A Project Report

On

DESIGNING NEURAL NETWORK FOR IMAGE CATEGORIZATION

Submitted in partial fulfillment of the requirement of University of Mumbai For the Degree of

Bachelor of Engineering

in

INFORMATION TECHNOLOGY

Submitted by

Parth Rajput
Tejas Rahate
Rahul Bhosale
Kiran Nambiar

Supervisor

Prof. Varunakshi Bhojane

DEPARTMENT OF INFORMATION TECHNOLOGY

PILLAI COLLEGE OF ENGINEERING
NEW PANVEL - 410206

UNIVERSITY OF MUMBAI

Academic Year 2017-18

PILLAI COLLEGE OF ENGINEERING
NEW PANVEL - 410206
DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the requirements for the project entitled "**Designing Neural Network for Image Categorization**" have been successfully completed by the following project group students:

Group Member	Roll Number
Parth Rajput	A869
Tejas Rahate	A868
Rahul Bhosale	A802
Kiran Nambiar	A866

in partial fulfilment of Bachelor of Engineering of Mumbai University in the Department of Information Technology, Pillai College of Engineering, New Panvel-410 206 during the Academic Year 2017-2018.

Prof. Varunakshi Bhojane
Supervisor
Department of Information Technology

Dr. Sharvari Govilkar
Head,
Department of Information Technology

Dr. Sandeep M. Joshi
Principal
PCE, New Panvel

PILLAI COLLEGE OF ENGINEERING
NEW PANVEL - 410206
DEPARTMENT OF INFORMATION TECHNOLOGY

PROJECT APPROVAL for B.E

This project report entitled "**Designing Neural Network for Image Categorization**" by **Parth Rajput, Tejas Rahate, Rahul Bhosale, and Kiran Nambiar** is approved for the degree of B.E. in **Information Technology**.

Examiners:

1. _____

2. _____

Supervisors:

1. _____

2. _____

Chairman:

1. _____

Date:

Place:

PILLAI COLLEGE OF ENGINEERING
NEW PANVEL - 410206
DEPARTMENT OF INFORMATION TECHNOLOGY

Declaration

We declare that this written submission for B.E. Declaration entitled "**Designing Neural Network for Image Categorization**" represent our ideas in our own words and where others' ideas or words have been included. We have adequately cited and referenced the original sources. We also declared that we have adhere to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any ideas / data / fact / source in our submission. We understand that any violation of the above will cause for disciplinary action by institute and also evoke penal action from the sources which have thus not been properly cited or from whom paper permission have not been taken when needed.

Project Group Members:

Parth Rajput : _____

Tejas Rahate : _____

Rahul Bhosale : _____

Kiran Nambiar : _____

Contents

Abstract

This project explores the scope of Neural Networks in the field of Image Categorization. A Neural Network is defined as a computing system that is biologically-inspired programming paradigm which enables a computer to learn from observational data. It was designed with the intent that it will learn to perform various tasks without providing a task specific code. In other words it can be said that a single neural network can be used in various applications with minimal changes. Neural networks provide the best solutions to many problems in image recognition, speech recognition, and natural language processing. This project demonstrates the use of a neural network to categorize images based on various criteria. It aims at implementing an neural network and the required algorithms without any use of predefined functions other than image processing libraries. The key to the network, then, is figuring out effective transformations of the data to get good decisions.Appropriate image datasets will be used to train and test our models to ensure maximum accuracy and low error rate. This project may produce results that vary in accuracy based on the objects that need to be classified in the image and the type of image being used.

List of Figures

List of Tables

Chapter 1

Introduction

This project consists the working of three models (i.e. Inception model , MobileNet and Tensorflow lite) , they help us to accomplish the application of traffic sign recognition . This is integrated using the tensorflow framework which allows us to implement machine learning and deep learning into scripts .We perform the categorization of the image in real time using an android app and an effort for comparative study amongst these models was also made to check efficiency under various circumstances . Later on a text to speech module was also attached to increase accessibility .

1.1 Fundamentals Of Neural Networks

A Neural Network is a data and information processing system that is inspired by the way biological nervous systems, such as the human brain processes information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements called neurons working in synergy to solve task specific problems.ANNs, like people, learn by example. A neural network is always configured in order to perform task specific applications like identical pattern recognition or data categorization, through a learning process. Learning in biological systems consists of adjustments to the synaptic connections that are present between the neurons. This is a similar approach of neural networks as well.

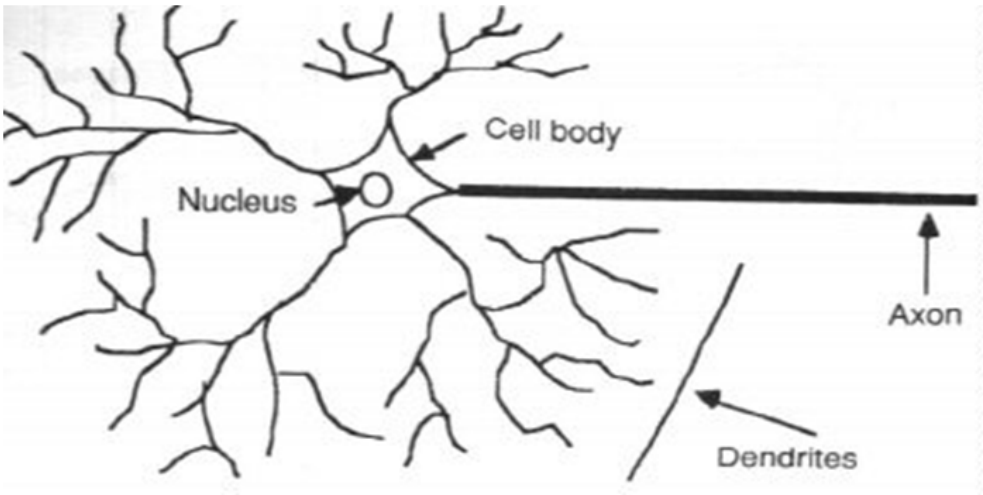

Figure 1.1: Structure of a Neuron

The Figure 1.1 shows the structure of a human brain, a conventional neuron receives signals from other neurons through a network of minute structures called dendrites.The neuron sends out spikes of electrical signals through a long and thin strand known as an axon, which splits into multiple branches. At the end of every axon , a structure known as a synapse converts the signal received from the axon into electrical effects that restrain the activity in the adjacent neurons.
At the point when the neuron gets boosts initiating input that is adequately more contrasted and its inhibitory information, it sends a spike of electrical movement through its axon. Learning inside the neurons happen by changing the proficiency of the synapse so the effect of a specific neuron does not mess with others.

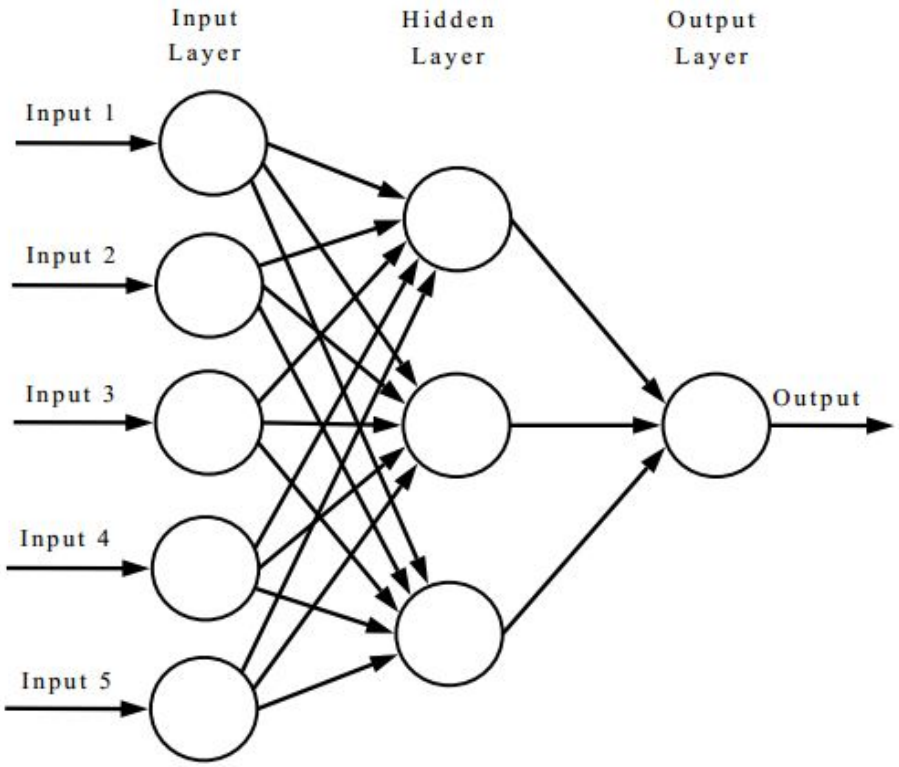

Figure 1.2: Structure of a neural network [5]

1.2 Objectives

The objective of this work is as follows:

1. To study and design a neural network without the use of external libraries or APIs to categorize images based on various criteria (Road Signs Classification).

2. To implement a neural network and the required algorithms without any use of predefined functions other than image processing libraries.

3. To train the network in a supervised way so that it is able to tag unclassified images.

1.3 Scope

The current projects on image categorization mainly make use of deep convolutional neural networks with many layers which make them difficult to deploy in embedded systems and other low resource areas, this project aims at exploring the scope of using a shallow convolutional neural network while still obtaining accurate results. While most projects that exist on this problem statement aim at expanding the scope of the application, this project focuses on designing the neural network itself. This project will not implement standard library function or packages and the project will be aimed at development of the necessary functions.

1.4 Outline

The report is organized as follows:

Chapter 1 The introduction is given in Chapter 1. It describes the fundamental terms used in this project. It motivates to study and understand the different techniques used in the field of Neural networks. This chapter also presents the outline of the objective of the report.

Chapter 2 It describes the Literature survey of the project, it describes about all the advancements in the field of Neural networks done so far.

