DON BOSCO COLLEGE OF ENGINEERING

FATORDA, MARGAO, GOA – 403602.

DEPARTMENT OF COMPUTER ENGINEERING

2020 – 2021

SHELLFISH ENCYCLOPEDIA

By

Mr. Dipesh Pritidas Lotliker
Mr. Aniket Anil Naik
Miss. Prachi Rajendra Naik
Mr. Rudresh Anand Naik
Mr. Gauraksh Ramlal Vernekar

Under the Guidance of

Mr. Amey D.S. Kerkar
Assistant Professor

&

Miss. Janhavi Naik
Assistant Professor

BACHELOR OF ENGINEERING: GOA UNIVERSITY

DON BOSCO COLLEGE OF ENGINEERING

FATORDA, MARGAO, GOA- 403602.

2020 – 2021

CERTIFICATE

This is to certify that this dissertation entitled

SHELLFISH ENCYCLOPEDIA

Submitted in partial fulfillment of the requirements for Bachelor's Degree in Computer Engineering of Goa University is the bonafide work of

Mr. Dipesh Pritidas Lotliker(1714027)
Mr. Aniket Anil Naik(1714030)
Miss. Prachi Rajendra Naik(1714033)
Mr. Rudresh Anand Naik(1714034)
Mr. Gauraksh Ramlal Vernekar(1714063)

Mr. Amey D.S. Kerkar	**Dr. Gaurang Patkar**
Assistant Professor	**Head of the Department**

DON BOSCO COLLEGE OF ENGINEERING

FATORDA, MARGAO, GOA- 403602.

2020 – 2021

CERTIFICATE

The dissertation entitled,

SHELLFISH ENCYCLOPEDIA

Submitted by

Mr. Dipesh Pritidas Lotliker
Mr. Aniket Anil Naik
Miss. Prachi Rajendra Naik
Mr. Rudresh Anand Naik
Mr. Gauraksh Ramlal Vernekar

In partial fulfillment of the requirements of the Bachelor's Degree in Computer Engineering of Goa University is evaluated and found satisfactory.

DATE: _____ **EXAMINER 1:** _____

PLACE: _____ **EXAMINER 2:** _____

ABSTRACT

Identifying shellfish can be hard and without the knowledge it is impossible to determine which shellfish it is. One of the main challenges faced in developing detection and identification system for shellfish is there are no sufficient datasets available. The proposed system uses image classification and artificial intelligence technique to identify shellfish species. User can simply take picture of the shellfish on the app and within seconds the app will identify which shellfish it is.

ACKNOWLEDGEMENT

We would like to take this opportunity to express our sincere and heartfelt gratitude to all the stakeholders who have helped us directly or indirectly by providing us with the opportunity, the necessary expertise and guidance, motivating and mentoring us constantly. Without their valuable help this project would have always remained as a dream.

We would like to thank our Director **Rev. Fr. Kinley D' Cruz** and our principal **Dr. Neena S. P. Panandikar** for providing us with the necessary facilities and the much needed motivation throughout the project. We are indebted to you.

We would like to express our gratitude to our Head of the Department, **Dr. Gaurang Patkar** for showing confidence in us and for giving us timely suggestions and advice with his rich experience and expertise.

Guides are like beacon of light showing us the right path in the phase of uncertainty and confusion during the course of the project. When the things seem to be drifting away from the goal, they put us back on the track with their constant monitoring and guidance and expertise. We are indebted to our Internal Guide Prof. Amey Kerkar Asst. Professor, Department of Computer Engineering, and our co-guide Prof. Janhavi Naik Asst. Professor, Department of Computer Engineering.

TABLE OF CONTENTS

LISTOF FIGURES

LISTOF TABLES

CHAPTER 1:
INTRODUCTION

1.1 INTRODUCTION TO PROJECT

Fish is apparently one of the most widely consumed foods on earth today. It is a staple food in many countries, with India ranked fifth in the world for fish consumption. Tripura, Lakshadweep, and Goa have been identified as the most fish-rich areas in India. Fish has been a major source of human protein and nutrients throughout history.

Shellfish are a major part of the fish above. Shellfish is a flexible and delicious source of nutritious food. Clams, Prawns, Oysters and Crabs are just a few of the famous fish. Shellfish are divided into two common categories that include Crustaceans and Mollusks. Crustaceans are easily identified by their strong, fragile bodies. Mollusks are divided into single-shell gastropods like snails; bivalves with interlocking shells like clams; and octopus-like cephalopods have no shells. The mussel variety makes buying fish a bigger challenge than working with meat and poultry. Adding to this challenge is the common problem of poorly labelled fish being sold at a high price in the market. The proposed system will take image from the user, detect if there is any shellfish and if present identify species of that shellfish and will tell user its species along with local name. Also provide additional information like its nutrition facts, health benefits and special dish related to that shellfish.

1.2 PURPOSE OF THE PROJECT

The main purpose of our project is to develop an app that will help user to recognize shellfish by just capturing its image. If an individual is not able to identify the shellfish, he must either ask someone knowledgeable about the shellfish or use the internet to manually compare and try to guess. This can be a problem as often there are mislabelled fishes on sale in the market. Our project aims to guide the user identifying the shellfish from its picture and also provide nutrition content & other useful information.

1.3 PROBLEM DEFINITION

There are various types of shellfishes, out of which there are some which look alike and are difficult to recognize without proper knowledge. The goal of this project is to provide the user with complete information about the shellfish.

1.3.1 EXISTING SYSTEM

Currently there are two major fish detection approaches, one is based on background subtraction and the other uses convolutional neural networks. The background subtraction methods are often adopted for a fixed scene to generate foreground masks, followed by post-processing steps for fish detection and position calculation.

Convolutional neural networks are widely used for image classification by learning invariant features of images on large-scale datasets. CNN for fish detection can achieve high accuracy if sufficient training data are provided. However, the public datasets are mostly out-dated and imbalanced. Thus, the development of learning-based fish detection algorithm which requires only very limited training data will be beneficial.

1.3.2 PROPOSED SYSTEM

The proposed system is an android mobile application which will use a well-trained and tested deep learning model. Activation functions are altered by trial and error to improve accuracy and implementation of dropout layers, to reduce over fitting.
Deep learning model architecture used will be Single Shot Multi-Box Detector (SSD) and will be using a feature extractor known as Mobile-net. The system will be using its smartphone to upload pictures of shellfish.
The image dataset will first undergo pre-processing and will be prepared to pass it onto the classification model. The system will use trained model, take the input image, and give a prediction score to identify the shellfish class.
Once system identifies shellfish it will display its name along with the local name. The system will display to the user all of the nutrition details and the health benefits of the particular shellfish. System will also inform user about its famous dishes.

1.4 SCOPE OF THE PROJECT

Existing systems can successfully detect presence of a fish in an image using the image captured from camera. Image gets processed by the system and predicts the species of the fish. These systems cannot give very accurate result because one of their limitations is that, they are able to predict only when backgrounds in the pictures are similar.

If a picture with fish in a completely different surrounding is given as input the results produced are not accurate. These systems require huge datasets to increase accuracy of the results. Scope of this application is to be able to successfully detect and identify species of the shellfish with the available datasets and resources. Thus, to increase the accuracy of identification and give accurate results.

An app that can be used on-the-go, on mobile devices that identifies and provides important information of shellfishes.

1.5 REPORT ORGANIZATION

The current introductory section provides a brief introduction about each chapter.

Chapter 1: Introduction

This chapter highlights the purpose and scope of the project, furthermore the limitations of the current existing systems and how the proposed system overcomes these limitations.

Chapter 2: Literature Survey

This chapter describes the various technologies and concepts employed in the development of the project.

Chapter 3: System Requirement Specification

This chapter focuses on the specific requirements and aims of the project upon completion.

Chapter 4: System Design

This chapter delineates the software life cycle model used in the development of the product and the reason(s) for opting for the model by stating its advantages and disadvantages.

