY DATA ANALYTICS

IMAGE

Every image is made up of a certain number of pixels. The word pixel means a picture element. A simple way to describe each pixel is using a combination of three colors, namely Red, Green, Blue. This is what we call an RGB image.Every photograph, in digital form, is made up of pixels. They are the smallest unit of information that makes up a picture. Usually round or square, they are typically arranged in a 2-dimensional grid. If all three values of Red,Green and Blue are at full intensity, that means they're 255. It then shows as white, and if all three colors are muted, or has the value of 0, the color shows as black. The combination of these three will, in turn, give us a specific shade of the pixel color. Since each number is an 8-bit number, the values range from 0–255.The combination of these three colors tends to the highest value among them. Since each value can have 256 different intensity or brightness values, it makes 16.8 million total shades.Hence, in the case of a colored image, there are three Matrices (or channels) - Red, Green, and Blue. Each matrix has values between 0-255 representing the intensity of the color for that pixel.

0 index value for Red channel, 1 index value for Green channel and 2 index value for Blue channel



DISTRIBUTION PLOT FOR RGB PIXELS

A distribution plot displays a distribution and range of a set of numeric values plotted against a dimension.Data is plotted as value points along an axis.Here Distribution plot is plotted for the RGB matrices to visualize different intensity values of the channels at a particular pixel.

```
<AxesSubplot:ylabel='Density'>
```

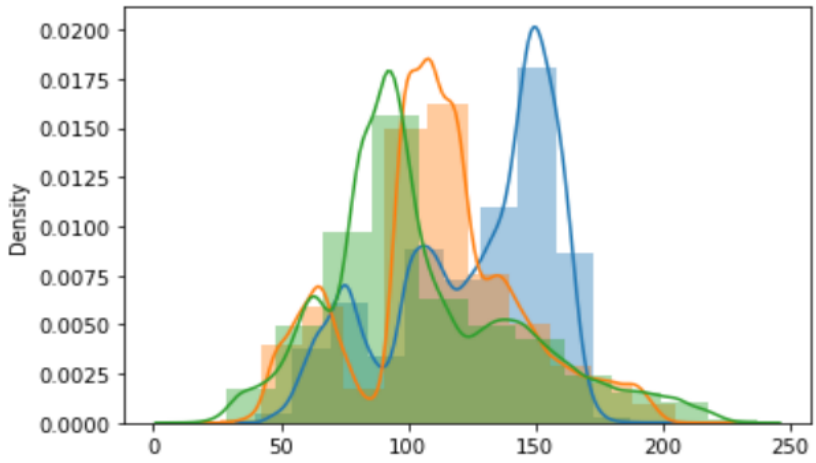


IMAGE SPLITTING

Image. split() method is used to split the image into individual bands. This method returns a tuple of individual image bands from an image. Splitting an "RGB" image creates three new images each containing a copy of one of the original bands (red, green, blue).

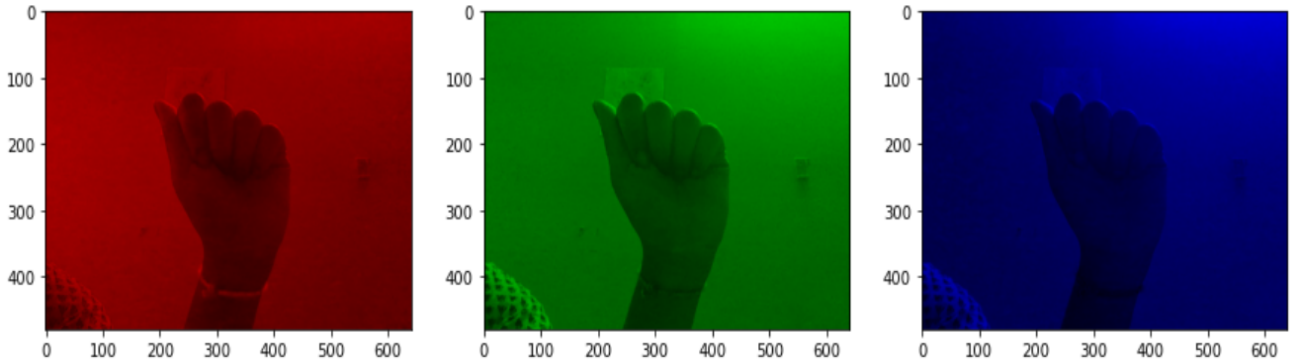
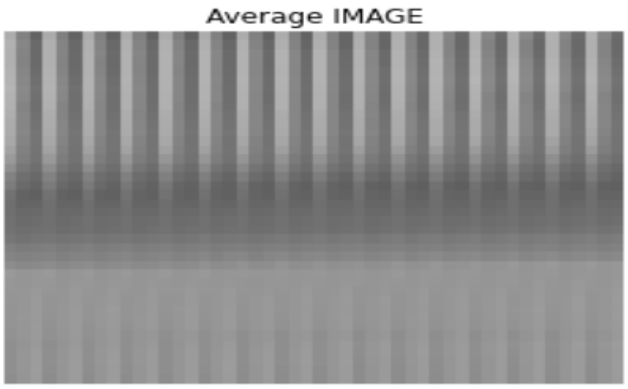


IMAGE AVERAGING

Image averaging is a digital image processing technique that is often employed to enhance images. The algorithm operates by computing an average or arithmetic mean of the intensity values for each pixel position in a set of captured images from the same scene or viewfield.

The intensity values **A(N, x, y)** of pixels located at coordinates **(x, y)** in the average image will be expressed by the equation**:**

$$A(N, x, y) = \frac{1}{N} \cdot \sum_{i=1}^{N} I(i, x, y)$$



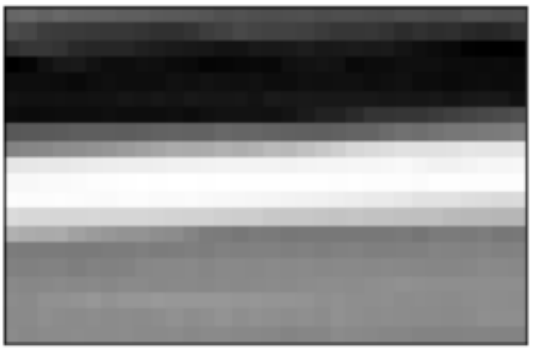
Here the average image is formed where it moves to every pixel and replaces the value with the average value of the neighbouring pixel including itself.