Chapter 3 This chapter presents the theory and proposed work. It describes the major approaches used in this project work.

Chapter 4 This Chapter shows some of the application areas where neural networks can be deployed.

Chapter 5 We summarize the project, its objectives and outcomes in Chapter 5.It describes the major approaches used in this work. it describes of how the system works in order to achieve the expected results.

Chapter 2

Literature Survey

2.1 Introduction

We have surveyed a variety of papers on image recognition. These papers discuss various convolutional networks and various related concepts. Our project is inspired by these concepts and makes use of convolutional networks for the purpose of image categorization. A short description on each of these papers has been given below

2.1.1 Multi-GPU convolutional network (Krizhevsky et al., 2012)

The researchers have trained a huge, deep convolutional neural network to categorize the high-resolution image dataset of 1.2 million from the LSVRC-2010 ImageNet contest and categorized it into 1000 different classes. Processing the test data, they achieved top 1 and top 5 error rates of 37.5% and 17.0% which is considerably way better than the earlier state-of-the-art version . This particular neural network, consists of 650,000 neurons and 60 million parameters, also consisting of five different convolutional layers, out of which some are succeeded by max-pooling layers, and 3 fully-connected layers completely with a final 1000-way softmax. To train the model faster, they used non saturating neurons and a very productive GPU implementation for the convolution implemention. A top 5 test gave a result of 15.3% error rate was achieved. [9]

2.1.2 COTS HPC unsupervised convolutional network (Coates et al., 2013)

This paper consists scaling up of deep learning algorithms which have been proved to increase performance and efficiency in benchmark tasks and also helps us to discover complex high level features in the provided images . Current efforts to train models with quite gigantic networks consisting of over 1 billion parameters have also relied on computing infrastructure based on cloud containing thousands of CPU cores. This paper also consists of results and technical details from their own system based on the Off-The-Shelf High Performance Computing (COTS HPC) technology. It consists of a cluster of GPU servers with MPI and Infini-band interconnects . This model is able to train networks with 1 billion parameter on 3 different machines in a few days, and hence can show that it can scale up to networks using just 16 machines with over 11 billion parameters .This approach enables us much widespread research capability with extremely huge neural networks hence, this particular infrastructure is way easier to marshal by others.[1]

2.1.3 GoogleNet (Szegedy et al., 2014a)

They proposed a profound convolutional neural network infrastructure called as Inception, which was in charge of setting the new best in class for categorization and detection in the Large-Scale Visual Recognition Challenge 2014 (ILSVRC14) ImageNet . The primary sign of this infrastructure is the enhanced use of the figuring assets inside the system. This was accomplished by a carefully made plan that takes into consideration for the increasing depth and width of the system while keeping the computational spending steady

2.1.4 TensorFlow: Large-scale machine learning on heterogeneous systems(Martn Abadi, Ashish Agarwal, et al.,2015.)

TensorFlow is a flexible data flow based programming model, as well as single machine and distributed implementations of this programming model. The system is borne from real-world experience in conducting research and deploying more than one hundred machine learning projects throughout a wide range of Google products and services.A TensorFlow calculation is depicted by a coordinated diagram, which is made out of an arrangement of nodes. The diagram shows to an data flow computation, with augmentations for enabling a few sorts of nodes to maintain and refresh persistent state and for branching and looping control structures inside the graph in a way like Naiad . [10]

2.2 Literature Summary

we are trying to implement same google net and replicate our own version of it

Table 2.1: Summary of literature survey

SN	Paper	Advantages and Disadvantages
1	Multi-GPU convolutional network (Krizhevsky et al., 2012)	GPU clusters were used to accelerate computing and first time top 5 error rates were lower than existing technologies Advantage: 15.3 error rate, GPU introduced Disadvantage: significant performance degrade if network changed
2	COTS HPC unsupervised convolutional network (Coates et al., 2013)	COTS can classify human faces, body(person) and animals requires GPU clusters for multiple days. Advantages: GPU acceleration efficiently implemented Disadvantage: Classification is limited to certain type of images.
3	GoogLeNet (Szegedy et al., 2014a)	GoogleNet has 22 layers, can be re-trained to classify wide range of objects Advantages: Optimized computing, scalable, efficient GPU utilization Disadvantage: Designed for competition hence retraining is limited to types of classes in IMAGENET dataset.
4	TensorFlow: Large-scale machine learning on heterogeneous systems(Martn Abadi, Ashish Agarwal, et al.,2015).	Advantage: Structure of the computation graphs and behavior of machine learning models is understood using a visualization tool called TensorBoard. Disadvantage: Applications are practically limited to deep learning since it lacks solid understanding of mathematical concepts and a steep learning curve where other methods are more profound .

Chapter 3

CatNet- Categorization Network

CatNet converts the real time feed into frames and passing it through convolution layer of the neural network.Batch normalization is performed in order to simplify the variations in the inputs for the hidden layers .Next, the ReLu layer helps to eliminate negetive values by replacing zeros and the result is displayed in the screen including a tts voice output .

3.1 Overview

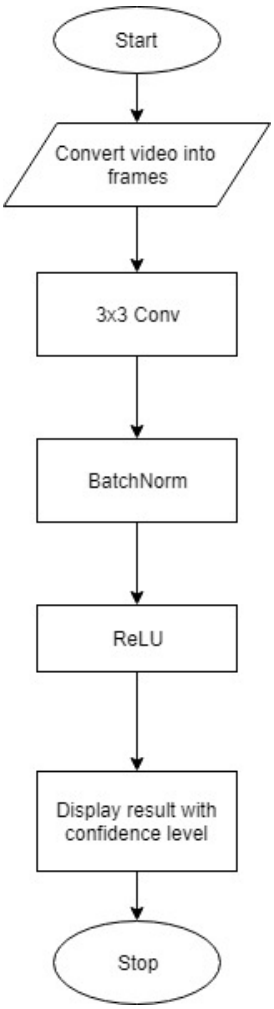

Figure 3.1: Flowchart of CatNet

Figure 3.1 shows the approach being used to implement the system. This architecture gives a brief explanation of the working of the proposed system.[12]

As seen in the above flowchart, an image that is to be categorized will first be pre-processed. This would include various image enhancement operations and removal of background noise. As in most systems, the partitions(segments) of the image that is significant for categorization, are segmented for greater accuracy.
Based on these partitions, features will be extracted from them. This can be understood with the example of the image of a No Parking road sign. The features that may be extracted from such an image would be the shape, color, text, etc. As various images are bound to have some number of similar features, they are processed to determine which features define a particular class.[2]
In the next stage, the image is finally classified. the image is labelled in various possible classes along with their probabilities. The confidence for the classification is calculated and compared with the results of classification of the image with other standard models. Based on this comparison a deviation is calculated to determine the performance of the model. All the results of image classification and deviation is displayed to the user.

3.1.1 Existing System Architecture

The system architecture for the existing system is as follows:

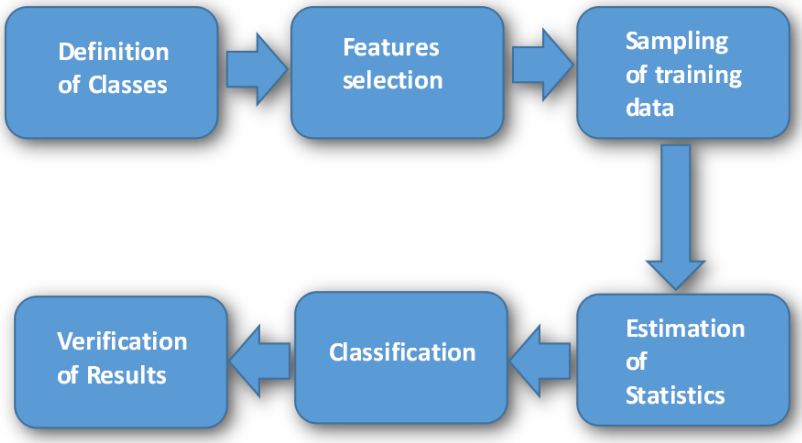

Figure 3.2: Existing system architecture used for Image Categorization

The road signs identification process is performed in six steps, each of which is handled by a separate component:

- **Definition of classes:** The main responsibility of the component is to represent the types of road signs (e.g.regulatory signs, warning signs, guidance signs, etc) coming from various information sources and datasets.
- **Feature selection:** This module collects data and tries to extract the features of the road signs. Usually, this is achieved through machine learning techniques, which are able to extract features based on the shape of the road sign, its colour,etc. [6]
- **Sampling of training data :** This module from the knowledge known from the features which were extracted , tries to identify various other road signs. It is the

most crucial component in identification process as good training of data results in better results.

- **Estimation of statistics :** The output from the previous step is analysed and the a estimation is made for which sign it is depending upon the support and confidence level percentage. For better accuracy it is required that ample amount of training is done on testing dataset.
- **Classification :** Depending on the various estimation given, the given road sign image is classified into the various classes of road sign. [8]
- **Verification of results :** This is the final step in the process of identification of the image. Based on the class the system has classified and the support and confidence level shown the user verifies the given result.

3.1.2 Proposed System Architecture

The previous system requires the user to supply a pre-processed dataset. It required image processing to be performed and supply the processed images as input to the neural network. This saved the need for extra convolution layers but increased the overhead of processing the images before using them. In order to save this overhead we allow input of colored images. In order to achieve better domain results, we used the existing system architecture as a template to build upon and make suitable changes to achieve the necessary functions and add new advantages to the system while removing the previous disadvantages. [11]

3.2 Implementation Details

3.2.1 Technique

1. First we acquire the image to be classified using a file explorer inside our application and display it for user confirmation.
2. The next step is to perform data preprocessing tasks on the image such as converting the image into necessary color scheme, perform bounding and clipping, scaling of aspect ratio to standardize the images used as input for the first layer.
3. Input Layer (First Layer) will be fed with full resolution image to be classified.
4. The stack comprising a combination of several layers is used [3]

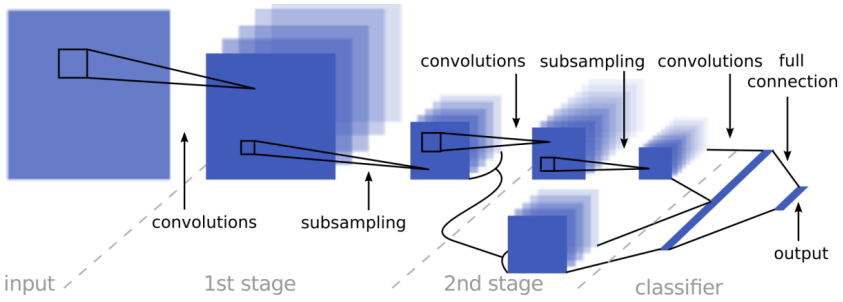

Figure 3.3: Architecture of Network

- **Convolution:** Filtering is used in this layer where the Feature matrix is lined up with image and corresponding image pixels are multiplied, added and averaged.