Chapter 5: Implementation

This chapter deals with the implementation of the project where in the snapshots of each execution steps are shown.

Chapter 6: Testing

This chapter shows the observations and graphs of various algorithms that were used during implementation of the project and comparison of their performance.

Chapter 7: Conclusion

This section deals with the conclusion that is derived after implementing the final system.

CHAPTER 2:

LITERATURE SURVEY

2.1 OBJECT DETECTION METHODS

2.1.1 EXEMPLAR-SVMS

It is a simple yet powerful method, which combines the operation of a discriminatory detector with clear text given the path closest to the neighbours. This approach is based on training the SVM's equitable distribution of all models in a training set. Each of these SVM Examples is described in one positive state and millions of negative ones. While each detector illustrates its example, we see that the combination of Exemplar-SVM offers incredibly good performance. The central advantage of our approach is that it creates a clear connection between each acquisition and a single training model. Because most acquisitions show good alignment with their corresponding model, it is possible to transfer any exemplar meta-data (fragmentation, geometric structure, 3D model, etc.) directly to the acquisition, which can be used as part of a general scenario understanding. [1]

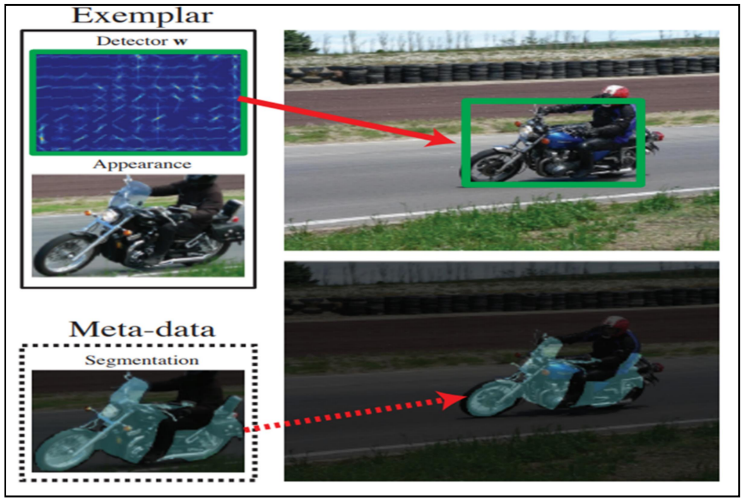


Fig 2.1: Working of Exemplar SVMs

The method is based on training the distinctive differentiation of each model and shows that standard performance is possible from one positive model with millions of disadvantages.

2.1.2 DEFORMABLE PART MODEL

This program represents variables that use a combination of high-level partial models. These models are trained using a discriminatory process that requires only the binding boxes on the set of images. [2]

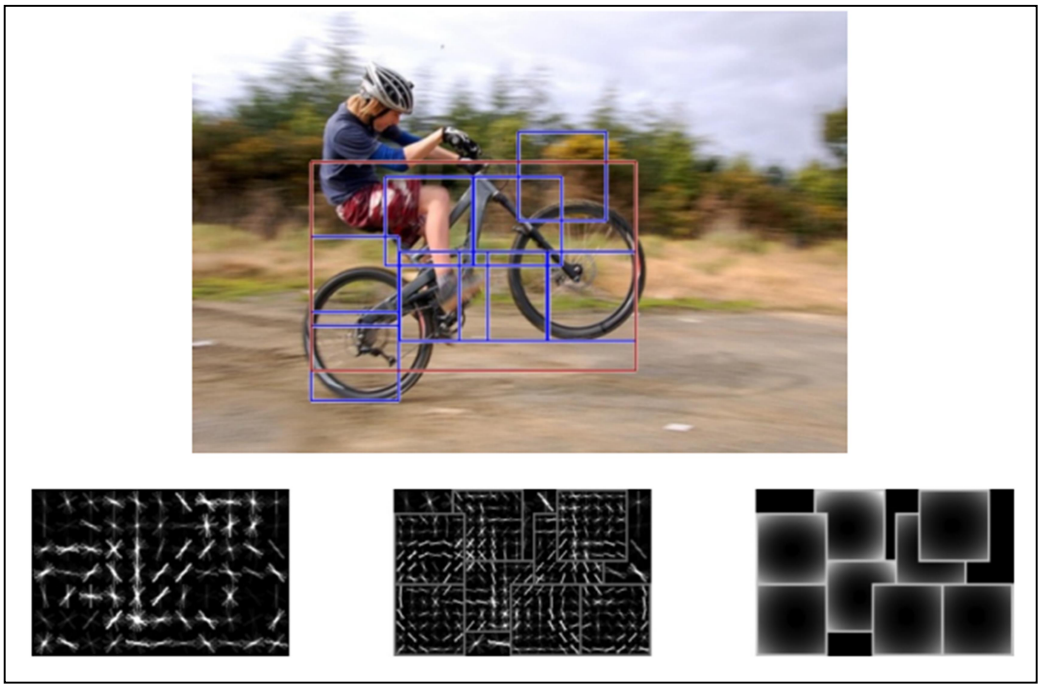


Fig 2.2: Working of Deformable Part Model

The method builds on the framework of image formats. Symbol structures represent objects by the collection of parts arranged in a disabling manner. Each component captures the visual appearance properties of the object while the deformed suspension is reflected by a spring-like connection between certain pairs of components.

2.1.3 HISTOGRAM OF SPARSE CODES

Creating sparse codes with dictionaries read from data using SVD, and compiling pixel sparse sparse codes to create local histograms. Deliberately stay true to the

sliding windows frame and only change the features below. To keep the training running smoothly, we use size reduction using SVD in learned models, and adopt supervised training where hidden root positions and components are provided externally e.g., from a HOG-based detector. By reading and using local presentations that are more specific than gradients. [3]

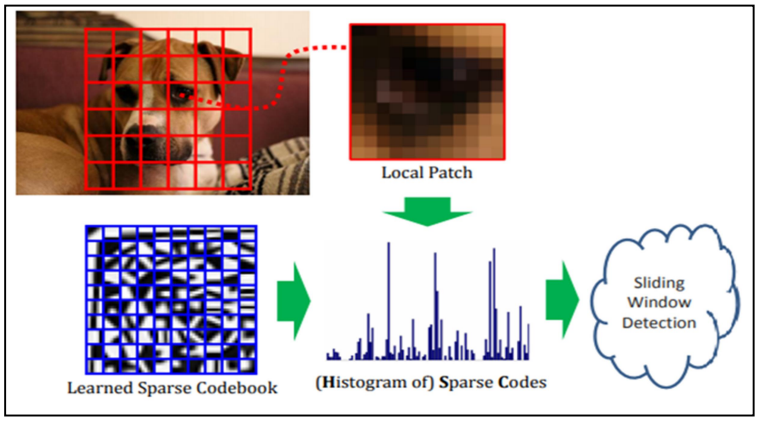


Fig 2.3: Working of Histograms of Sparse Codes

2.1.4 HISTOGRAM OF ORIENTED GRADIENT

The Histogram of Oriented Gradients (HOG) is a widely accepted feature of object acquisition, which provides a logical trade between finding accuracy and difficulty compared to other rich features, a multi-object detector using Historical Oriented Gradient (HOG) features and support Vector Machine Separation (SVM). The same detectors with a moderate work load are used to enable the processing of multiple scales. [4]