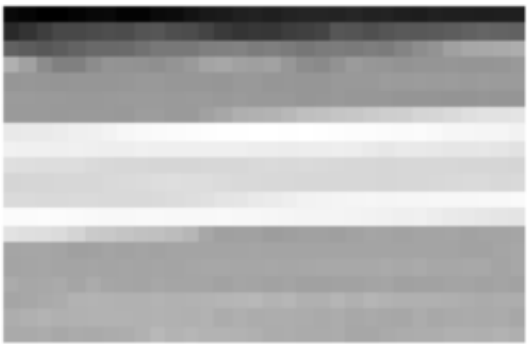
EIGEN IMAGE

An eigen image is formed by the eigenvectors that are derived from the covariance matrix of the probability distribution over the high-dimensional vector space of face images. The eigenimages themselves form a basis set of all images used to construct the covariance matrix. This produces dimension reduction by allowing the smaller set of basis images to represent the original training

images. Classification can be achieved by comparing how images are represented by the basis set.A set of eigenimages can be generated by performing a mathematical process called principal component analysis (PCA).The eigenimages that are created will appear as light and dark areas that are arranged in a specific pattern. This pattern is how different features of a gesture are singled out to be evaluated and scored.

```
Number of PC:  2
```





Here, eigen image is is formed by initially subtracting the average image from the normal image, then each row of pixels are taken and formed into a vector, So as a result a matrix can be formed with the extraction of the vectors, then the covariance matrix is formed with the higher probability distribution vector and finally that covariance matrix of pixels are converted into an image.
This produces dimension reduction by allowing the smaller set of basis images to represent the original training images. Once the eigenimages of a database are calculated, image recognition can be achieved in real time. Eigenimages can handle large database and also reduce the statistical complexity of image recognition.

EDGE IMAGE

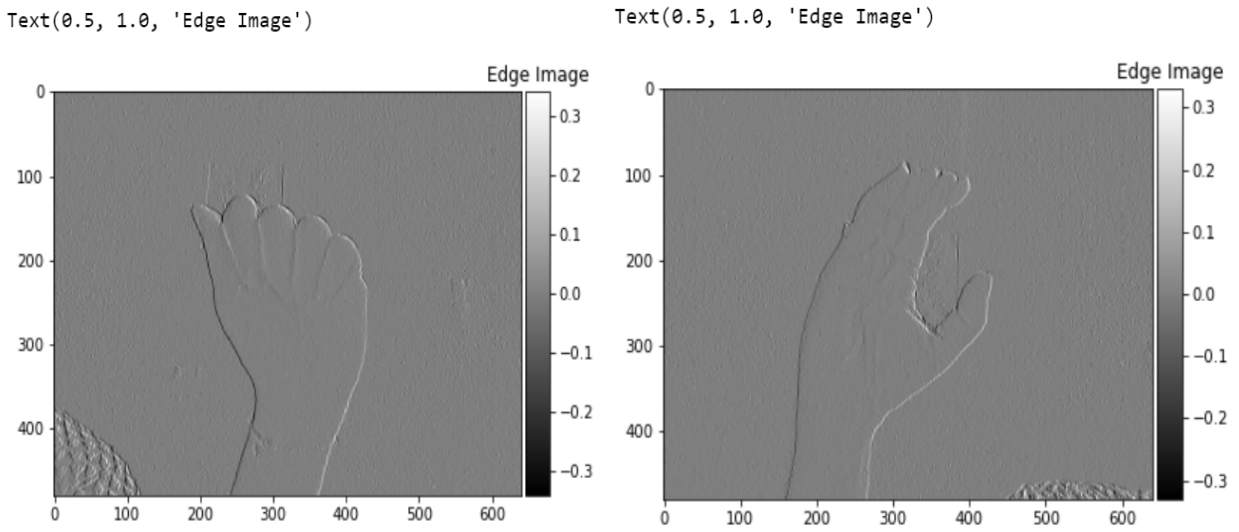
Edge image is formed by identifying points in a digital image at which the image brightness changes sharply or has discontinuities. Here we have used the Prewitt operator for edge detection.
The Prewitt operator is used in image processing, particularly within edge detection algorithms. It is a discrete differentiation operator, computing an approximation of the gradient of the image intensity function. At each point in the image, the result of the Prewitt operator is either the corresponding gradient vector or the norm of this vector. The Prewitt operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical directions.

Prewitt operator used for edge detection in an image, detects two types of edges

1. Horizontal edges
2. Vertical Edges

It uses two 3×3 kernels which are convolved with the original image to calculate approximations of the derivatives - one for horizontal changes, and one for vertical.

The operator calculates the gradient of the image intensity at each point, giving the direction of the largest possible increase from light to dark and the rate of change in that direction. The gradient of a two-variable function is at each image point a 2D vector with the components given by the derivatives in the horizontal and vertical directions. At each image point, the gradient vector points in the direction of largest possible intensity increase, and the length of the gradient vector corresponds to the rate of change in that direction. This implies that the result of the Prewitt operator at an image point which is in a region of constant image intensity is a zero vector and at a point on an edge is a vector which points across the edge, from darker to brighter values.



DEEP LEARNING MODEL

BACKGROUND WORK

Artificial Intelligence has been witnessing a monumental growth in bridging the gap between the capabilities of humans and machines. Researchers and enthusiasts alike, work on numerous aspects of the field to make amazing things happen. One of many such areas is the domain of Computer Vision.The advancements in Computer Vision with Deep Learning has been constructed and perfected with time, primarily over one particular algorithm — a Convolutional Neural Network.