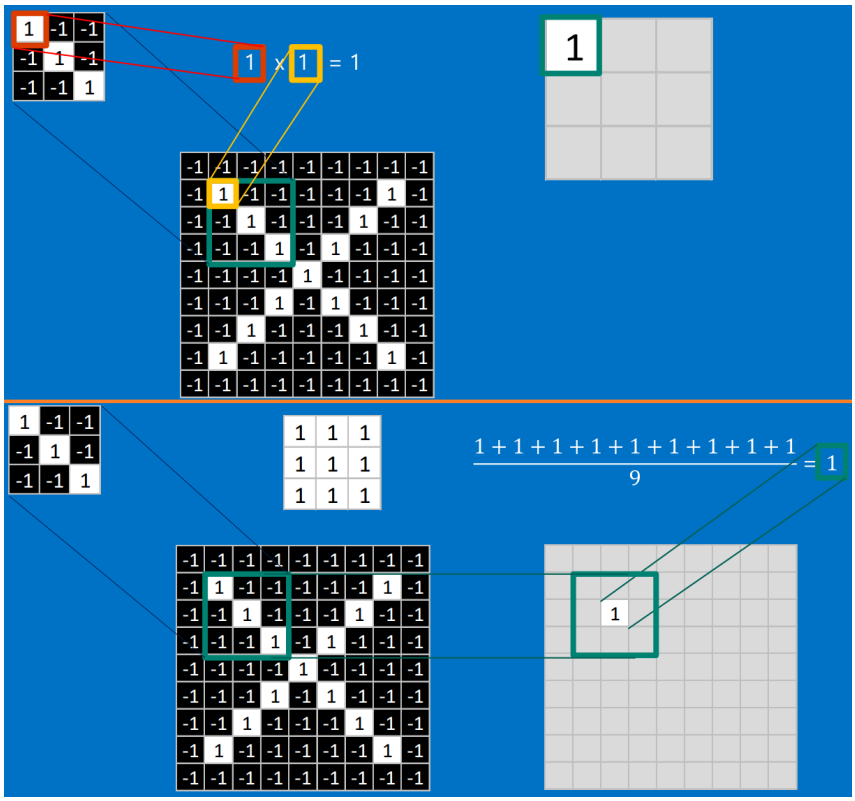

Figure 3.4: Filtering and Convolution

- **Pooling:** We shrink the image by max-pooling discretizing technique, we choose a window size of 3X3 and a stride of 3. This window is walked over the filtered image and will output the maximum value from that window, thus generating a sub-sampled image. [7]

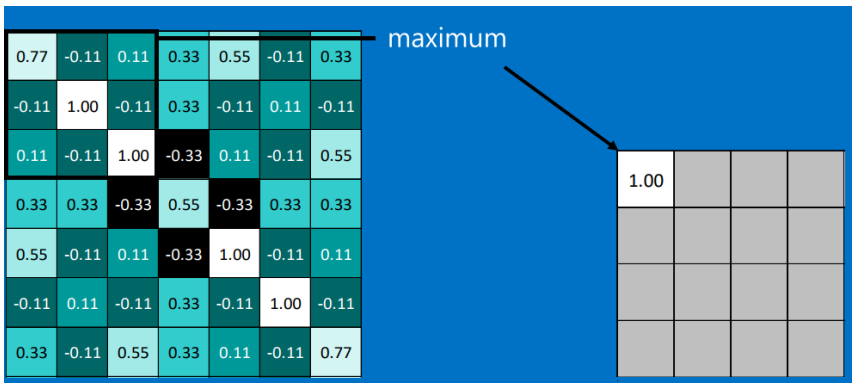

Figure 3.5: Max-Pooling

- **ReLU:** Rectified Linear Units replace all the negative values with zeros.
- **Fully Connected Layer:** The final feature matrices are flattened and each value

5. The First convolution layers detect simple features (eg. edges) and the higher layers combine these simple features at different spatial positions. [4]

6. The final layer will be output layer which will display classes that the image falls into and its confidence.

7. To learn the features in convolution layers and the voting weights in fully connected layers we use back propagation along with Gradient Descent

$$Error = CorrectO/P - ActualO/P \tag{3.1}$$

where correct=[1] and actual=0-1

8. The same image will be classified using other (existing) models and the output will be shown alongside the proposed systems output for a comparative analysis.

3.2.2 Model

TensorFlow is an open-source programming library for dataflow programming over a scope of various tasks. It is a math library based on symbols, and furthermore utilized for machine learning applications, for example, neural systems. The reference execution keeps running on single devices, TensorFlow can keep running on various CPUs and GPUs (with conditional CUDA and SYCL for general-purpose computing on graphics processing units). It is utilized for both research and production at Google.
Its calculations are communicated as stateful dataflow charts. Google additionally gives RankBrain built over tensorflow which handles a considerable number of pursuit inquiries, supplanting and supplementing customary static algorithm-based results.

Inception V3 Google recently released a model called Inception v3 with Tensorflow.This model can be retrained by training the last layer per specific categorization requirements. It is named as Inception v3 due to the Inception modules it uses, which are basically mini models inside the bigger model. A similar Inception architecture was used in the GoogLeNet model which was a state of the art image recognition network in the year 2014.The user can make the decision as to what type of convolution the user wants to make at each layer.This can be done by performing each convolution in parallel and concatenating the final feature map results before proceeding to the next layer.

MobileNet They are low-power ,low-latency and small model which is implemented in order to meet the resource constraints of use cases covering a more success rate. They can be built based on for detection, categorization, segmentation and embedding which is similar to the implementation of other popular heavy duty models (eg. Inception) which are used. Mobile Net can be implemented on mobile devices where TensorFlow Mobile is previously installed but at the same time trades off between accuracy, size and latency while comparing in favor with all the popular models from the literature review.

TFlite TensorFlow Lite is a lightweight platform for embedded gadgets and mobile devices. It enables on-device machine learning inference with a small binary size and low latency. TFLite also supports hardware acceleration within the Neural Networks API supported by Android. It also supports a set of core operators(both quantized and float) which have been specifically tuned for mobile platforms. They incorporate previously

fused activators and bias functions to further improve accuracy and performance. Adding to its functionality, TensorFlow Lite also supports custom operations in implemented models. TFLite has a new interpreter which is optimised for mobile platforms , which has the key primary goals of keeping apps short and efficient. This interpreter uses a custom (less-dynamic) memory allocator and a static graph ordering method to ensure minimal load, execution latency and initialization .

The the above mentioned models were used in the implementation of the android application which is to be used as the final product.

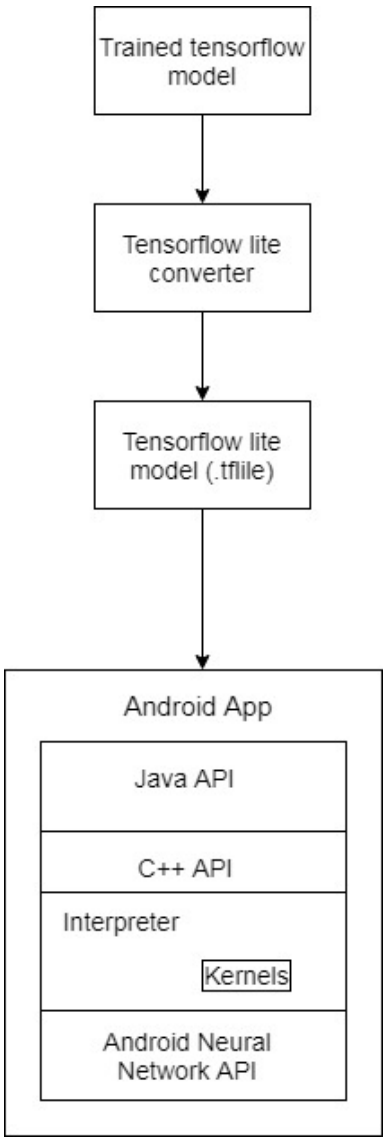

Figure 3.6: Architecture of Tensorflow Lite

3.2.3 Implementation Method

- **Overview :** We will implementing transfer learning, which implies we are beginning with a model that has been as of now prepared on another issue. We will then retrain it on a comparable issue. Deep Learning starting with no outside help can take days, yet transfer learning should be possible in shorter timeframe. We will utilize a model prepared on the ImageNet Large Visual Recognition Challenge dataset.
  These models can separate between 1,000 distinct classes. We will have a decision of model structures, so you can decide the correct tradeoff between speed, size and precision for the problem at hand. We will utilize this same model, however retrain it to differentiate few classes in light of our own illustrations.

- **Setup :** Introducing tensorflow and making a usable dataset. This includes either finding a current picture dataset (for this situation traffic signs) or making another dataset form scratch which is worked by undertaking particular intentions. Duplicate all the code and pictures into the primary working index.

- **Training the network :** The script used for training can either retrain Inception V3 or the MobileNet model. The foremost distinction is that Inception V3 is upgraded for higher accuracy, while the MobileNets are improved to be little and proficient, at the cost of poor accuracy. Inception V3 has an accuracy of 78% on ImageNet, however is the model is 85 MB, and requires commonly more handling than even the biggest MobileNet design, which accomplishes 70.5% accuracy using a package download of just 19MB.