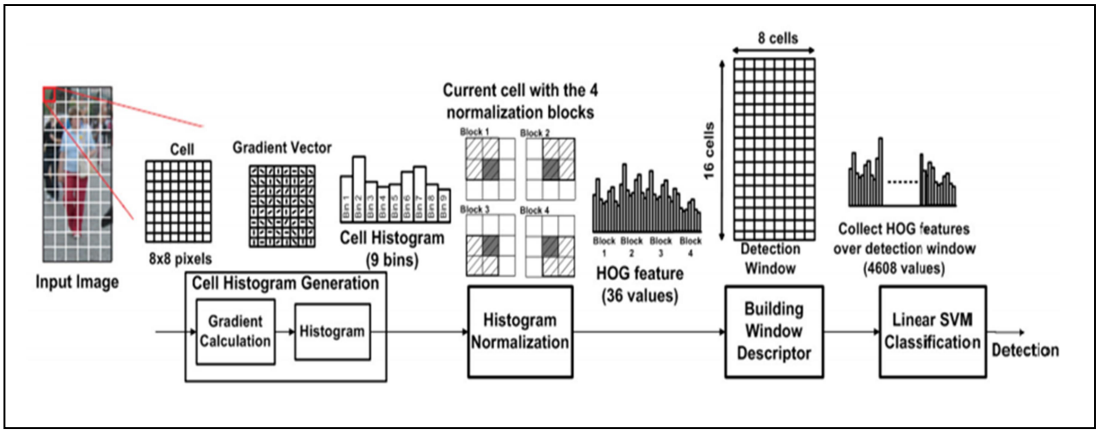


Fig 2.4: Working of Histogram of Oriented Gradient

The image is divided into non-overlapping 8 x 8 pixels patches called cells, where gradients are calculated for each pixel. A gradient-shaped histogram of 9 barrels is generated in each cell. The histogram is then organized by its neighbouring cells to increase texture strength and lighting variability. In each cell, standardization was performed in 2 x 2 cell blocks that resulted in the final 36 elements. Linear SVM classifiers are often used to find an object in combination with HOG features.

2.1.5 REGIONS WITH CONVOLUTIONAL NEURAL NETWORK

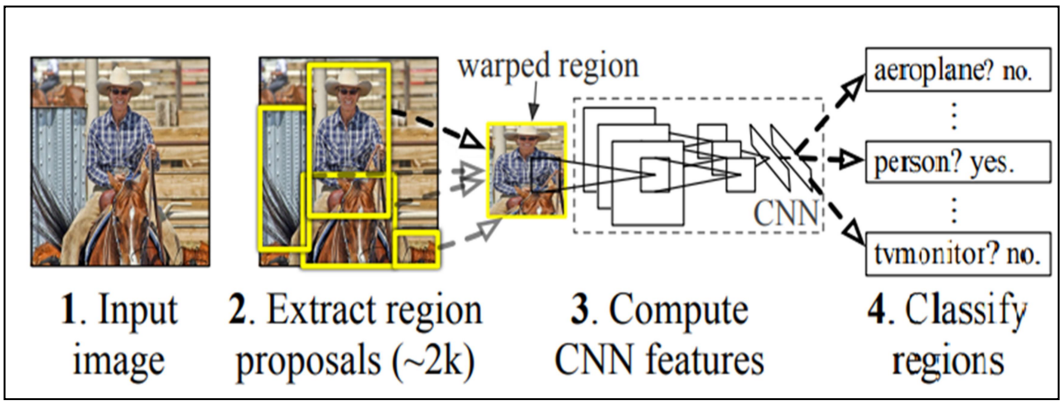


Fig 2.5: Working of R-CNN

The system captures the input image, extracts around 2000 subdivisions from top to bottom, integrates the features of each proposal using a large convolutional neural network (CNN), and separates each region using specific phase SVMs. [5]

2.1.6 SPATIAL PYRAMID POOLING

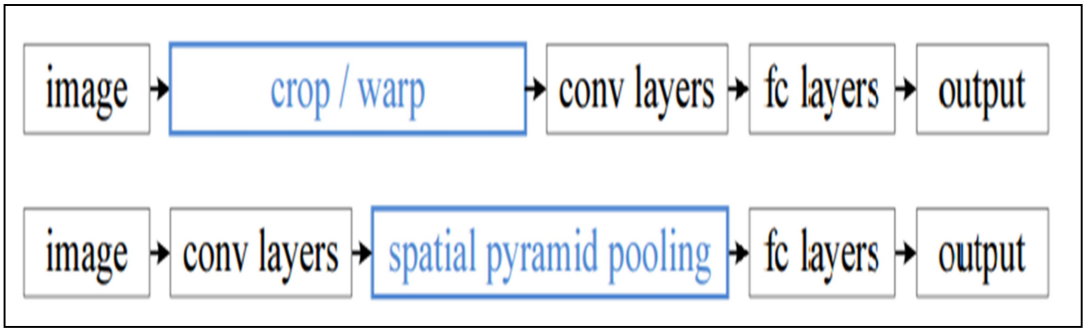


Fig 2.6: Working of SPP

Spatial pyramid pooling (SPP) layer is used to remove a fixed network size limit. Specifically, we include the SPP layer above the final layer of specification. The SPP layer sets the pools in the elements and produces consistent results, which are then applied to the fully connected layers (or other sections). It divides an image into categories from the most beautiful to the largest, and combines local features into them. [6]

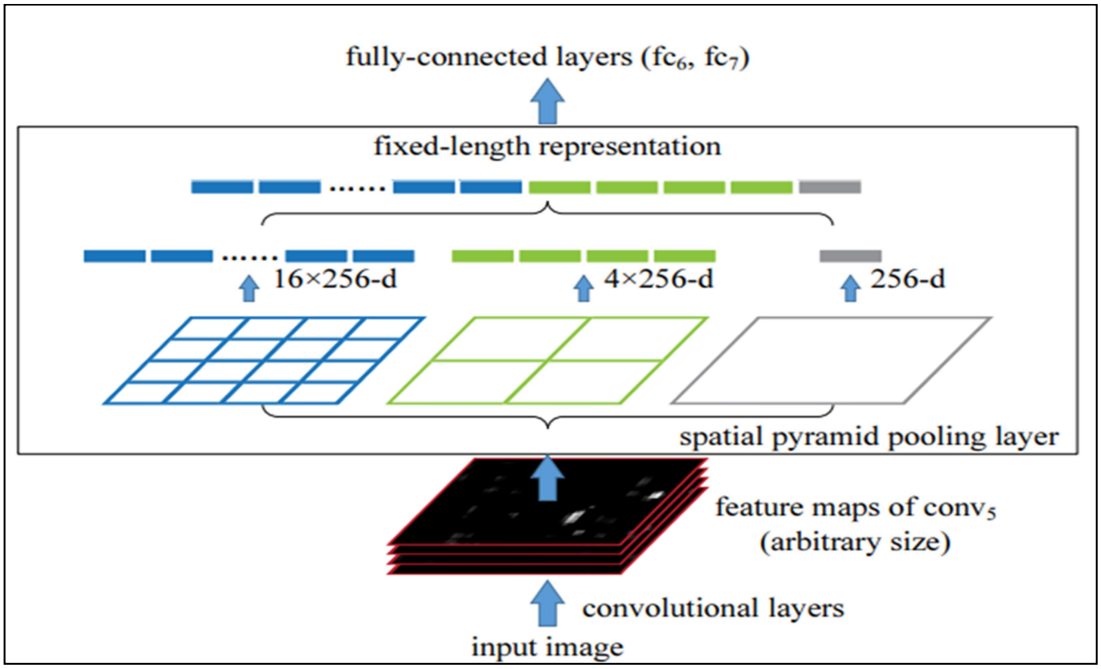


Fig 2.7: A network structure with a spatial pyramid pooling.

The SPP-net layout can produce a consistent representation of any length regardless of image size / scale. We change the last layer of integration (e.g., pool5, after the last convolutional layer) by the location of the integration pyramid. The integration of the pyramid is also strong in counteracting disability. With these benefits, SPP-net should generally improve all image classification methods designed for CNN.