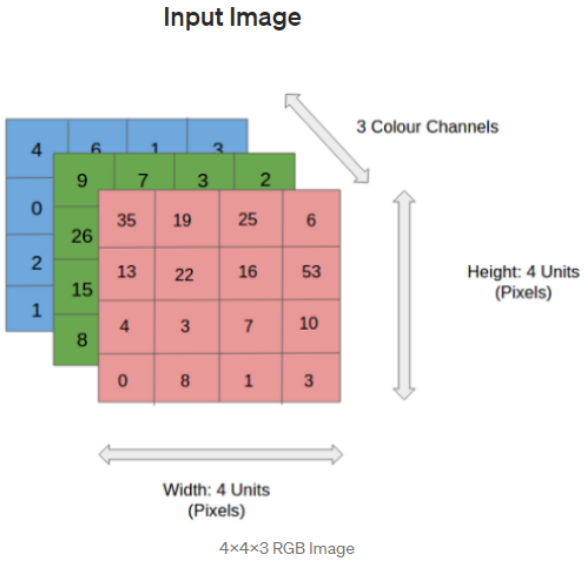
Convolutional Neural Network

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. The pre-processing required in a ConvNet is much

lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

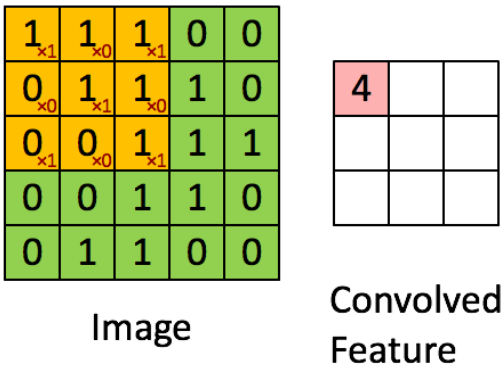
The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the. Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

In the figure, we have an RGB image which has been separated by its three color planes — Red, Green, and Blue. There are a number of such color spaces in which images exist — Grayscale, RGB, HSV, CMYK, etc.



The role of the ConvNet is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction. This is important when we are to design an architecture which is not only good at learning features but also is scalable to massive datasets.

The kernel



ting a 5x5x1 image wit

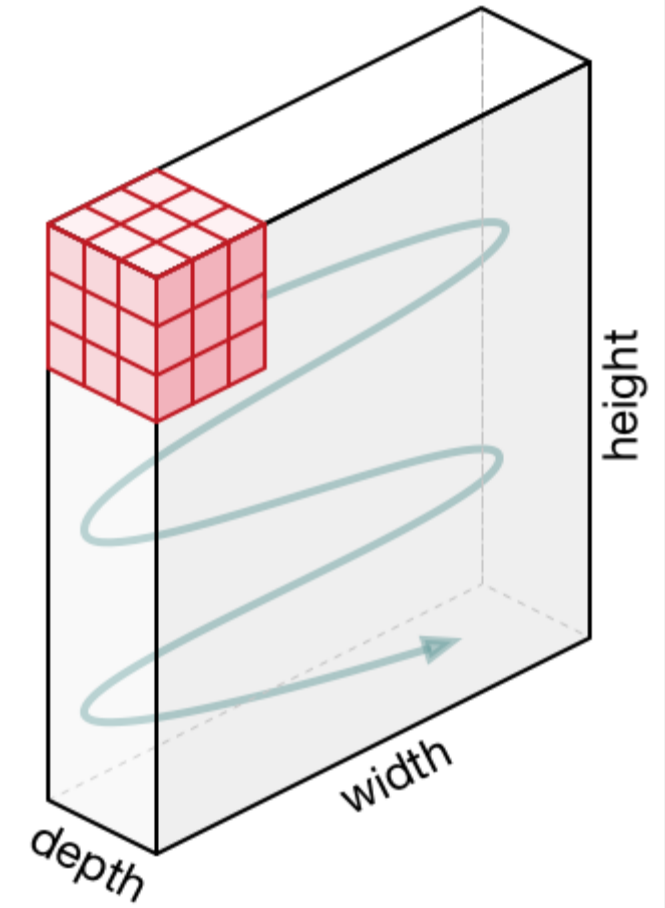
1 kernel to get a 3x3x1 convolved feature

Image Dimensions = 5 (Height) x 5 (Breadth) x 1 (Number of channels, eg. RGB)

In the above demonstration, the green section resembles our 5x5x1 input image, I. The element involved in carrying out the convolution operation in the first part of a Convolutional Layer is called the Kernel/Filter, K, represented in the color yellow. We have selected K as a 3x3x1 matrix.

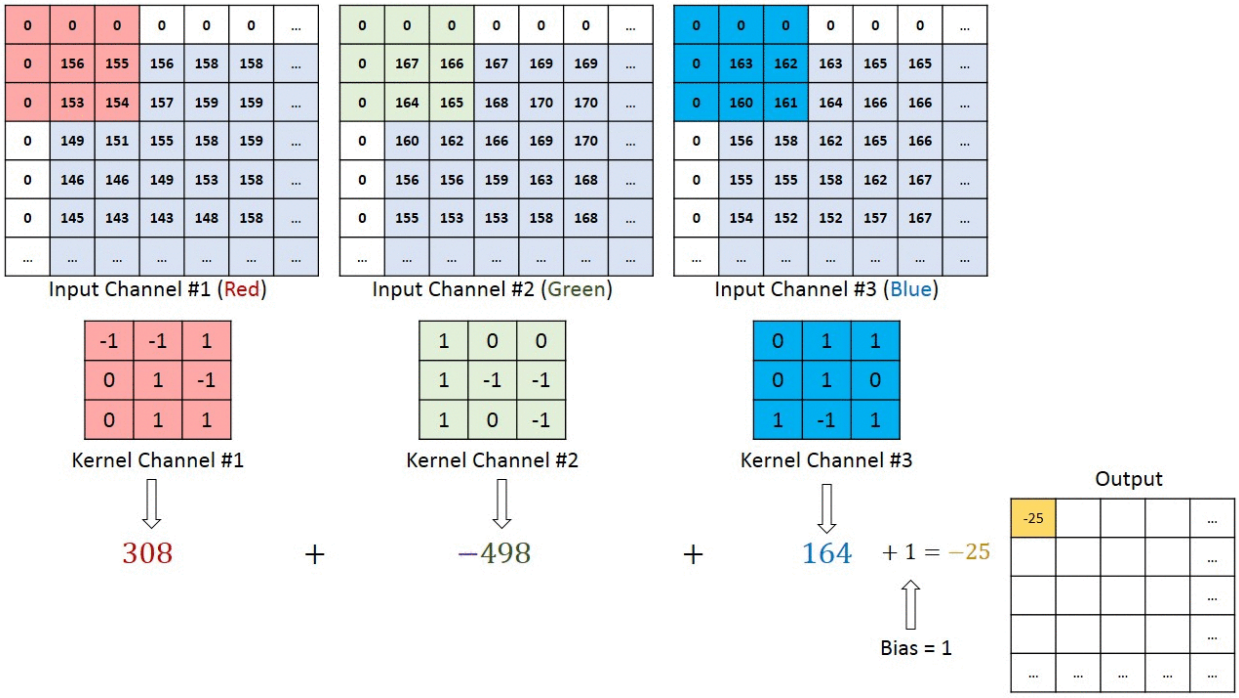
Kernel/Filter, K =1 0 1 0 1 0 1 0 1

The Kernel shifts 9 times because of Stride Length = 1 (Non-Strided), every time performing a matrix multiplication operation between K and the portion P of the image over which the kernel is hovering.