  - Two configurations are available to train the particular models.
    - **Input image resolution:** 128,160,192, or 224px. Unsurprisingly, feeding in a higher resolution image takes more processing time, but results in better classification accuracy. We recommend 224 as an initial setting.
    - The relative size of the model as a fraction of the largest MobileNet: 1.0, 0.75, 0.50, or 0.25. We recommend 0.5 as an initial setting. The smaller models run significantly faster, at a cost of accuracy.
  - Using the recommended configuration, it usually takes only a few minutes to retrain on a laptop. You will pass the settings inside Linux shell variables.
    The graph below shows the first-choice-accuracies of these configurations (y-axis), vs the number of calculations required (x-axis), and the size of the model (circle area).16 points are shown for mobilenet.
    For each of the 4 model sizes (circle area in the figure) there is one point for each image resolution setting. The 128px image size models are represented by the lower-left point in each set, while the 224px models are in the upper right. [13]
  - Tensorboard : Its main purpose is to monitor the training progress. Before starting the training, start tensorboard and keep running it in the background. It is a monitoring and inspection tool which is included with tensorflow.
  - The ImageNet models are made up of many layers stacked on top of each other, a simplified picture of Inception V3 from TensorBoard .These layers are pretrained and are already very valuable at finding and summarizing information that will help classify most images. In this case we are training only the last layer ie. final_training_ops . Whereas all the preceding layers retain their previously trained state.

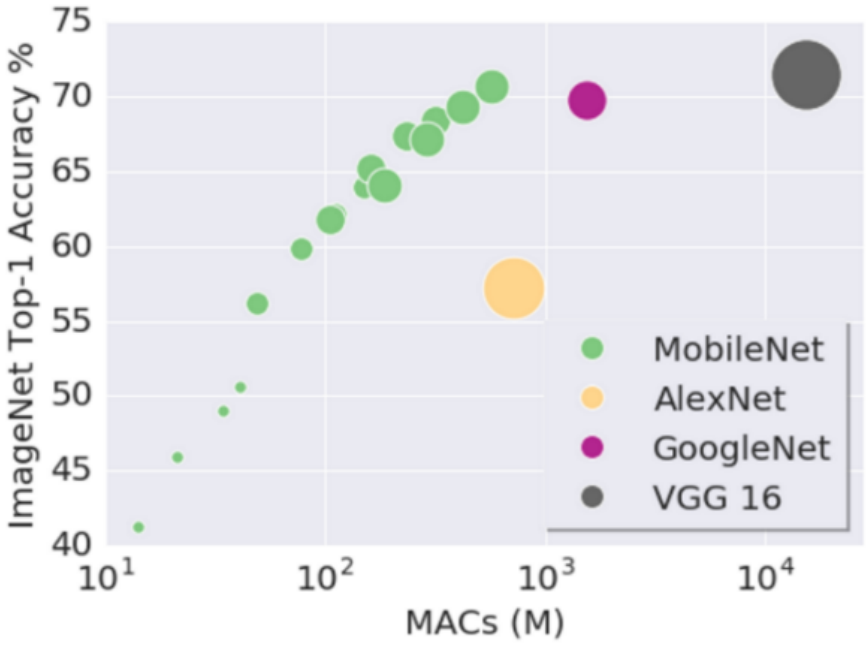

Figure 3.7: Comparison of Various Models

– A bottleneck is a casual term we frequently use for the layer just before the last output layer that really does the categorization. "Bottleneck" isn't utilized to suggest that the layer is slowing the overall system. We utilize the term bottleneck on the grounds that it is nearer to the output, the portrayal is considerably more conservative than in the fundamental body of the system. Every picture is reused under different circumstances during training. Figuring the layers behind the bottleneck for each picture takes a lot of time. Since these lower layers of the system are not being changed their yields can be stored and reused. So the content is running the consistent piece of the system, everything beneath the node is marked as bottleneck above, and caching the result . The command you ran spares these documents to the "bottlenecks/" registry. In the event that you rerun the content, they'll be reused, so you don't need to sit tight for this part once more.

– Once the content wraps up all the bottleneck records, the genuine preparing of the last layer of the system starts. The preparation works effectively by sending the stored value for each image into the Bottleneck layer. The genuine name for each picture is likewise sustained into the node named GroundTruth. Simply these two sources of information are sufficient to figure the classification probabilities, preparing updates, and the different performance metrics.As it trains you'll see a progression of step output , every one indicating accuracy, validation accuracy, and the cross entropy:

  * The training accuracy shows the percentage of the images used in the current training batch that were labeled with the correct class.
  * Validation accuracy: The validation accuracy is the precision (percentage of correctly-labelled images) on a randomly-selected group of images from a different set.
  * Cross entropy is a loss function that gives a glimpse into how well the

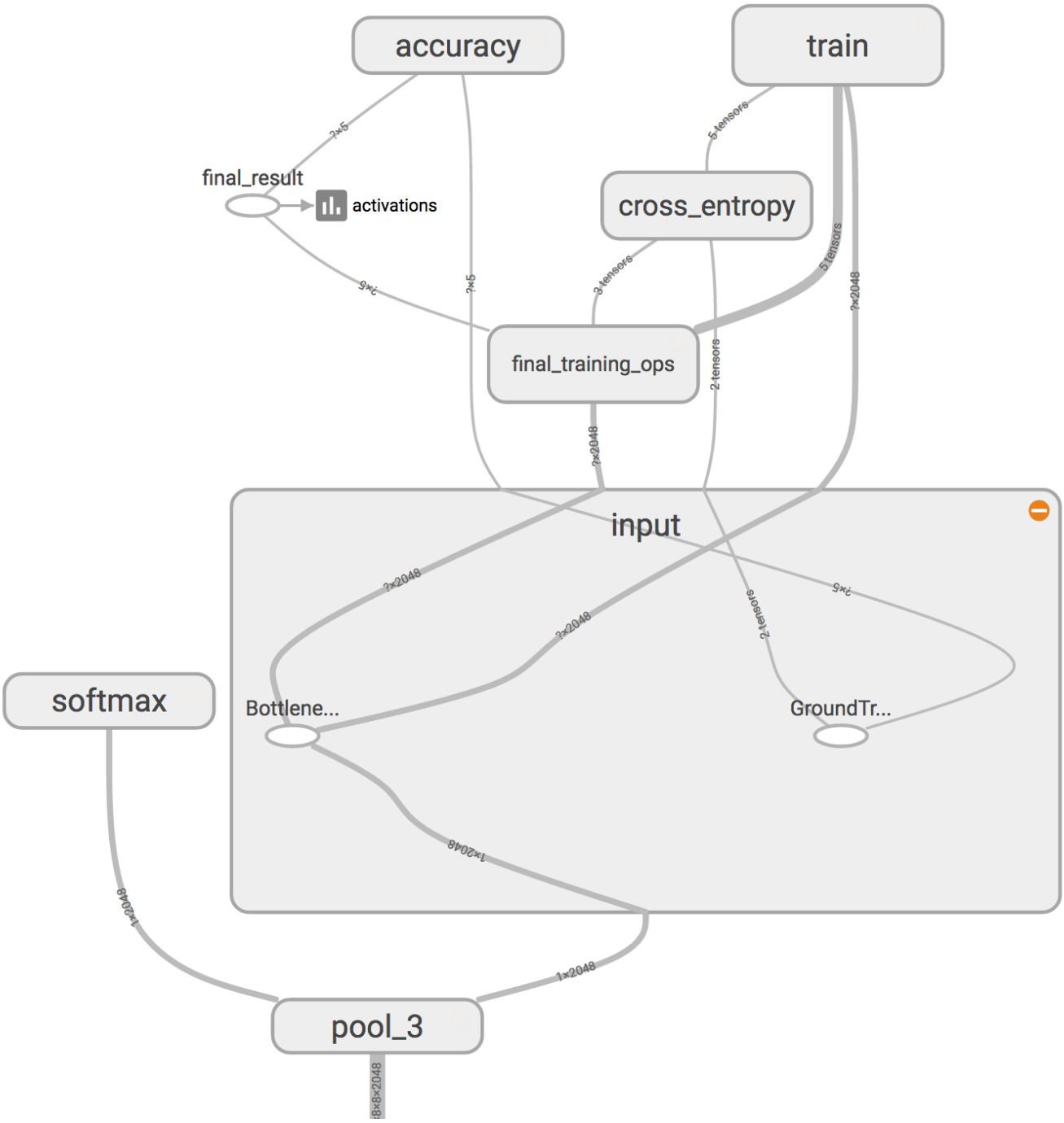

Figure 3.8: The Training Process

learning process is progressing (lower numbers are better here).