2.1.7 ACCURACY COMPARISONS

Sr. No	TITLE	METHOD	ACCURACY (mAP)
[1]	Ensemble of Exemplar-SVMs for Object Detection and Beyond	E-SVM	22.7%
[2]	Discriminatively trained deformed part models Version5	DPM	33.4%
[3]	Histograms of Sparse Codes for Object Detection	HSC	35.2%
[4]	Energy-Efficient HOG-based Object Detection at 1080HD 60 fps with Multi-Scale Support	HOG	37.9%
[5]	Rich feature hierarchies for accurate object detection and semantic segmentation	R-CNN	58.5%

Table 2.1: Comparison of object detection methods

2.2 CONVOLUTIONAL NEURAL NETWORK (CNN)

The only significant difference between CNN and traditional ANNs is that CNNs are widely used in the field of pattern recognition within images. This allows us to incorporate image-related features into architecture, making the network more suitable for image-based functions - while continually reducing the parameters needed to set the model. One of the major limitations of traditional ANN models is that they often struggle with the computer complexity required to process image data.

CNN contains three types of layers. These are convolutional layers, integration layers and fully integrated layers. When these layers are retained, the construction of CNN is built. [7]

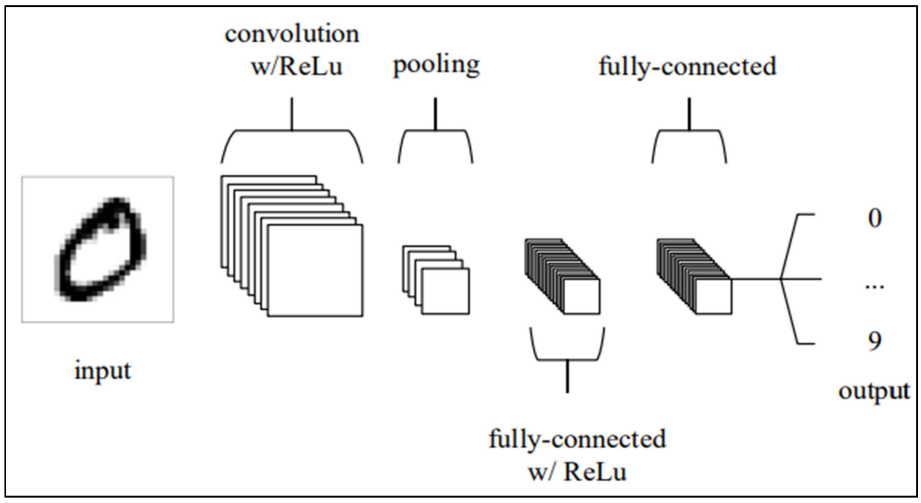


Fig 2.8: Layers in CNN

1. As with other ANN models, the **input layer** will capture pixel values of the image.

2. The **convolutional layer** will determine the release of neurons connected to the local input regions by calculating the scalar product between their weights and the region connected to the input volume. The targeted line unit (usually accessed by ReLu) aims to use the element 'activation function' as a sigmoid in the extraction used by the previous layer.

3. The **pooling layer** will then simply perform down sampling along the spatial dimensionality of the given input, further reducing the number of parameters within that activation.

4. The **fully connected layers** will perform the same functions found in standard ANNs and attempt to generate category scores from activation, which will be used for segmentation. It is also suggested that ReLu can be used between these layers, to improve performance.

Prior to the popularity of in-depth reading in computer vision, object acquisition was performed using manual machine learning features such as scale invariant feature transform (SHIFT), histogram of targeted gradients (HOG). At that time, the best acquisition models received about 35% mean intermediate (mAP) in the PASCAL VOC Challenge database. However, between 2010-2012, the progress of the acquisition was not static. The best performing models get their results by building integration systems and applying fewer modifications to successful models.

CNN-based acquisition models can be divided into two distinct categories [8]:

2.2.1 TWO-STAGE APPROCH

In the two-stage object detection, the first stage generates region or object proposals, and in the second stage, those proposals are classified and detected using bounding boxes.

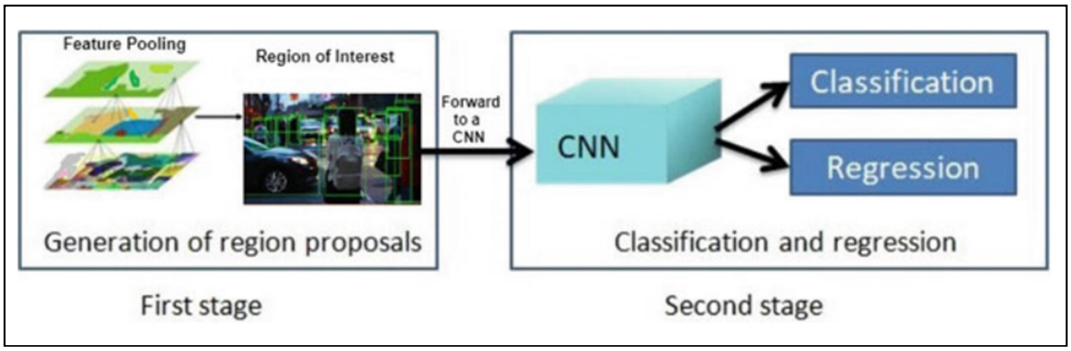


Fig 2.9: Working of Two stage CNN

Ex: R-CNN, SPP-net, Fast R-CNN, Faster R-CNN, FPN & Mask R-CNN.

2.2.2 ONE-STAGE APPROCH

In the one-stage approach, classification and regression is done in a single shot using regular and dense sampling with respect to locations, scales, and aspect ratio.

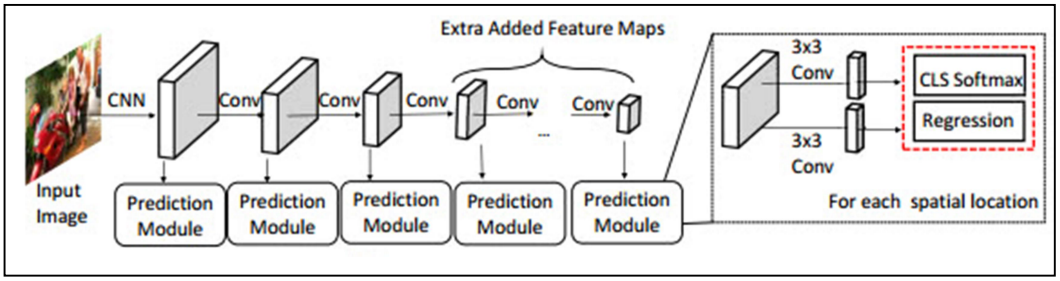


Fig 2.10: Working of One stage CNN

Ex: YOLO & SSD.

One-stage approach methods are much faster and also accurate.

Method	mAP (%)	FPS
Faster R-CNN (VGG16)	73.2	7
Faster R-CNN (ZF)	62.1	17
YOLO	63.4	45
SSD300	74.6	46
SSD500	76.8	119

[9]Table 2.2: Comparisons of CNN architectures

As per the results in the research, it shows SSD came out to be the most accurate standard model.

2.3 SINGLE SHOT MULTI-BOX DETECTOR (SSD)

SSD has a single integrated network that simultaneously predicts binding boxes and box labels with confidence values. This is inspired by multi-box which is a method used to predict an agnostic-designed box quickly. This removes computational overhead from the most widely used suggestion networks in the region or other regional search methods; therefore, it is proven very quickly with a slight decrease in accuracy.

Single Shot: this means object localisation and classification are done in single forward pass. **Multiple Box**: This is the name of the box binding process. **Detector**: A network is a detector that separates those detected objects.

The SSD has two components: the spine model and the SSD head. The Backbone model is usually an image separation network that is pre-trained as a feature output. The SSD head is one or more solution layers added to this backbone and the results are interpreted as binding boxes and object segments in the area with the final layer functionality.