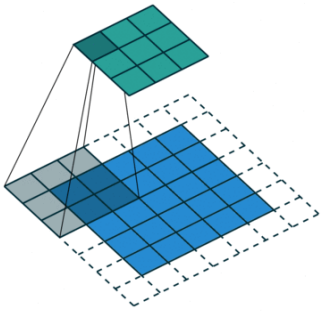
Movement of the Kernel

The filter moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.



Convolution operation on a MxNx3 image matrix with a 3x3x3 Kernel

In the case of images with multiple channels (e.g. RGB), the Kernel has the same depth as that of the input image. Matrix Multiplication is performed between Kn and In stack ([K1, I1]; [K2, I2]; [K3, I3]) and all the results are summed with the bias to give us a squashed one-depth channel Convoluted Feature Output.



Convolution Operation with Stride Length = 2

The objective of the Convolution Operation is to extract the high-level features such as edges, from the input image. ConvNets need not be limited to only one Convolutional Layer. Conventionally, the first ConvLayer is responsible for capturing the Low-Level features such as edges, color, gradient orientation, etc. With added layers, the architecture adapts to the High-Level features as well, giving us a network which has the wholesome understanding of images in the dataset, similar to how we would.

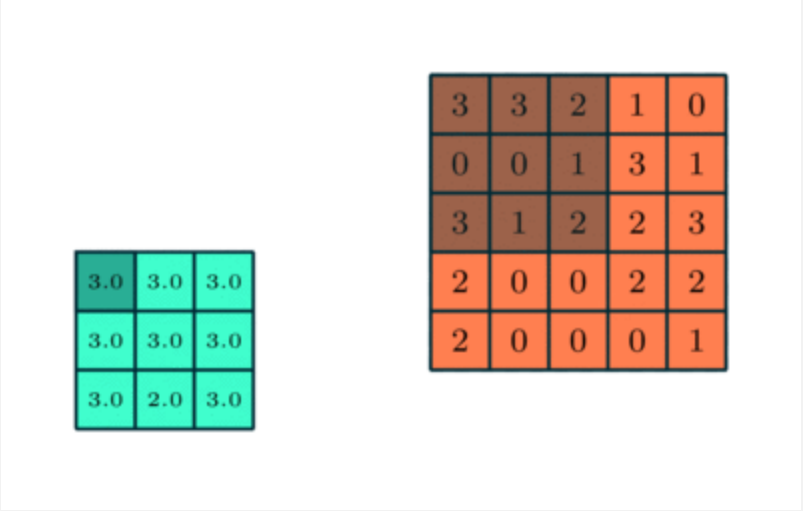
There are two types of results to the operation — one in which the convolved feature is reduced in dimensionality as compared to the input, and the other in which the dimensionality is either increased or remains the same. This is done by applying Valid Padding in case of the former, or Same Padding in the case of the latter.

SAME padding: 5x5x1 image is padded with 0s to create a 6x6x1 image

When we augment the 5x5x1 image into a 6x6x1 image and then apply the 3x3x1 kernel over it, we find that the convolved matrix turns out to be of dimensions 5x5x1. Hence the name — Same Padding.

On the other hand, if we perform the same operation without padding, we are presented with a matrix which has dimensions of the Kernel (3x3x1) itself — Valid Padding.

Pooling Layer



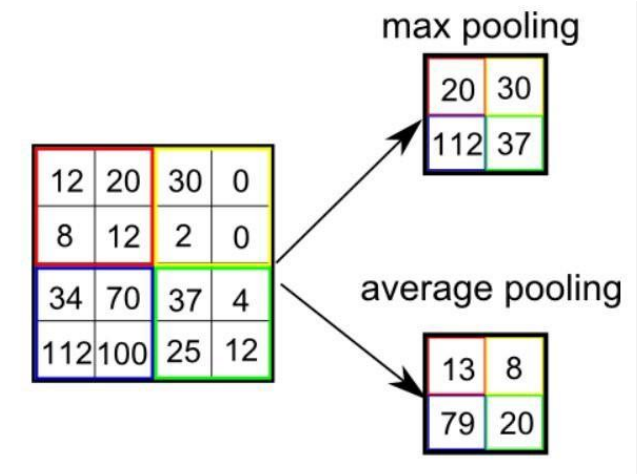
3x3 pooling over 5x5 convolved feature

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model.

There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel.

Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction. On the other hand, Average Pooling simply

performs dimensionality reduction as a noise suppressing mechanism. Hence, we can say that Max Pooling performs a lot better than Average Pooling.