- **Testing:** Next, check that the model is creating normal outcomes before deciding to change it.The "scripts/" registry contains a straightforward command line script, label_image.py to test the system. Presently we'll test label_image.py on a specific image.Now test the model. In the event that you are utilizing an alternate design you should set the "- - input_size" flag.
  The content will print the likelihood the model has allocated to each type of traffic signs .This ought to ideally deliver a sensible best mark for your illustration. You'll be utilizing this command to ensure despite everything you're getting sensible out-

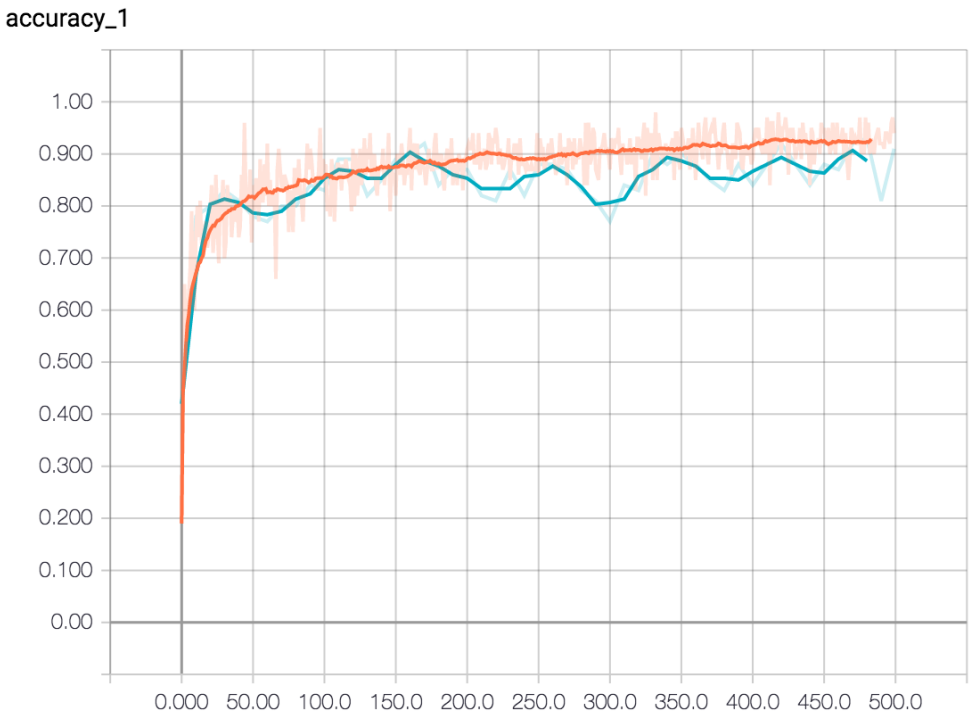

Figure 3.9: Accuracy {Orange:training Blue:Validation }

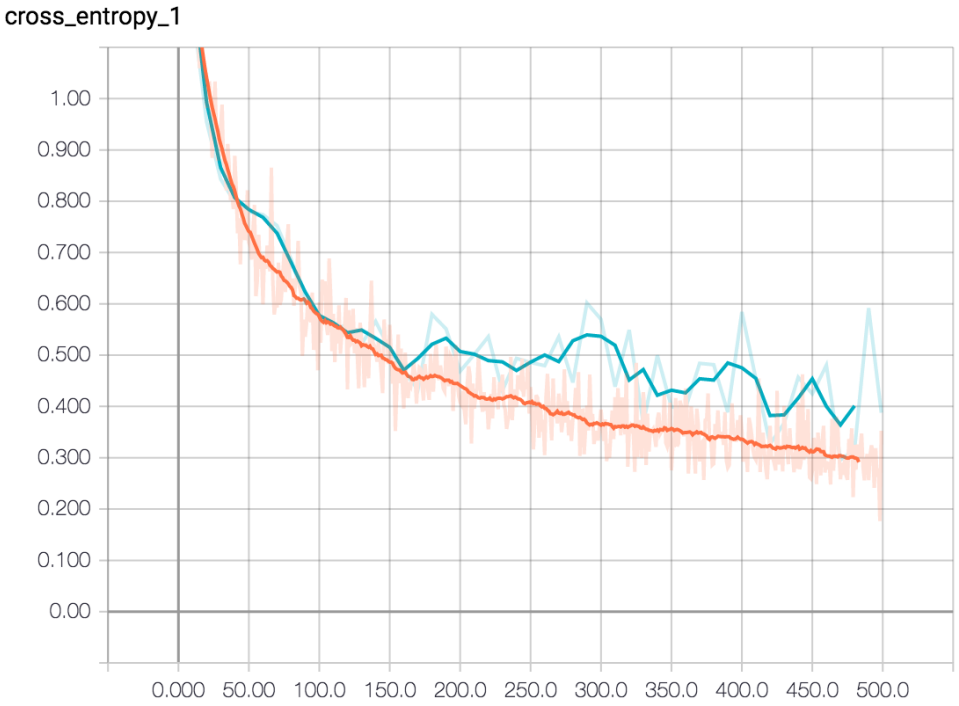

Figure 3.10: Cross Entropy {Orange:training Blue:Validation }

comes as you do additionally handling on the model file to set it up for use in a mobile application.

- **Setup for android app:** Download and install android studio and in the file selector, choose tensorflow-for-poets-2/android/tfmobile from your working directory.Set up an Android device or set up the emulator with camera access.

- **Running the customized app :** The default application setup categorization pictures into one of the 1000 ImageNet classes, utilizing the standard MobileNet, without the retraining we did. Now we should alter the application so that the application will utilize our retrained morel for our custom dataset .The demo venture is designed to look for a graph.pb, and a labels.txt documents in the android/tfmobile/resources registry. Supplant those two documents with your versions.
  The TensorFlow Interface utilized by the application requires that you request your outcomes by name. The application is as of now set up to read the output of the baseline MobileNet, named "MobilenetV1/Predictions/Softmax". The output node for our model has an alternate name: "final_result". Open ClassifierActivity.java and replace the OUTPUT_NAME variable .

Figure 3.11: Demonstration of the app detecting the "no u turn" traffic sign

Figure 3.12: Demonstration of the app detecting the "speed limit 80" traffic sign

Figure 3.13: Demonstration of the app detecting the "stop" traffic sign

- **Text-To-Speech :** Android allows the user to convert the available text into vocal outputs .By providing voice backup it also allows the output to be in multiple languages . Android libraries provides a separate TextToSpeech class for this purpose. For implementing this class we have to create an instance of the object of this class and specify the Listener along with its properties such as its language ,pitch etc. Once the traffic sign is successfully categorized by the inception model , the app would also provide a voice output reading the the sign which is provided in the above mentioned class.

3.2.4 Possible Implementation Issues

- This git directory should contain three other subdirectories

  The android/tfmobile/ registry contains all the files needed to construct a simple Android application that categorizes images as input which is coming from the camera. The only files missing for the application are the ones which define the image categorization model .

  The scripts/ registry contains the python scripts which we will be using throughout this implementation. These scripts include code for preparation , testing and evaluation of the model.

  The tf_files/ directory consists of the files which were generated in the first section . At the bare minimum you should have all of the following files consisting the re-trained tensorflow program.

- To avoid any issues caused by the unsupported training ops, the TensorFlow installation consists of a tool, optimized for interpretation, it also removes all nodes that are irrelevant for a given set of input and outputs.
  The code also performs a few different optimizations that helps to speed up the model, such as combining explicit batch normalization operations inside the convolutional weights to lessen the number of ongoing calculations. This can give a 30% speed increase, depending on the input model being implemented.

- If the user implements some other kind of model the input size flag has to be reset likewise .This should hopefully prepare a sensible top label for the example. The user will be using certain commands to make sure that the model is giving sensible results as you do further processing on the model file in order to prepare it for implementing it in a mobile app.

- One way the TensorFlow library is kept at a minimum , for mobile application and it done so by only supporting the subset of operations that are commonly utilized during inference. This is a an acceptable approach, as training is usually not conducted on mobile platforms.
  Similarly it also does not support for operations with huge external needs.By default, most graphs consist training ops that is not supported by the mobile version of TensorFlow . TensorFlow does not load a graph that consists an unsupported operation even if the unsupported operation is irrelevant for inference.

3.2.5 Use Case Diagram / Activity Diagram

The functional blocks of the system are explained below.

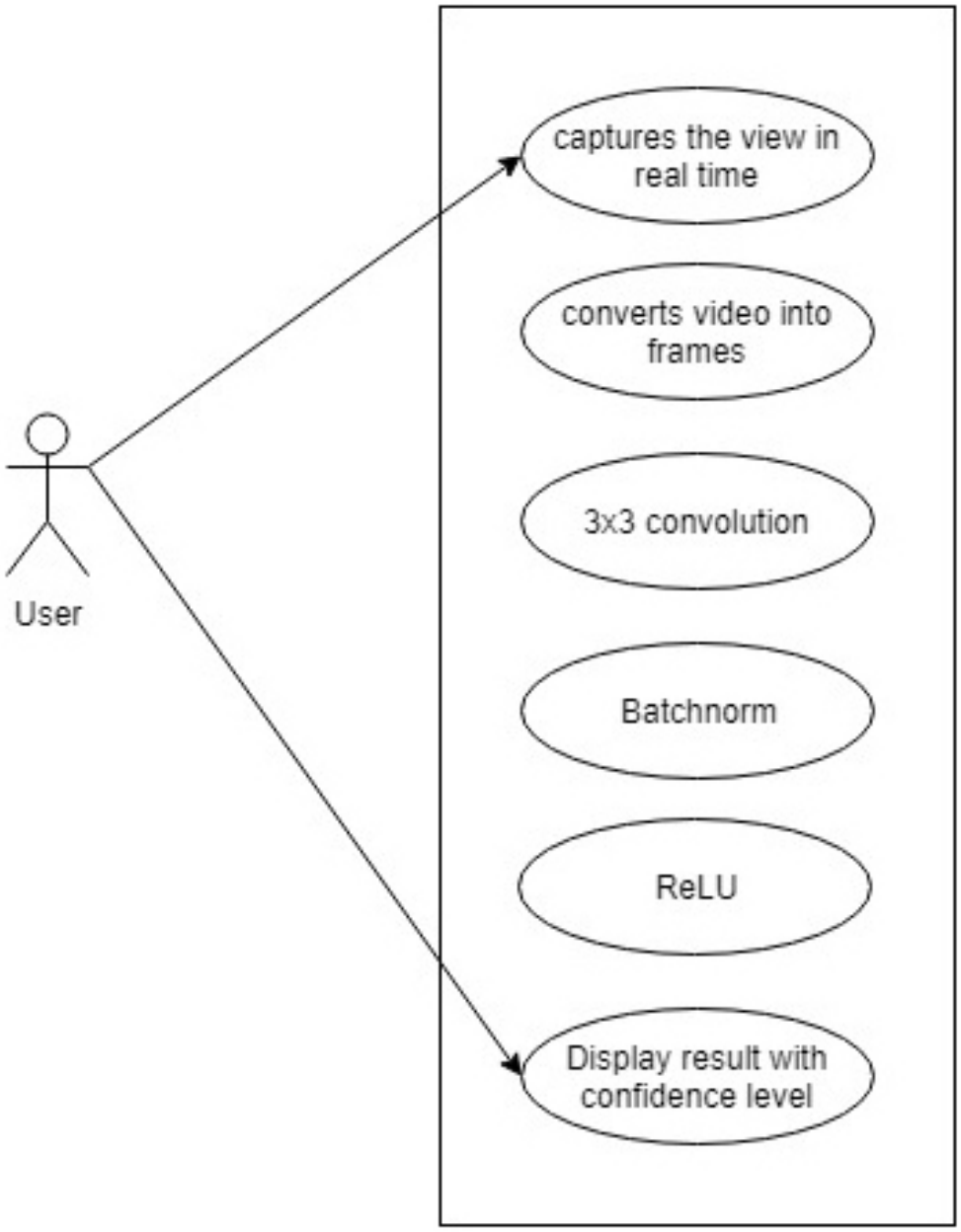

Figure 3.14: Use-Case Diagram

User Module : User uploads Image
Pre-processing Module : System Pre-processes Image
Segmentation module : System performs segmentation