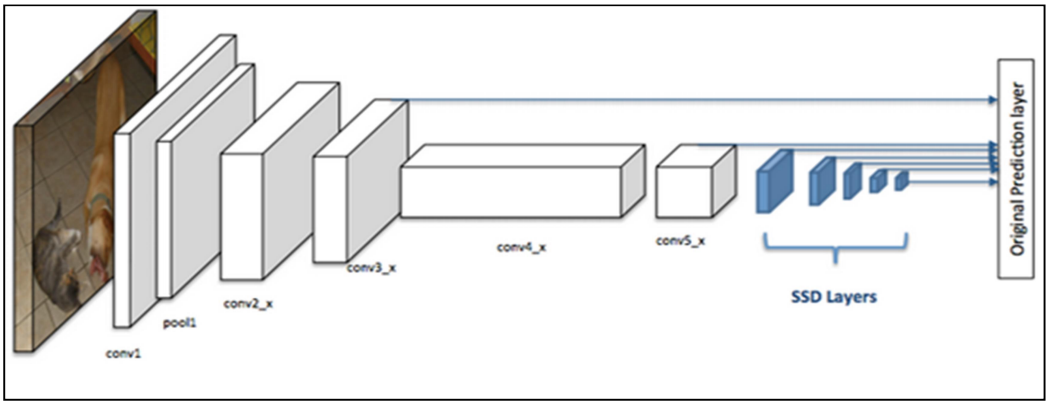


Fig 2.11: Architecture of SSD

SSD uses automatic boxes for various scales, shapes, and aspect ratios for different output layers.

It uses 8000+ boxes to get the best coverage, measurement, and position ratings. Most of the predictions will not contain anything. SSD drops forecasts with confidence points below 0.01. We then use Non-Max (NMS) intervals of 0.45 intervals per class and keep a maximum of 200 directions per image.

Important points to remember about SSD:

- While Yolo has a limited amount of cell grid feature. The SSD uses a wide range of contrast for most boxes with better accuracy.
- SSD has additional conv layers at the end of the backbone for object detection. Convolutional layer has multiple features with different scale and hence it is able to detect objects in multiple scales better.

2.3.1 SSD FRAMEWORK

SSD The SSD only needs a photo insert and ground truth boxes for each item during training (a). In a flexible way, we test a small set of default boxes for different dimensions in each area on several feature maps with different scales (e.g., 8 × 8 and 4 × 4 in (b) and (c)).

In each default box, we estimate both the stability and confidence of all categories of objects ((c1, c2, • • •, cp)). During training, we begin by comparing these default

boxes to the ground boxes.

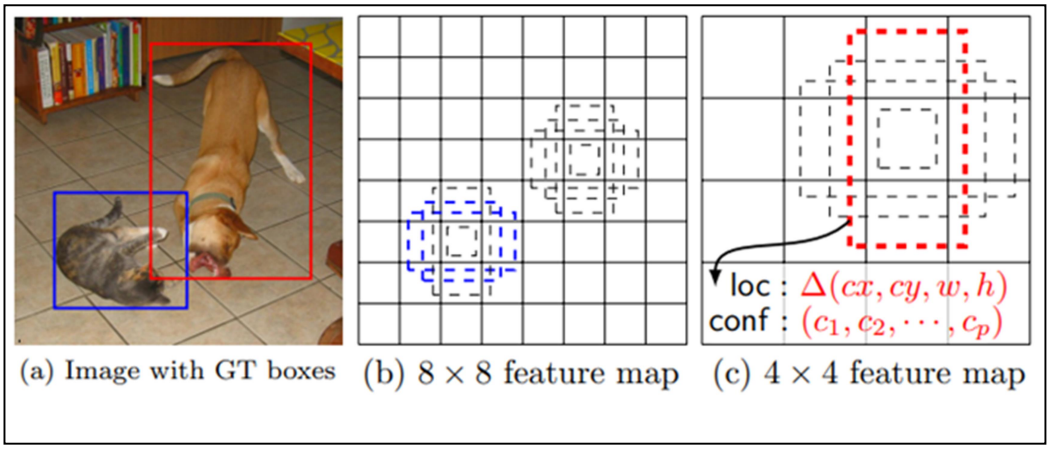


Fig 2.12: Feature maps in SSD

For example, we have paired two automatic boxes with a cat and one with a dog, which are treated as good and everything else as bad. Model loss is a significant amount between local performance loss and loss of confidence (e.g., SoftMax).

2.3.2 CONCEPTS AND PARAMETERS OF SSD

2.3.2.1 GRID CELL

Instead of using slide windows, the SSD separates the image using a grid and each grid cell is responsible for finding objects in that image region. Finding things simply means predicting the stage and location of an object in that region. If nothing else, we view it as a backend and a neglected area. For example, we can use a 4x4 grid in the example below. Each cell grid is able to extract the position and shape of the object it contains.

2.3.2.2 ANCHOR BOX

If there are multiple objects in one grid cell or we need to detect multiple objects of different shapes. There is where anchor box and receptive field come into play.

Each grid cell in SSD can be assigned with multiple anchor/prior boxes. These anchor boxes are pre-defined and each one is responsible for a size and shape within a grid cell.

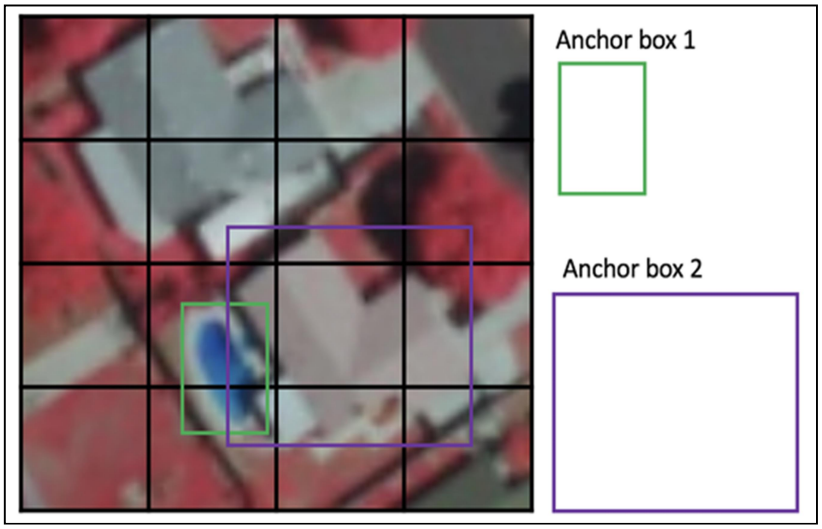


Fig 2.13: Anchor boxes in SSD

2.3.2.3 ASPECT RATIO

Not all objects are square in shape. Some are long and some are wide, to varying degrees. SSD design allows pre-defined measurement of anchor boxes to respond to this. The ratio parameter can be used to determine the different dimensions of the configuration boxes that meet each cell grid at each zoom / scale level.

2.3.2.4 ZOOM LEVEL

Contract box boxes do not need to be the same size as a grid cell. We may be interested in finding small or large objects within a grid cell. The zoom parameter is used to determine how many stop boxes need to be raised up or down in relation to each cell grid. As we have seen in the example of the anchor box, the size of a building is usually larger than a swimming pool.

2.3.2.5 RECEPTIVE FIELD

It is defined as a region in the input field that is considered a specific feature of CNN. Due to the function of convolution, the features of different layers represent different regional sizes in the input image. As it gets deeper, the size represented by the feature increases.

In the example below, we start with the bottom layer (5x5) and apply a convolution that leads to the middle layer (3x3) where one element (green pixel) represents the 3x3 region of the input layer (bottom layer). Then apply convolution to the middle layer and find the top layer (2x2) where each element corresponds to the 7x7 region in the installation image. This type of green and orange 2D layout is also called feature maps referring to a set of features built using the same pointer feature in different areas of the input map in the form of a moving window. Features on the same feature map have the same reception field and look at the same pattern but in different locations. This creates ConvNet local diversity.