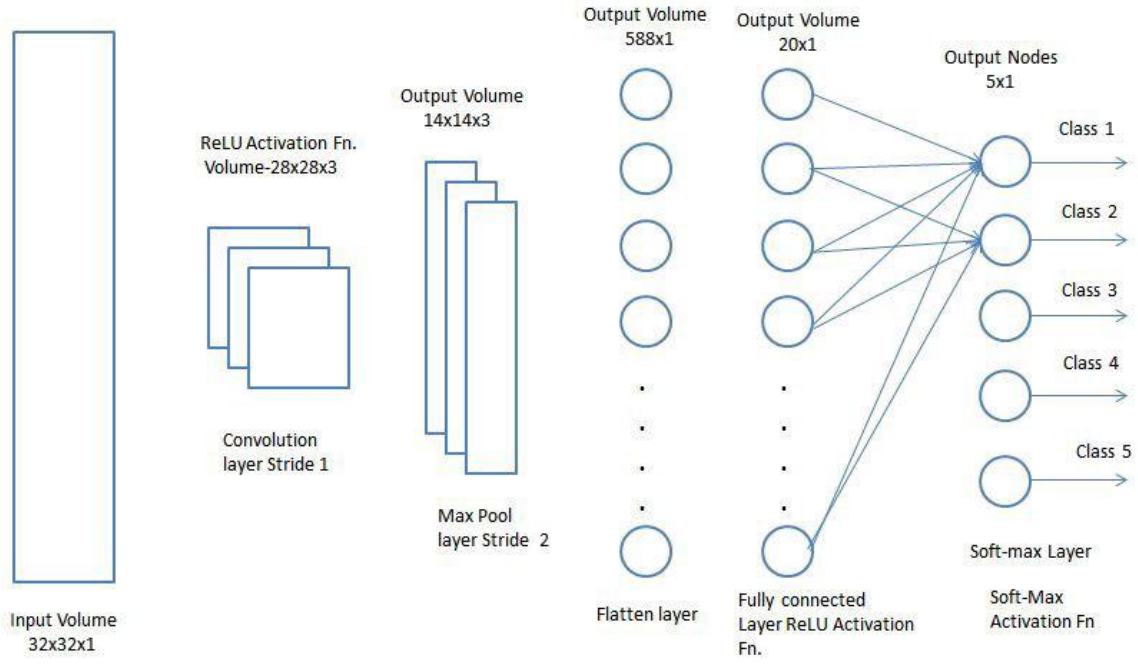


Types of Pooling

The Convolutional Layer and the Pooling Layer, together form the i-th layer of a Convolutional Neural Network. Depending on the complexities in the images, the number of such layers may be increased for capturing low-levels details even further, but at the cost of more computational power.

After going through the above process, we have successfully enabled the model to understand the features. Moving on, we are going to flatten the final output and feed it to a regular Neural Network for classification purposes.

Classification — Fully Connected Layer (FC Layer)\



Adding a Fully-Connected layer is a (usually) cheap way of learning non-linear combinations of the high-level features as represented by the output of the convolutional layer. The Fully-Connected layer is learning a possibly non-linear function in that space.

Now that we have converted our input image into a suitable form for our Multi-Level Perceptron, we shall flatten the image into a column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training. Over a series of epochs, the model is able to distinguish between dominating and certain low-level features in images and classify them using the Softmax Classification technique.

There are various architectures of CNNs available which have been key in building algorithms which power and shall power AI as a whole in the foreseeable future. Four of them were chosen for comparison in this project

MobileNetV2

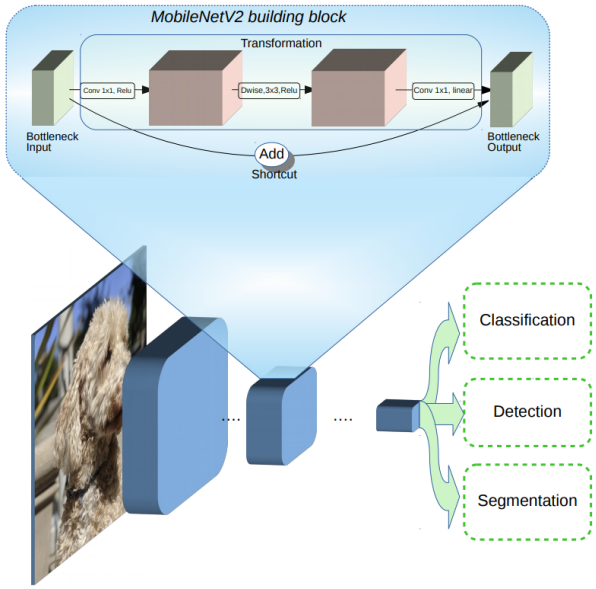
In MobileNetV2, there are two types of blocks. One is a residual block with stride of 1. Another one is block with stride of 2 for downsizing.There are 3 layers for both types of blocks. This time, the first layer is 1×1 convolution with ReLU6. The second layer is the depthwise convolution. The third layer is another 1×1 convolution but without any non-linearity. It is claimed that if ReLU is used again, the deep networks only have the power of a linear classifier on the non-zero volume part of the output domain.

Input	Operator	Output
$h \times w \times k$	1x1 conv2d , ReLU6	$h \times w \times (tk)$
$h \times w \times tk$	3x3 dwise s=s, ReLU6	$\frac{h}{s} \times \frac{w}{s} \times (tk)$
$\frac{h}{s} \times \frac{w}{s} \times tk$	linear 1x1 conv2d	$\frac{h}{s} \times \frac{w}{s} \times k'$

And there is an expansion factor t. And t=6 for all main experiments. If the input got 64 channels, the internal output would get 64×t=64×6=384 channels.2. Overall Architecture

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

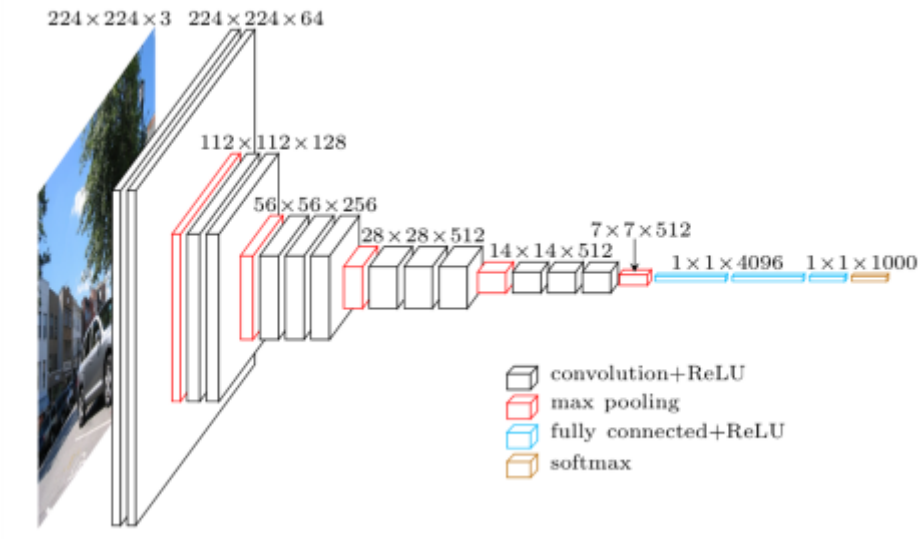
where *t*: expansion factor, *c*: number of output channels, *n*: repeating number, s: stride. 3×3 kernels are used for spatial convolution.In typical, the primary network (width multiplier 1, 224×224), has a computational cost of 300 million multiply-adds and uses 3.4 million parameters. The performance trade offs are further explored, for input resolutions from 96 to 224, and width multipliers of 0.35 to 1.4. The network computational cost up to 585M MAdds, while the model size varies between 1.7M and 6.9M parameters. To train the network, 16 GPUs are used with a batch size of 96.