Table 3.1: User Module

User Case	Descriptor
Actor	User
Precondition	Input image should be available.
Main Scenario	User uploads image.
Extension Scenario	If the image is not compatible, Not possible to upload file.
Post Condition	Image successfully uploaded

Table 3.2: Pre-processing module

User Case	Descriptor
Actor	User
Precondition	Upload input image.
Main Scenario	Pre-processing is carried out by thresholding the image colors and limiting them to those necessary for recognition
Extension Scenario	If the image contains colors totally different from that trained prompt generated.
Post Condition	Extract characters before segmentation

Table 3.3: Segmentation module

User Case	Descriptor
Actor	User
Precondition	Pre-processed image should be available.
Main Scenario	The pre-processed input image is segmented into isolated partitions by assigning a number to each partition using a labelling process. This labeling provides information about number of characters in the image. Each individual partitions is uniformly resized into pixels
Extension Scenario	If the image is not compatible. Not possible to upload file.
Post Condition	Image successfully uploaded

3.2.6 Sample Dataset Used

To classify images we need a training set of images from standardized set. Necessary datasets were obtained from sources with all the necessary permissions.

Table 3.4: Sample Dataset Used for Experiment

Dataset	Items	Type
Street Sign India	600	Information

100 images per class were used to train this particular model .These are some sample images of their respective classes .

Figure 3.15: Sample images from "No Entry" class

Figure 3.16: Sample images from "No Parking" class

Figure 3.17: Sample images from "No U Turn" class

Figure 3.18: Sample images from "Stop" class

Figure 3.19: Sample images from "Speed Limit 80" class

Figure 3.20: Sample images from "Yield" class

3.2.7 Hardware and Software Specifications

The experiment setup is carried out on a computer system which has the different hardware and software specifications as given in Table 3.5 and Table 3.6 respectively.

Table 3.5: Hardware details

Processor	2 GHz Intel
SSD	120 GB
RAM	16 GB

Table 3.6: Software details

Operating System	Ubuntu 14/ Windows 10
Programming Language	Python 2.7, Java.
Platform	Android
IDE	Android studio
Frameworks	Tensorflow

3.3 Performance Evaluation Metrics

The quality of a domain system can be evaluated by comparing the proposed system with that of established systems that can classify images with great accuracy.

Support: Support is an indication of how frequently the items appear in the dataset .

Confidence : Confidence indicates the number of times the if/then statements have been found to be true. Confidence is given in Equation 3.2.

$$conf(X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)} \tag{3.2}$$

3.3.1 Result Evaluation Table

A test dataset of 240 images was created which consists of 40 images dedicated to each type of traffic signs . These images were tested individually to create the confusion matrices and to determine the accuracy for each traffic sign.

Table 3.7: Accuracy for detection of all trained classes

sr.no	Traffic Sign	no. of signs detected successfully	total images	accuracy
1	No Entry	40	40	100%
2	No Parking	28	40	70%
3	No U Turn	35	40	87.5%
4	Stop	30	40	75%
5	Speed Limit 80	35	40	87.5%
6	Yield	28	40	70%