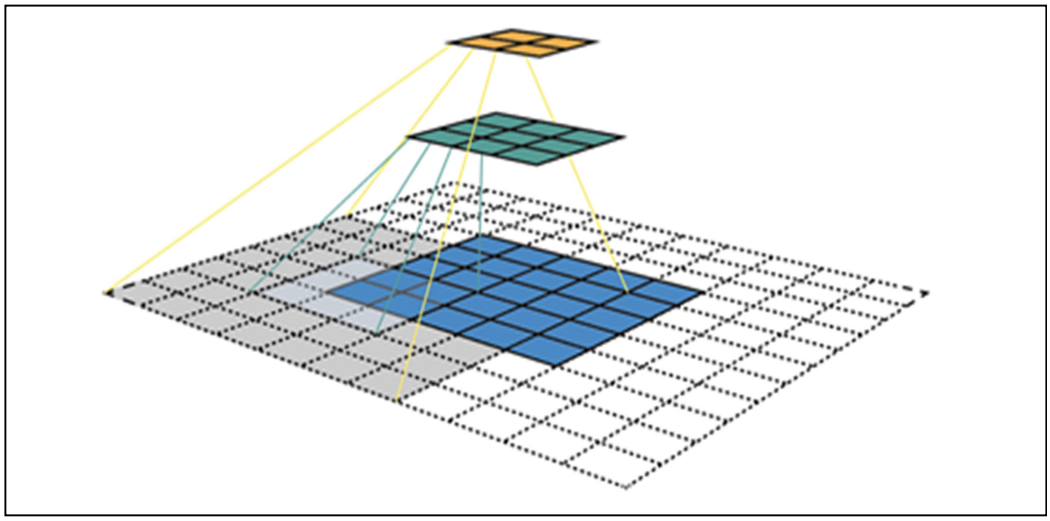


Fig 2.14: Receptive Field in SSD

The reception field is the centrepiece of SSD design as it helps us to get things in different sizes and produce a solid box. Because of this, the SSD allows us to define the successive phase of grid cells in different layers.

For example, we can use a 4x4 grid to find small objects, a 2x2 grid to find objects in the middle and a 1x1 grid to find objects that cover the whole picture.

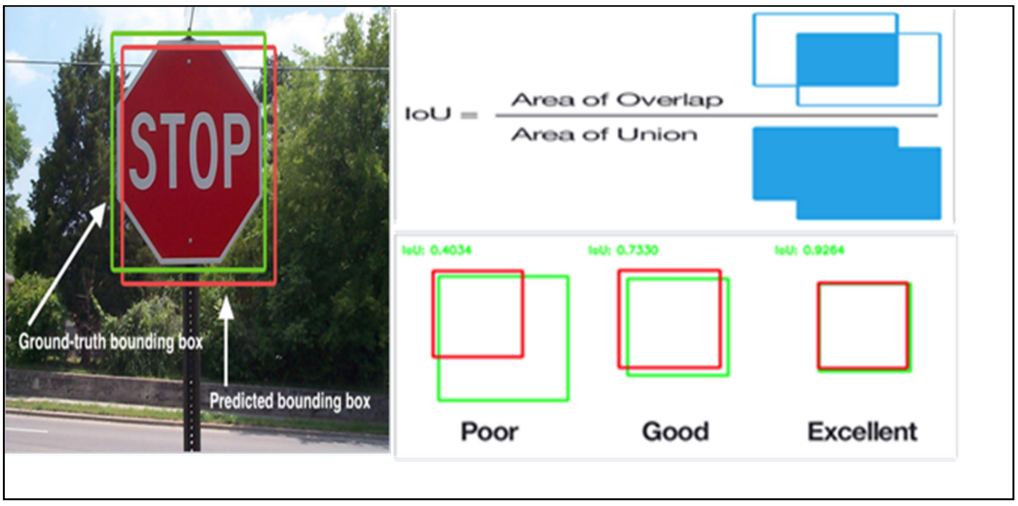


Fig 2.15: Predicted boxes in SSD

2.4 MOBILE NETS

Mobile net is a neural network that is used for classification and recognition whereas the SSD is a framework that is used to realize the multibox detector. Only the combination of both can do object detection. Visual recognition for on device and embedded applications poses many challenges — models must run quickly with high accuracy in a resource-constrained environment making use of limited computation, power, and space.

MobileNet, a family of mobile-first computer vision models for TensorFlow, designed to effectively maximize accuracy while being mindful of the restricted resources for an on-device or embedded application.

MobileNet are small, low-latency, low-power models parameterized to meet the resource constraints of a variety of use cases. It uses **depth wise separable convolutions** to build light-weight deep neural networks.

2.4.1 DEPTH WISE SEPARABLE CONVOLUTION

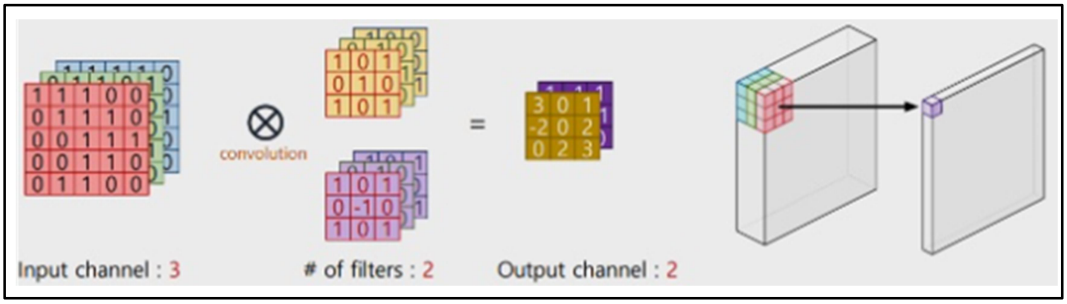


Fig 2.16: Regular and Depth wise separable convolutions

A regular convolutional layer applies a convolution kernel (or “filter”) to all of the channels of the input image. It slides this kernel across the image and at each step performs a weighted sum of the input pixels covered by the kernel across all input channels. If an image has 3 input channels, then running a single convolution kernel across this image will result in an output image with only 1 channel per pixel.

MobileNet's architecture also uses this standard convolution, but just once as the very first layer. All other layers do “depth wise separable” convolution instead. This is actually a combination of two different convolution operations: a **depth wise convolution** and a **point wise convolution**.

Depthwise Convolution:

Make it confusing for each channel separately. With a 3-channel image, deep convolution creates an 3-channel output image. Each channel gets its own set of weights. The purpose of the depth agreement is to filter input channels. Consider edge detection, color filtering, and so on.

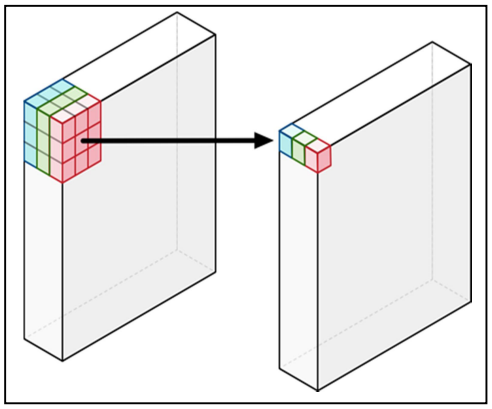


Fig 2.17: Depthwise Convolution

Pointwise Convolution:

Depthwise convolution is followed by pointwise convolution. This is similar to a normal merger but with a 1 × 1 kernel, this simply adds up all the channels (such as the total weight). In a standard agreement, it usually combines most of these dotted characters to create a multi-channel output image. The purpose of this absurd decision is to integrate deep solution delivery channels to create new features.

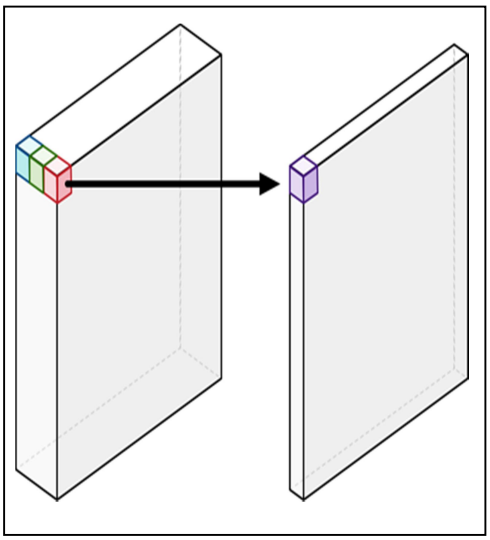


Fig 2.18: Pointwise Convolution

When we put these two things together a depth wise convolution followed by a pointwise convolution the result is called a depth wise separable convolution. A regular convolution does both filtering and combining in a single go, but with a depth wise separable convolution these two operations are done as separate steps.