VGG 16

VGG16 is a convolution neural net (CNN) architecture which was used to win ILSVR(Imagenet) competition in 2014. It is considered to be one of the excellent vision model architecture till date. Most unique thing about VGG16 is that instead of having a large number of hyper-parameter they focused on having convolution layers of 3x3 filter with a stride 1 and always used same padding and maxpool layer of 2x2 filter of stride 2. It follows this arrangement of convolution and max pool layers consistently throughout the whole architecture. In the end it has 2 FC(fully connected layers) followed by a softmax for output. The 16 in VGG16 refers to it has 16 layers that have weights. This network is a

pretty large network and it has about 138 million (approx) parameters.



Inception V3

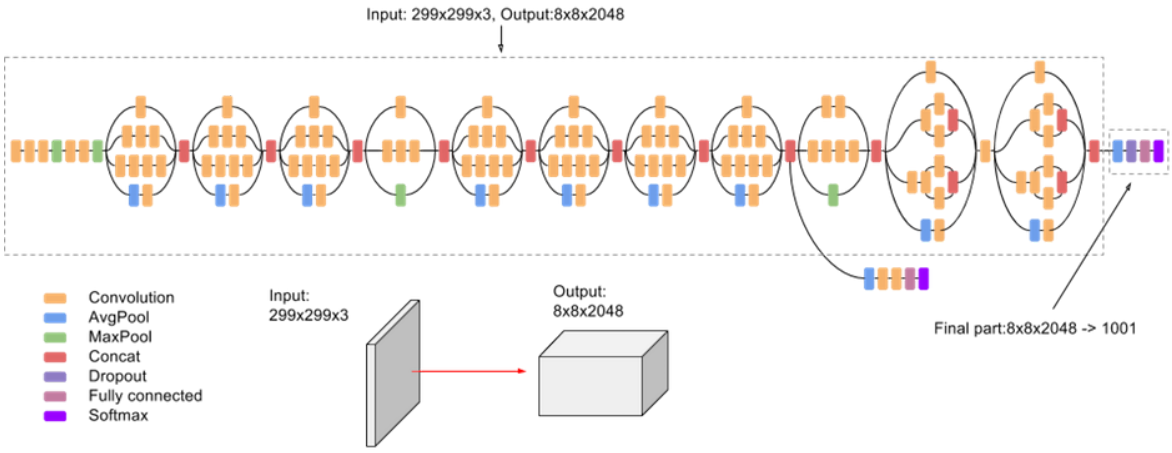
The authors noted that the auxiliary classifiers didn't contribute much until near the end of the training process, when accuracies were nearing saturation. They argued that they function as regularizers, especially if they have BatchNorm or Dropout operations.Possibilities to improve on the Inception v2 without drastically changing the modules were to be investigated.

The Solution

Inception Net v3 incorporated all of the above upgrades stated for Inception v2, and in addition used the following:

- RMSProp Optimizer.
- Factorized 7x7 convolutions.
- BatchNorm in the Auxiliary Classifiers.

- Label Smoothing (A type of regularizing component added to the loss formula that prevents the network from becoming too confident about a class. Prevents overfitting).



Xception

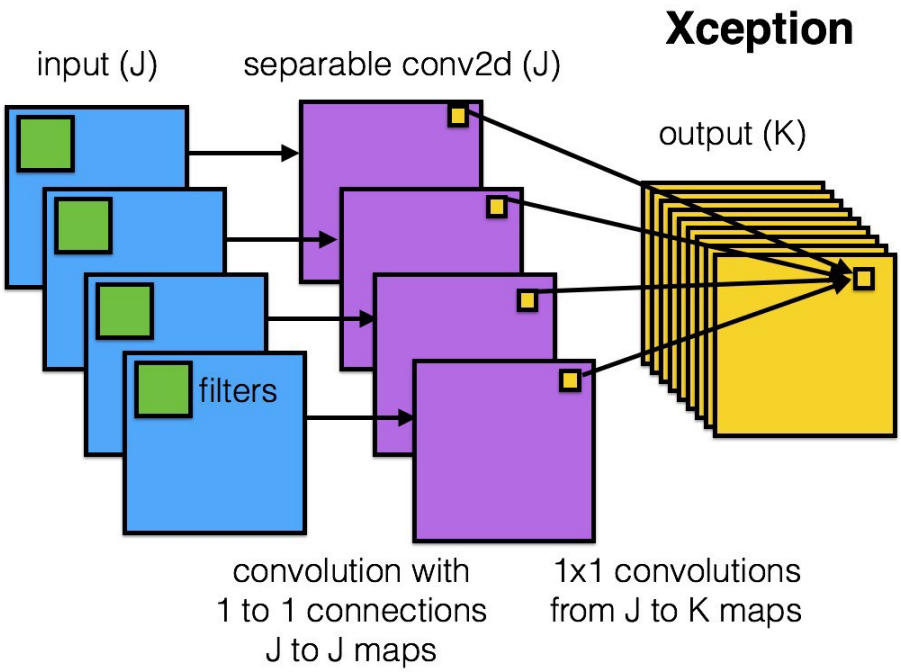
The modified depthwise separable convolution is the pointwise convolution followed by a depthwise convolution. This modification is motivated by the inception module in Inception-v3 that 1×1 convolution is done first before any n×n spatial convolutions. Thus, it is a bit different from the original one. (n=3 here since 3×3 spatial convolutions are used in Inception-v3.)

Two minor differences:

- The order of operations: As mentioned, the original depthwise separable convolutions as usually implemented (e.g. in TensorFlow) perform first channel-wise spatial convolution and then perform 1×1 convolution whereas the modified depthwise separable convolution perform 1×1 convolution first then channel-wise spatial convolution. This is claimed to be unimportant

because when it is used in a stacked setting, there are only small differences that appear at the beginning and at the end of all the chained inception modules.

- The Presence/Absence of Non-Linearity: In the original Inception Module, there is non-linearity after first operation. In Xception, the modified depthwise separable convolution, there is NO intermediate ReLU non-linearity.



Finally to wrap up we would craft this project to be user friendly with a Mobile Android App. This will display the respective words for the gestures made. This will rule out the need for a translator and instills a sense of confidence in the disabled person. This will enable him/her to communicate their troubles in times of distress. The proposed project promises high reliability and speed.

ANDROID MOBILE APPLICATION DEVELOPMENT

A user-friendly sign language translator is developed on a portable platform like Android Studio that may lead to a practical solution in helping the verbally-disabled to overcome the communication barrier in society.