Table 3.8: Confusion Matrix for "No Entry" traffic sign

No Entry	True	False
Positive	40	0
Negetive	5	195

Table 3.9: Confusion Matrix for "No U Turn" traffic sign

No U Turn	True	False
Positive	28	10
Negetive	5	197

Table 3.10: Confusion Matrix for "No Parking" traffic sign

No Parking	True	False
Positive	35	10
Negetive	10	185

Table 3.11: Confusion Matrix for "Stop" traffic sign

Stop	True	False
Positive	30	6
Negetive	4	200

Table 3.12: Confusion Matrix for "Yield" traffic sign

Yield	True	False
Positive	35	15
Negetive	5	185

Table 3.13: Confusion Matrix for "Speed Limit 80" traffic sign

Speed Limit 80	True	False
Positive	28	5
Negetive	15	192

3.3.2 Result Evaluation Graphs

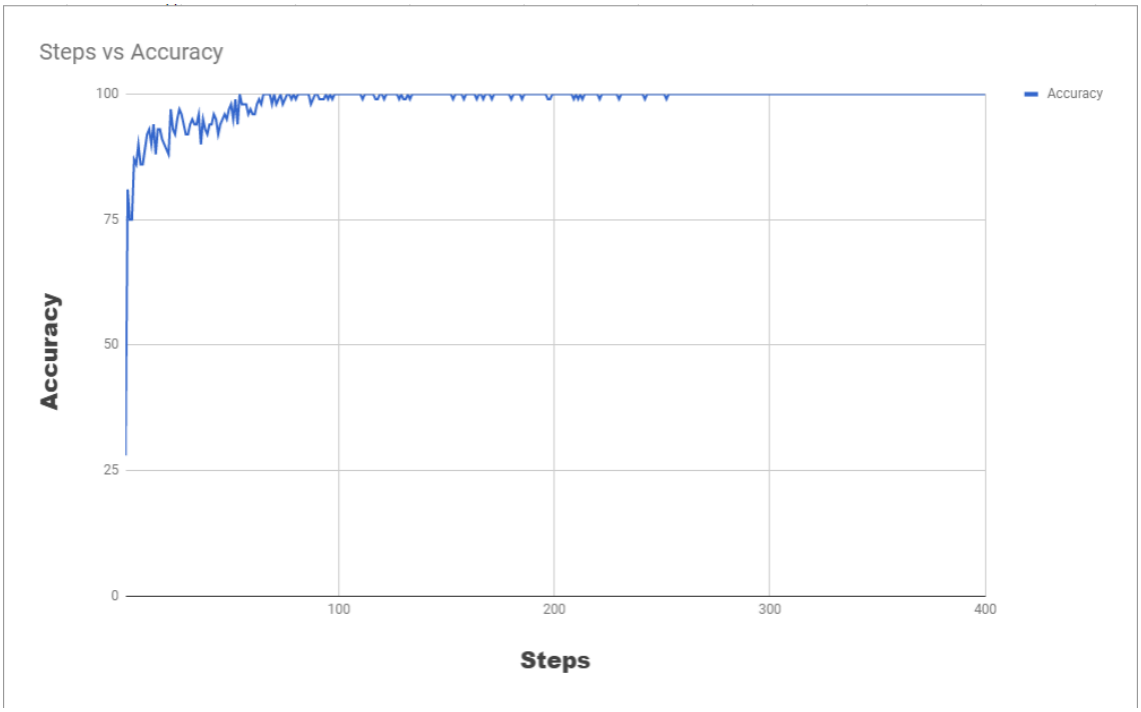

Figure 3.21: Accuracy of the implemented model

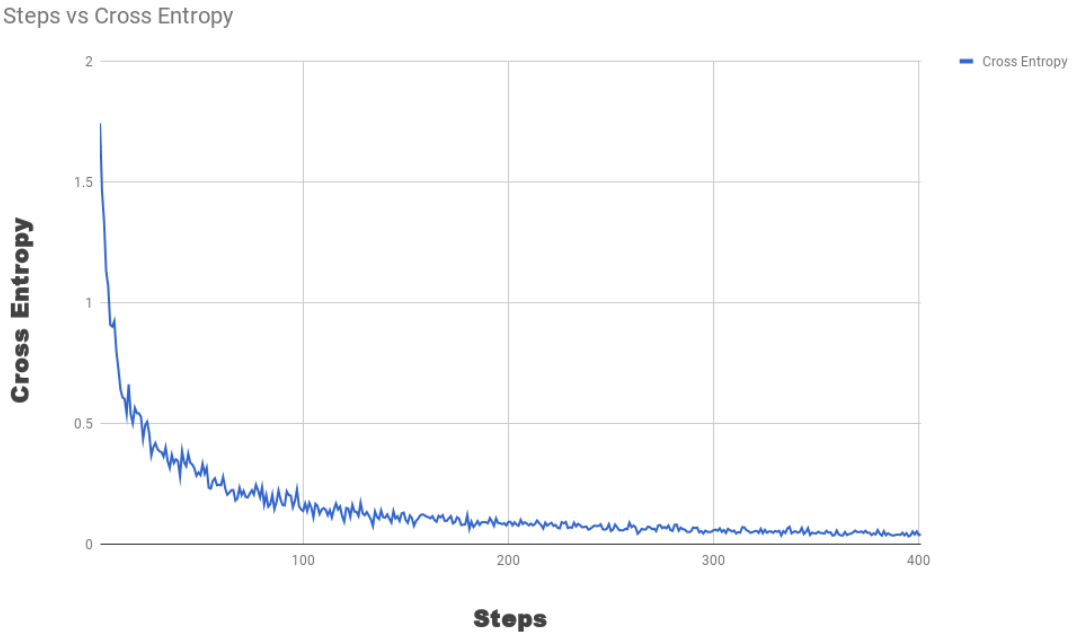

Figure 3.22: Cross entropy of the implemented model

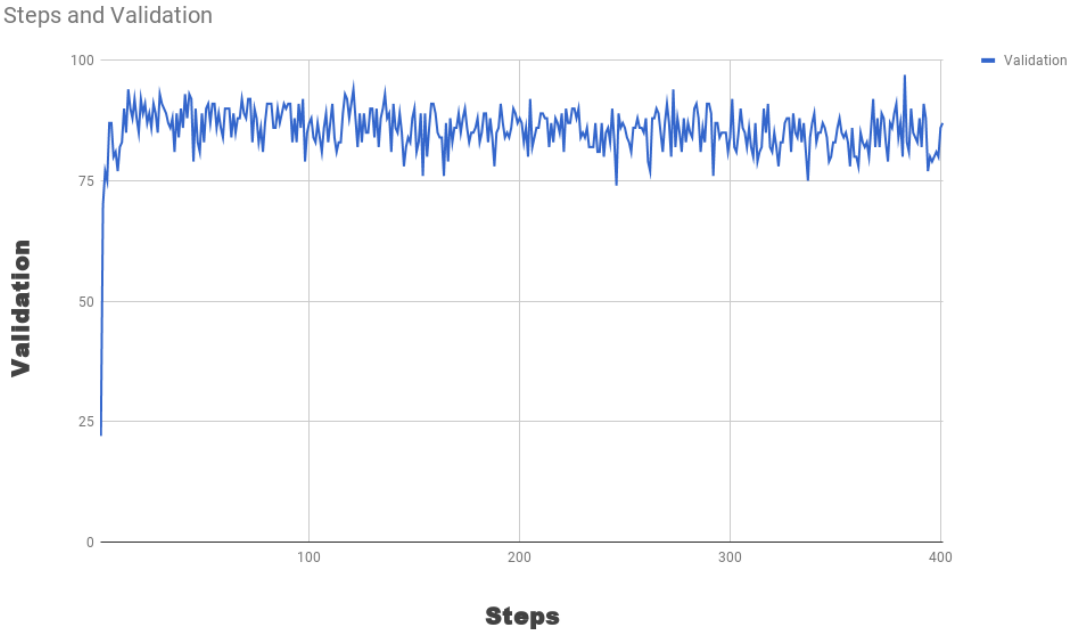

Figure 3.23: Validation of the implemented model

Chapter 4

Applications

There are various applications of this domain system. The application is listed here.

4.1 Traffic and Road sign recognition

This system first searches signs within images captured by the sensor on the vehicle, and then identifies road signs helping the driver of the vehicle to properly drive the vehicle. It identifies between the different types of road signs based on shape of the sign (circle,triangle, square) and the shapes inside the recognised border . Besides it can be used to recognize traffic. The model can be used to recognize vehicles a single person or a crowd on the road.
Road sign recognition using android app is a support for self driving cars that can be used to notify and warn the self driving car. Examples for such identification of road signs are speed limit or no parking indications. The present investigation targets the recognition of road and traffic signs in real-time. Real-time video is taken by a digital camera from a moving vehicle and real world road signs are then extracted using vision-only information.The system is based on two stages, one performs the detection and another one is for recognition. In the first stage, a hybrid color segmentation algorithm has been developed and tested. In the second stage, an introduced robust custom feature extraction method is used for the first time in a road sign recognition approach. Finally, a multilayer artificial neural network (ANN) has been created to recognize and interpret various road signs. It is robust because it has been tested on both standard and non-standard road signs with significant recognition accuracy.

4.2 Military Application

With the ability to detect the objects in real-time, it can be used in robots or drone to identify the targets or to identify the campsites for delivery of essential supplies to the soldiers. Processing, analysis, and understanding of visual information is important in many military situations. Intelligent recognition can aid robots in making decisions and are able to relieve humans of the burden of interpreting image and video data. Development of image sensor technologies and embedded computer systems, together with

development of image analysis methods, has enabled implementation of intelligent computer vision-based systems, which enable visual detection, recognition, and tracking of objects in real-time. Applications include systems for navigation of autonomous vehicles, automatic target recognition, missile guidance, smart helmets to detect enemy soldiers, and battle control systems in command centers. Development in sensor technologies bring new sensors with increasingly high resolutions, high low-light sensitivity, and ability to acquire images in multiple spectral ranges including visible, infrared and thermal images.

4.3 Remote sensing

Remote sensing relies to the interaction of electromagnetic radiation with matter. In remote sensing, fuzzy neural networks have been used for a variety of applications such as military reconnaissance, flood estimation, crop prediction, mineral detection, and oil exploration . Active systems such as SAR can infiltrate clouds that block the view of passive systems, like the multispectral and panchromatic sensors.

4.4 Google Net

The image processing method are also implemented using the mobileNet model which classifies images based on classes determined by training dataset (CIFAR-10) and extracts relevant features to determine what class does the image belong to. This model minimises human error which may occur while specifying classification features .
Convolutional neural networks are the state of the art technique for image recognition-that is, identifying objects such as people or cars in pictures. While object recognition comes naturally to humans, it has been difficult to implement using machine algorithms and until the advent of convolutional neural networks (beginning in earnest with the development of LeNet-5 in 1998) the best computer was no match for the average child at this deceptively challenging task. Recent advances in CNN design, notably deeper models with more layers enabled by the availability of cheap computing power and enhanced techniques such as inception modules and skip connections, have created models that rival human accuracy in object identification. Moreover, CNNs are poised to make real-world impacts in areas from self-driving vehicles to medical imaging evaluation (a field where computers are already outperforming humans).

4.5 Automated taxi routing

Similar system is applied in vehicle stabilization while automated driving . Google and Tesla have adapted the technologies to center the car according to the lane borders tracked using the image processing models and using neural networks to predict the required position of the car and setting the car variables accordingly .Such an applications are also applied to valet parkings and automated taxi routing .

4.6 Real-Time Object Detection

YOLO[7] is a real-time object detection system. On a Titan X, it processes images at 40-90 FPS. It uses a single neural network to the full image. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. Hence similar method is used to implement image processing over videos.
Research shows that the detection of objects like a human eye has not been achieved with high accuracy using cameras and cameras cannot be replaced with a human eye. Detection refers to identification of an object or a person by training a model by itself. Detection of images or moving objects have been highly worked upon, and has been integrated and used in commercial, residential and industrial environments. But, most of the strategies and techniques have heavy limitations in the form of computational resources, lack of proper data analysis of the measured trained data, dependence of the motion of the objects, inability to differentiate one object from other, and also there is a concern over speed of the movement and lighting conditions . Hence, there is a need to draft, apply and recognize new techniques of detection that tackle the existing limitations .

Chapter 5

Conclusion and Future Scope

5.1 Conclusion

In this report, the study of different approaches for Categorization of Images is presented. These approaches make use of the concepts of Neural networks, of which models such as convolutional networks have been discussed. Various other approaches have also been mentioned. The comparative study of various techniques mentioned above is presented in this report. For this project, Image Categorization is being implemented with the help of a convolution network with multiple layers. The network uses three main layers which are found to be especially useful for image classification techniques viz. Convolution Layer, Rectified Neuron Layer and Max Pooling Layers. We use filtering techniques in convolution layer to generate feature matrices which will detect the edges of street signs.The performance measures like confidence and support are described in this report. The different standard datasets or variable inputs are defined that may be used in experiment for this domain systems. One of the datasets identified for this project is the Government dataset for Road-signs. The applications of this domain is identified and presented.

In our project, the dataset has been generated once, saved and used every time to train the neural network. The network can be retrained using different dataset of street signs. The model is lightweight and can be used on embedded systems and mobile applications, it requires less computing resources compared to other popular convolution models and the accuracy of this model will be compared with that of other popular models. The end product is a desktop application with a GUI, displaying all the necessary details such as classified labels, support confidence and comparative study of other models which can be used for a variety of purposes.

5.2 Future Scope

In the future we can expect the same model to be trained for applications across multiple domains and not just automated driving cars . The application of the model is not just limited to one particular type of dataset and can be retrained for various other purposes .The feature extraction process is performed on the images irrespective of what is consists , it breaks down the image into basic shapes and appearance to extract features hence the type of dataset is not a compulsion .

The applications in industry will firther help fingerprint or retina recognition, processing records of security or traffic cameras. The applications can be spaned in medicine include ultrasound imaging, magnetic resonance. Stereography is the art of using two almost identical photographs to create a three-dimensional (3D) image.The viewer requires special glasses or a stereoscope to see the 3D image. With good image categorization, it has applications in motion picture and television industry to record the required footage.

A major challenge for automatic image analysis is that the sheer complexity of the visual task which has been mostly ignored by the current approaches. New technological breakthrough in the areas of digital computation and image categorization has relevance for future applications of image processing. The satellite imaging and remote sensing applications programs of the future will feature a variety of sensors orbiting the earth. This technology is required for military and other types of surveillance and can vastly improved using this particular concept.

References

[1] Brody Huval Adam Coates. Cots hpc unsupervised convolutional network. 2013.

[2] Huizhong Chen. Robust text detection in natural images with edge-enhanced maximally stable extremal regions. 2011.

[3] Yangqing Jia Christian Szegedy, Wei Liu. Going deeper with convolutions. 2011.

[4] Darknet. Yolo: Real-time object detection. 2012.

[5] Stack Exchange. [https://tex.stackexchange.com/questions/132444/diagram-of-an-artificial-neural-network]. 2015.

[6] Karl Moritz Hermann. Teaching machines to read and comprehend. 2013.

[7] Geoffrey E. Hinton. How neural networks learn from experience. 2012.

[8] Yangqing Jia Jeff Donahue. Decaf: A deep convolutional activation feature for generic visual recognition. 2012.

[9] Alex Krizhevsky. Imagenet classification with deep convolutional neural networks, advances in neural information processing systems 25. 2012.

[10] Ashish Agarwal Martn Abadi. Tensorflow: Large-scale machine learning on heterogeneous systems. 2015.

[11] Quoc Le Tomas Mikolov. Distributed representations of sentences and documents. 2013.

[12] M. Priya Sasikumar Gurumurthy, Balakrushna K. Tripathy. Study of image recognition using cellular associated artificial neural networks. *International Journal on Pattern Recognition*, 1:1–5, 2011.