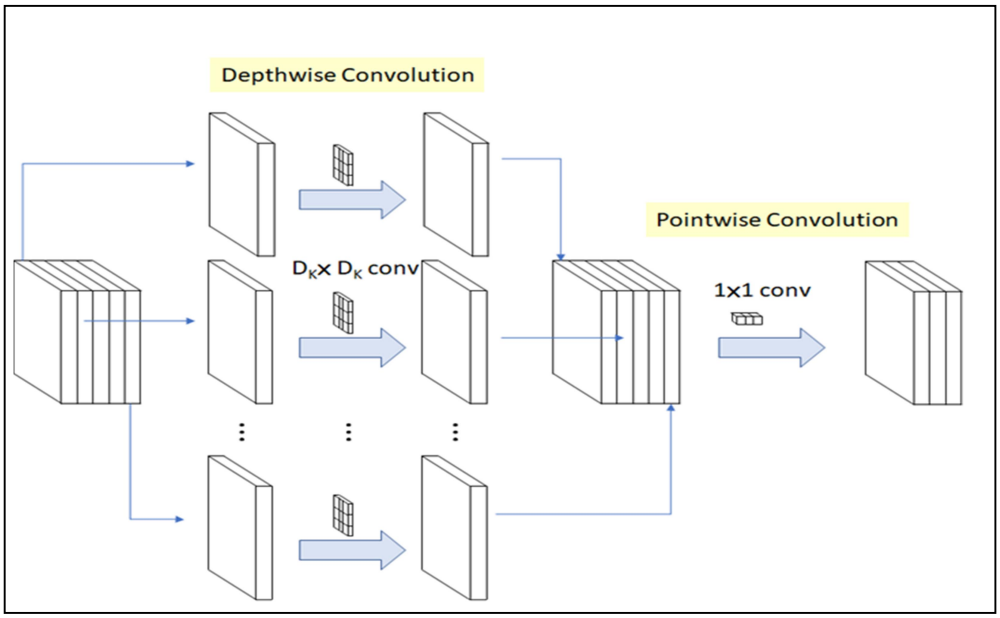


Fig 2.19: Convolution in MobileNet.

The end result of both methods is exactly the same as they both filter the data and create new features. But a typical convolution has to do a lot of math work to get there and you need to learn a lot of instruments. So even if it does (almost or less) the same thing, a depth wise convolution will be much faster!

- Standard convolutions have the computational cost of
  - $D_K \times D_K \times M \times N \times D_F \times D_F$
- Depthwise separable convolutions cost
  - $D_K \times D_K \times M \times D_F \times D_F + M \times N \times D_F \times D_F$
- Reduction in computations
  - $1 / N + 1 / D_K^2$
  - If we use 3x3 depthwise separable convolutions, we get between 8 to 9 times less computations

D_K : width/height of filters
D_F : width/height of feature maps
M : number of input channels
N : number of output channels(number of filters)

Model	Image-Net Accuracy	Million Parameters
Mobile-Net-224	70.6%	4.2
Google-Net	69.8%	6.8
VGG1116	71.5%	138

Table 2.3: Feature extractor comparisons. [10]

2.5 DATA AUGMENTATION

First, we collect the images manually afterwards we perform filtering and remove the unwanted images from Dataset, since convolutional networks require huge amount of data to yield better accuracy we apply augmentation for the Dataset like rotations, zooms and flips randomly.

Suxia Cui proposed [11] a CNN model with image segmentation of fish catching in shallow sea water. A specific set of data was developed to support this study. The data-adding scheme was adopted to acquire more learning resources because the original images in a particular area were not sufficient for the purpose of training, it also helped them to reduce training losses and increase accuracy.

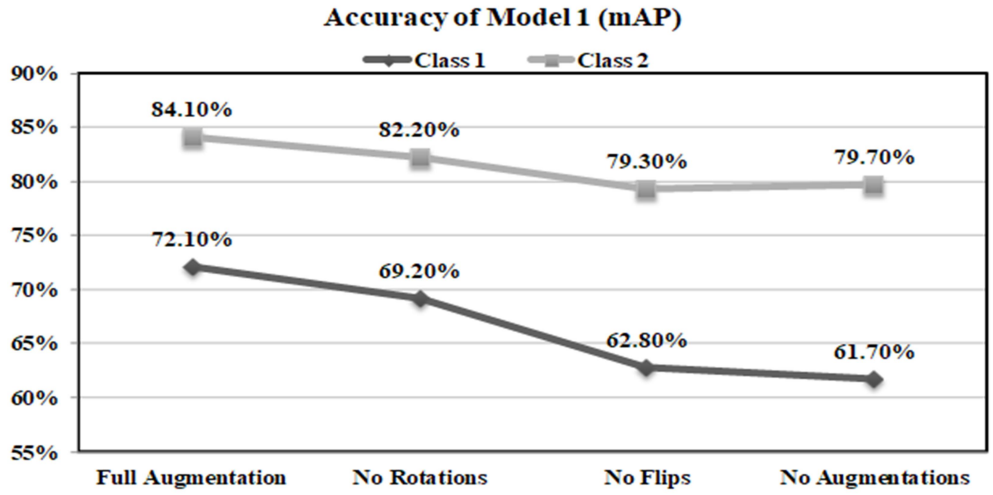


Fig 2.20: Results of Augmentations. [12]

Pushyami Kaveti and Hanumant Singh proposed system [12] using data augmentation which greatly helped in boosting the accuracy of detection. They included a horizontal flip, a vertical flip, and a rotation by 90 degrees randomly on a certain number of the images and added them to the training data. Also showed that increasing the batch size helps in getting a smoother loss curve and made a significant improvement in the localization loss component.

CHAPTER 3:

SOFTWARE REQUIREMENT SPECIFICATION

3.1 INTRODUCTION

3.1.1 PURPOSE

Purpose of this application is to be able to successfully detect and identify species of the shellfish with the available datasets and resources. To increase the accuracy of identification and produce accurate results. An application that can be used on mobile devices that identifies and provides important information of shellfishes.

3.1.2 SCOPE

It will classify two shellfish species clams and mussels. System will only work if storage permissions are provided. Application will only work on android 7.0 or above devices. Application will work efficiently for more than 4GB RAM. System will work only when there is active internet connection. If the mobile device does not satisfy any android specifications or if the permissions are not granted then, the system might not work properly. Sometimes if images are not clear enough then the output might not be accurate.

3.2 THE OVERALL DESCRIPTION

3.2.1 PRODUCT PERSPECTIVE

This software is designed to identify local shellfish by providing user a useful tool that will assist them and give information about species. It will help user understand nutrition details, health benefits and famous Goan dishes. It will also help user to identify and reduce the mislabelled fish sale in market.

3.2.2 PRODUCT FUNCTION

The system is an android app that will be used on a smartphone which will use its camera to capture a live shellfish at any point of time. The image is then being sent to a server that has trained deep learning model architecture. It performs classification and prediction to classify and provide shellfish name, nutrition details, health benefits and dishes to the user.

3.3 EXTERNAL INTERFACE REQUIREMENT

3.3.1 USER INTERFACE

- Frontend Software: Android Studio 4.1.2.
- Backend Software: Django, TensorFlow, Keras, NumPy, OpenCV.

3.3.2 HARDWARE INTERFACE

The hardware requirements are:

- Android Smartphone with minimum 7.0 Android Version and 4GB minimum RAM.

3.3.3 SOFTWARE INTERFACE

The software interfaces required for the project are:

- Windows OS.
- Jupyter Notebook.
- Python.
- LabelImg.
- Android OS 7.0 and above.
- Java
- Kotlin.