TensorFlow Lite Gesture Classification

Overview

- This app performs gesture classification for sign language detection on live camera feed and displays the results in real-time on the screen.
- Application can run either on device or emulator.
- Build the demo using Android Studio

Platform: Android studio

The application is developed on Android Studio. Android Studio provides a unified environment where you can build apps for Android phones, tablets, Android Wear, Android TV, and Android Auto. Structured code modules allow you to divide your project into units of functionality that you can independently build, test, and debug in Java language.

Prerequisites

1. If you don't have one already, install Android Studio, following the instructions on their website.
2. You need an Android device and Android development environment with minimum API 21.
3. Android Studio 3.2 or later.
4. Name the .tflite file as model.tflite and the subset of labels should be named as labels.txt.

Building

1. Open Android Studio, and from the Welcome screen, select Open an existing Android Studio project.
2. From the Open File or Project window that appears, navigate to and select the tensorflow-lite/examples/gesture_classification/android directory. Click OK.
3. If it asks you to do a Gradle Sync, click OK.
4. You may also need to install various platforms and tools, if you get errors like "Failed to find target with hash string 'android-21'" and similar. Click the Run button (the green arrow) or select Run > Run 'android' from the top menu. You may need to rebuild the project using Build > Rebuild Project.
5. If it asks you to use Instant Run, click Proceed Without Instant Run.
6. Also, you need to have an Android device plugged in with developer options enabled at this point. See here for more details on setting up developer devices.
7. Generate a TFLite model file from google colab.
8. Copy the labels.txt and put it into the assets folder.

9. Now once you get the TFLite and label file, put that into the assets folder.

Code

The functionalities are achieved by creating different java files.
Imageclassifer.java file is used to classify images using Tensorflow lite.
Functions it performs:

1. Captures the live image and splits them into frames.
2. Retrieves the image specifications.
3. Smoothes the images by passing through filters.
4. Read the label list from the assets folder.

CameraActivity.java file is used to control the camera of the application
Functions it performs:

1. Takes permissions from the user to turn the camera on.
2. Creates intent to Imageclassifier.java and passes control to it if the permissions are granted.
3. Returns to the homepage if the permissions are not granted.

MainActivity.java file is the main file of the application where all the activities are controlled.
It performs:

1. Controls the UI widgets.
2. Manages other java files.
3. Creates intents for getting the camera on.
4. Sets control to Imageclassifier.java to perform machine learning and display the outcome on the screen.

Additional Note

Ensure that labels.txt and model.tflite files are added into the project which are downloaded from the web app. Also ensure the labels.txt file is not deleted or modified from the assets folder.

- As this is the first application of sign language recognition on a mobile platform, the scope is limited to basic sign alphabets. It recognizes 26 English alphabets along with space and no data.
- Our application has a very simple UI with “IOT_PROJECT!” and real time captioning starts as soon as one taps the button. An example is given below.