[13] Safat B. Wali. An automatic traffic sign detection and recognition system based on colour segmentation, shape matching, and svm. 2012.

Publications

The paper " DESIGNING NEURAL NETWORK FOR IMAGE CATEGORIZATION " is based on this project. It has been posted on researchgate.net.

Abstract : This project explores the scope of Neural Networks in the field of Image Categorization. A Neural Network is a computing system that is biologically-inspired from a biological neuron, for the purpose of enabling a computer to learn from observational data. It was designed with the intent that it will learn to perform various tasks without providing a task specific code. In other words it can be said that a single neural network can be used in various applications with minimal changes. Neural networks provide the best solutions to many problems in image recognition, speech recognition, and natural language processing. This project demonstrates the use of a neural network to categorize images of Traffic Signs. It aims at implementing a neural network model and implementing it in an Android Application for real time detection of Traffic Signs. The key to the network, then, is figuring out effective transformations of the data to get good decisions. Due to the lack of appropriate image datasets, a custom image dataset was created using images found online. These images were used to train and test our models to ensure maximum accuracy and low error rate. This project produces results that vary in accuracy based on the objects that need to be classified in the image and the type of image being used.

A copy of the paper is enclosed with this document

The link to the paper is
https://www.researchgate.net/publication324686176_ DESIGNING_ NEURAL_NETWORK_ FOR_ IMAGE_ CATEGORIZATION

Acknowledgment

We sincerely express our gratitude to our Mentor Prof. Varunakshi Bhojane for guiding us through our project. Her counsel and advice have made it easy for us to achieve our goals in a more informative and productive manner.
We further thank the HOD Dr. Sharvari Govilkar and Dr.Madhumita Chatterjee, the faculty for allowing us to pursue this project and also in helping us to guide and decide between the plethora of options the college had to offer .
We would also like to thank Dr.Sandeep M. Joshi providing the required resources and guidance for the project , without his support this project would not have been possible and we are very grateful for his encouragement.

Parth Rajput
Tejas Rahate
Rahul Bhosale
Kiran Nambiar

Appendix

- **Android Archive (AAR) :** An Android library is structurally the same as an Android app module. It can include everything needed to build an app, including source code, resource files, and an Android manifest. However, instead of compiling into an APK that runs on a device, an Android library compiles into an Android Archive (AAR) file that one can use it as a dependency for an Android app module. Unlike JAR files, AAR files can contain Android resources and a manifest file, which allows you to bundle in shared resources like layouts and drawables in addition to Java classes and methods.

- **Batch Normalization :** Batch Normalization is responsible for normalizing a tensor by mean and variance, and applies (optionally) a scale γ to it, as well as an offset β.
  It helps achieve accuracy with fewer training steps. Batch Normalization allows us to use much higher learning rates and be less careful about initialization. It also has a beneficial effect on the gradient flow through the network, by reducing the dependence of gradients on the scale of the parameters or of their initial values.

- **Bottlenecks :** The bottleneck in a neural network is a layer with less neurons then the layer below or above it. Having such a layer encourages the network to compress feature representations to best fit in the available space, in order to get the best loss during training.
  In a CNN (such as Google's Inception network), bottleneck layers are added to reduce the number of feature maps (channels) in the network, which otherwise tend to increase in each layer. This is achieved by using 1x1 convolutions with less output channels than input channels.

- **Convolution Layer :** The convolutional layer is the core building block of a CNN. The layer's parameters consist of a set of learnable filters (or kernels), which have a small receptive field, but extend through the full depth of the input volume. During the forward pass, each filter is convolved across the width and height of the input volume, computing the dot product between the entries of the filter and the input and producing a 2-dimensional activation map of that filter. As a result, the network learns filters that activate when it detects some specific type of feature at some spatial position in the input.
  Stacking the activation maps for all filters along the depth dimension forms the full output volume of the convolution layer. Every entry in the output volume can thus also be interpreted as an output of a neuron that looks at a small region in the input and shares parameters with neurons in the same activation map.

- **Cross Entropy :**Cross entropy is a loss function that gives a glimpse into how well the learning process is progressing . A lower value is generally preferable for Cross Entropy.

- **Gradle :** Gradle is a build system that takes the best features from other build systems and combines them into one. It is improved based off of their shortcomings. It is a JVM based build system, what that means is that you can write your own script in Java, which Android Studio makes use of.

- **Gzip compression :** Gzip is based on the DEFLATE algorithm, which is a combination of LZ77 and Huffman coding. DEFLATE was intended as a replacement for LZW and other patent-encumbered data compression algorithms which, at the time, limited the usability of compress and other popular archivers. "gzip" is often also used to refer to the gzip file format.

- **Image Quantization :** Quantization, involved in image processing, is a lossy compression technique achieved by compressing a range of values to a single quantum value. When the number of discrete symbols in a given stream is reduced, the stream becomes more compressible. For example, reducing the number of colors required to represent a digital image makes it possible to reduce its file size. Specific applications include DCT data quantization in JPEG and DWT data quantization in JPEG 2000.

- **Inception :** Inception is a Deep Convolutional Neural Network architecture, had set-up the new state of the art for classification and detection in the ImageNet Large-Scale Visual Recognition Challenge 2014 (ILSVRC 2014). The main hallmark of this architecture is the improved utilization of the computing resources inside the network. This was achieved by a carefully crafted design that allows for increasing the depth and width of the network while keeping the computational budget constant. To optimize quality, the architectural decisions were based on the Hebbian principle and the intuition of multi-scale processing.

- **MobileNets :** MobileNets are a class of efficient models that were particularly designed for mobile and embedded vision applications. MobileNets are based on a streamlined architecture that uses depth-wise separable convolutions to build light weight deep neural networks. MobileNets come in various sizes controlled by a multiplier for the depth (number of features), and trained for various sizes of input images.

- **optimize _ for _ inference :** To avoid problems caused by unsupported training operations, the TensorFlow installation includes a tool, optimize_ for_ inference, that removes all nodes that are not needed for a given set of input and outputs.

- **Pooling Layer :** Pooling is a form of non-linear down-sampling. There are several non-linear functions to implement pooling among which max pooling is the most common. It partitions the input image into a set of non-overlapping rectangles and, for each such sub-region, outputs the maximum. The intuition is that the exact location of a feature is less important than its rough location relative to other features. The pooling layer serves to progressively reduce the spatial size of the representation, to reduce the number of parameters and amount of computation in the network, and hence to also control overfitting. It is common to periodically insert a pooling layer between successive convolutional layers in a CNN architecture. The pooling operation provides another form of translation invariance.

- **RelU :** ReLU (Rectified Linear Unit) is an activation function.The rectifier is, as of 2018, the most popular activation function for deep neural networks. The ReLu

function is as shown below.

$$A(X) = max(0, x) \tag{5.1}$$

It gives an output x if x is positive and 0 otherwise. ReLu is nonlinear in nature and combinations of ReLu are also non linear. ReLu is less computationally expensive than tanh and sigmoid because it involves simpler mathematical operations. That is a good point to consider when we are designing deep neural nets.

- **Softmax :** In mathematics, the softmax function, or normalized exponential function, is a generalization of the logistic function that "squashes" a K-dimensional vector z of arbitrary real values to a K-dimensional vector σ(z) of real values in the range (0, 1) that add up to 1.

- **Talk Back (Text-to-Speech) :** Android allows you convert your text into voice. Not only you can convert it but it also allows you to speak text in variety of different languages.
  Android provides TextToSpeech class for this purpose. In order to use this class, you need to instantiate an object of this class and also specify the initListener.

- **Tensorboard :** Tensorflow computations, like training a massive neural network, can be complex. For this purpose, suite of visualization tools called TensorBoard is used. Tensorboard makes it easier to understand debug and optimize tensorflow programs. One can use TensorBoard to visualize your TensorFlow graph, plot quantitative metrics about the execution of your graph, and show additional data like images that pass through it.

- **Tflite :** TensorFlow Lite is TensorFlows lightweight solution for mobile and embedded devices. It enables on-device machine learning inference with low latency and a small binary size. TensorFlow Lite also supports hardware acceleration with the Android Neural Networks API.
  TensorFlow Lite uses many techniques for achieving low latency such as optimizing the kernels for mobile apps, pre-fused activations, and quantized kernels that allow smaller and faster (fixed-point math) models. TensorFlow Lite defines a new model file format, based on FlatBuffers instead of using Protocol buffers. FlatBuffers is an open-sourced, efficient cross platform serialization library.

- **Tfmobile :** Tensorflow Mobile is one of the solutions that were provided for deploying machine learning applications on mobile and embedded devices. TensorFlow was designed to be a good deep learning solution for mobile platforms. It can be very useful for a mobile app to be able to make sense of a camera image. If your users are taking photos, recognizing whats in them can help your camera apps apply appropriate filters, or label the photos so theyre easily findable. Its important for embedded applications too, since you can use image sensors to detect all sorts of interesting conditions, whether its spotting endangered animals in the wild or reporting how late your train is running.

- **Training Accuracy :**The training accuracy shows the percentage of the images used in the current training batch that were labeled with the correct class.

- **Validation accuracy :**The validation accuracy is the precision (percentage of correctly-labelled images) on a randomly-selected group of images from a different set.

- **Weights :** Weights or synaptic weights refer to the strength or amplitude of a connection between two nodes, corresponding in biology to the amount of influence the firing of one neuron has on another. Weights are the learnable parameters of the neural network model. Most machine learning algorithms include some learnable parameters like this. The values of these parameters before learning starts are initialised randomly (this stops them all converging to a single value). Then when presented with data during training, they are adjusted towards values that have correct output.

- **Width Multiplie (α):** alpha represents the Width Multiplier, is responsible for shrinks the number of channels. If the width multiplier is 1, the network starts off with 32 channels and ends up with 1024. The role of the width multiplier is to thin a network uniformly at each layer. (For this project α =0.5 for MobileNet and α =1 for Inception). For a higher value of α , the accuracy is also high.