H
IOT_PROJECT


HE
IOT_PROJECT


HEL
IOT_PROJECT

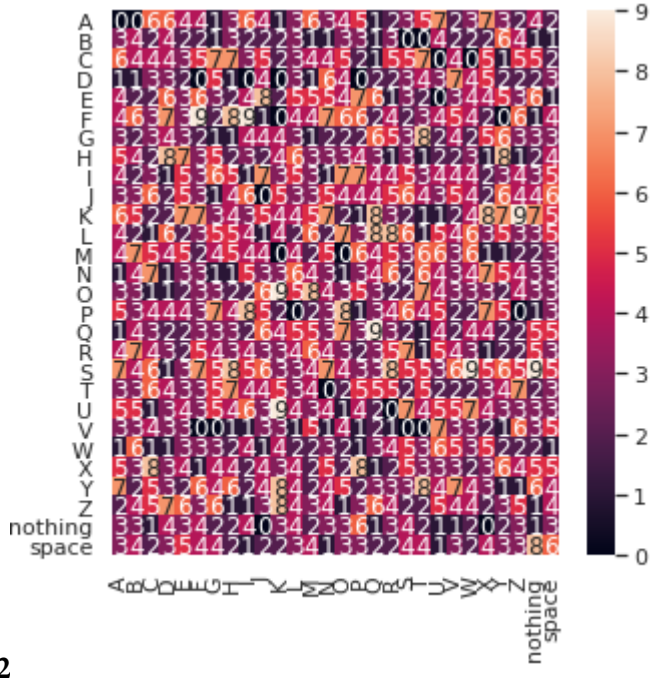

HELL
IOT_PROJECT


HELLO
IOT_PROJECT

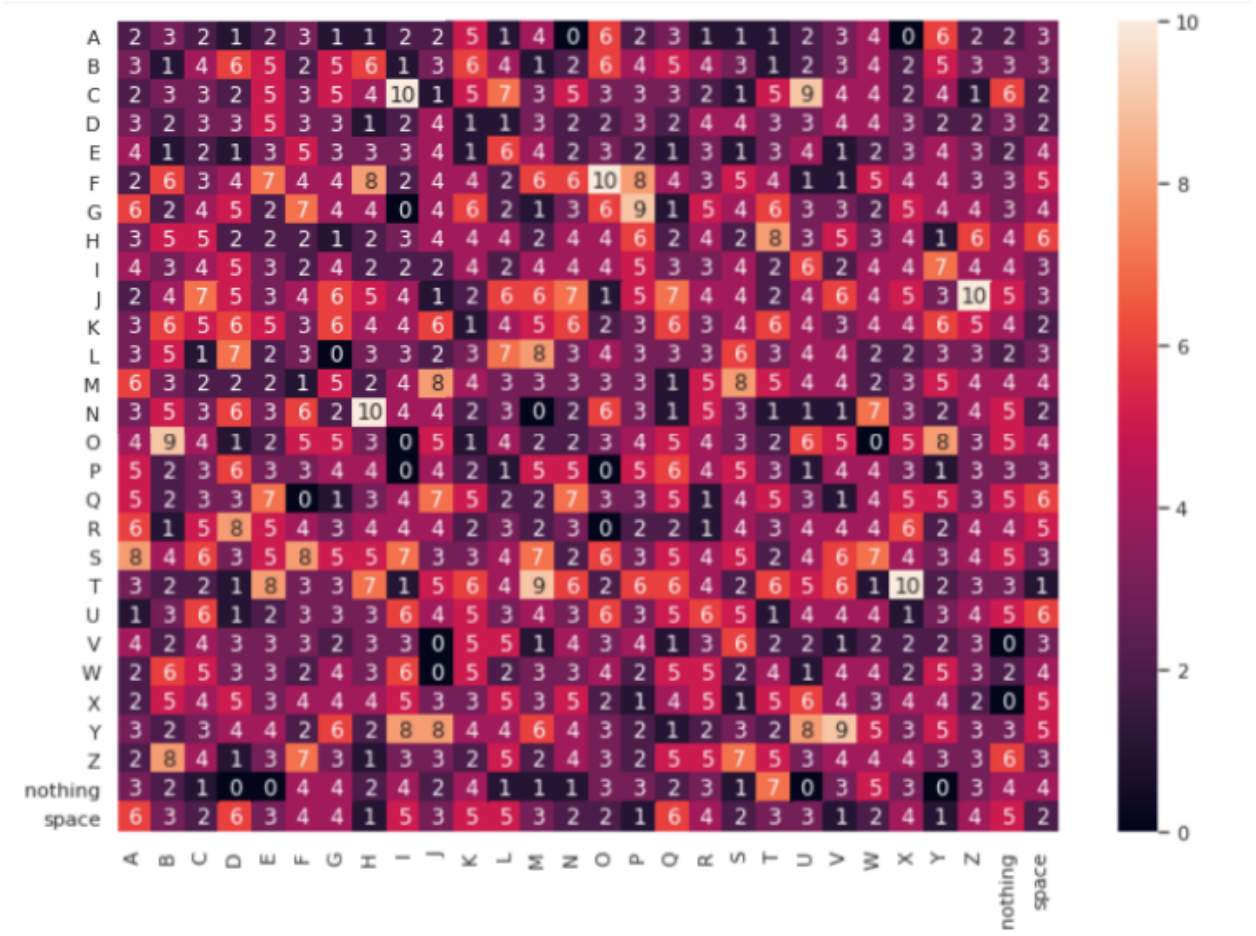
RESULTS AND DISCUSSION

The performance of 4 pretrained models with transfer learning and fine tuning the final layers can be compared using various metrics. It can be observed from the metrics that the

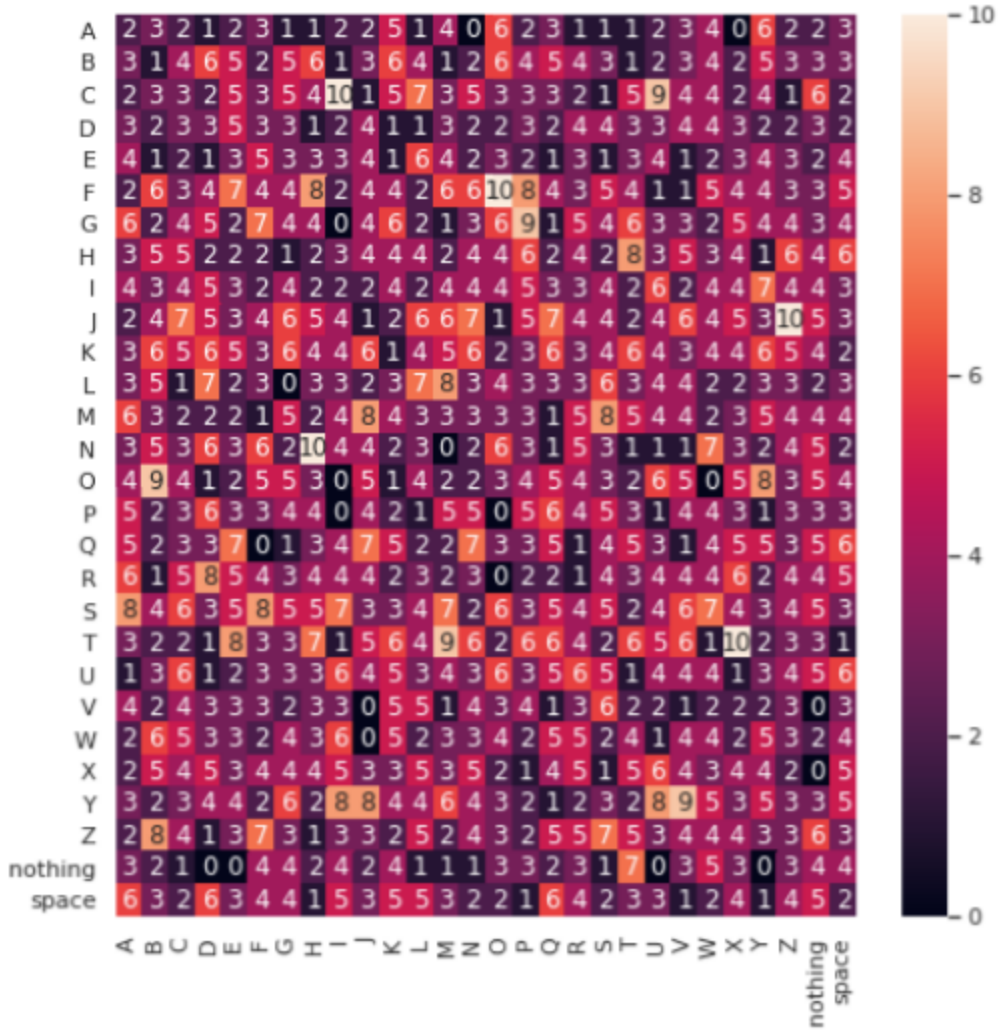
Model	Train Acc	Test Acc	Train Loss	Test Loss	F1-Score	Recall	Precision
Mobile Net V2	**0.9366**	**0.9228**	**0.2099**	**0.2973**	0.033	0.033	0.033
Inception V3	0.9598	0.9135	0.1420	0.3586	0.032	0.032	0.032
Xception	**0.9726**	**0.9060**	**0.0866**	**0.5302**			
VGG 16	**0.8428**	**0.8589**	**0.5195**	**0.5144**			



Confusion matrix of Mobile Net V2



Confusion matrix of Inception V3



Confusion matrix of Xception

CONCLUSION & FUTURE WORK

Inability to speak and hear is one major challenge for the human race. As an easy and practical way to achieve human-computer- interaction, this solution of gesture to speech and text conversion has been used to facilitate the reduction of hardware components. This solution aims to provide aid to those in need thus ensuring social relevance. The people can easily communicate with each other. The user-friendly nature of the system ensures that people can use it without any difficulty and complexity. The application is also cost efficient and eliminates the usage of expensive technology.

Future improvements to our framework may be to expand the application to a wider vocabulary. Further, more advanced algorithms may replace the current in place so as to improve the accuracy and processing speed of the system.