3.3.4 COMMUNICATION INTERFACE

The interaction between the user and the application is using the Developed Mobile App where the user can upload the image of the shellfish using the smartphone and the details of the shellfish will be provided.

3.4 SYSTEM FEATURES

3.4.1 SHELFISH CLASSIFICATION

A mobile application working on any smartphone will ask the user for an image which can be from the camera-roll or device storage, it will use the image and implement classification on it and identify if it is a Clam or Mussel. Results of identification will be displayed helping the user to identify the shellfish.

3.4.2 SHELFISH INFORMATION

Once system has classified the shellfish user can then use the same application and find out more details about the shellfish like nutrition, dishes, and health benefits.

3.5 OTHER NONFUNCTIONAL REQUIREMENTS

3.5.1 ACCURACY

Accuracy of the system is very important; our system should try and obtain an accuracy of 100% to provide user with correct and reliable information.

3.5.2 EFFICIENCY

Since deep learning systems have high hardware and software requirements and our systems has to work smoothly on entry-level smartphones for it to be effective. Therefore, achieving an efficient application that can deliver high performance on limited system is important.

CHAPTER 4:

DESIGN

4.1 SOFTWARE DEVELOPMENT MODEL

Iterative Waterfall Life Software Development Model

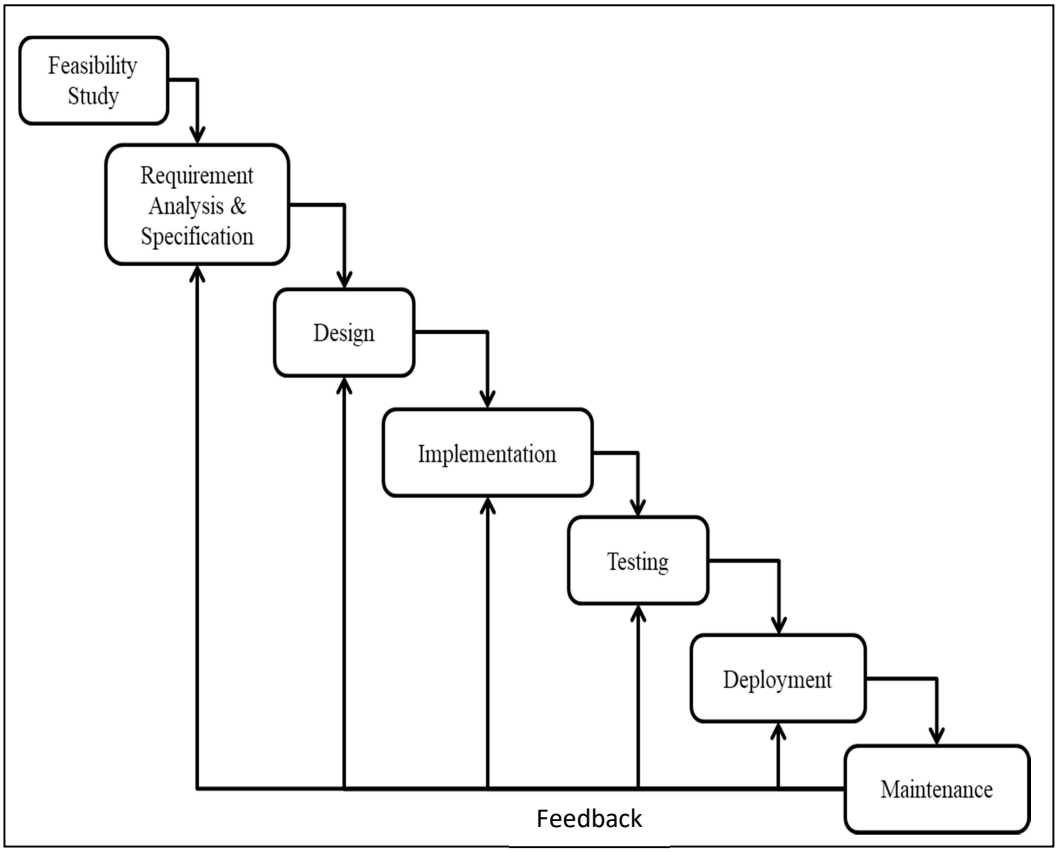


Fig 4.1: Iterative Waterfall Life Software Development Model

Iterative Waterfall provides a steady forward flow in the project. It is similar to waterfall model but there is addition of feedback paths which improve performance. This will allow us to perform updation of weights and biases in the model according the prediction scores. These feedback paths will allow us to go back on previous stage

and do modifications. Since we need to perform lots of training and testing, perform modifications to improve accuracy these feedback paths will be useful.

4.2 SYSTEM DESIGN

4.2.1 OVERALL SYSTEM BLOCK DIAGRAM

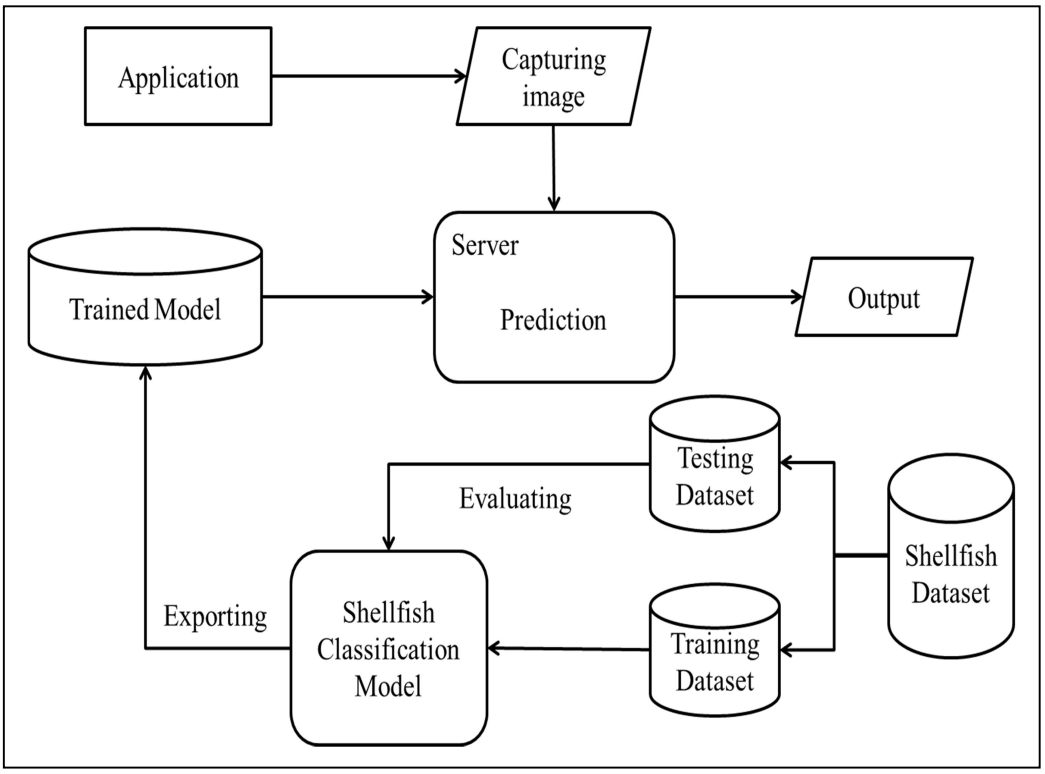


Fig 4.2: Overall System Block Diagram

System takes the shellfish data that is partitioned into testing and training, and using the training dataset system trains the shellfish classification model. Once training is finished system is then verified for its accuracy using testing dataset. After successfully evaluating the shellfish classification model the trained model is then passed onto the server.

The Application contains the front-end part of the system. It displays various functions like capture image, display other information etc. Once the user has captured the image, the image is sent to server where the main part that is prediction takes place. Here, the captured image is evaluated according to shellfish classification model and the result is predicted.

Once the result is predicted, this result is sent back to the server from where the output is displayed. The output is displayed on the application along with some additional information.

4.3 DETAIL DESIGN

4.3.1 PRE-PROCESSING

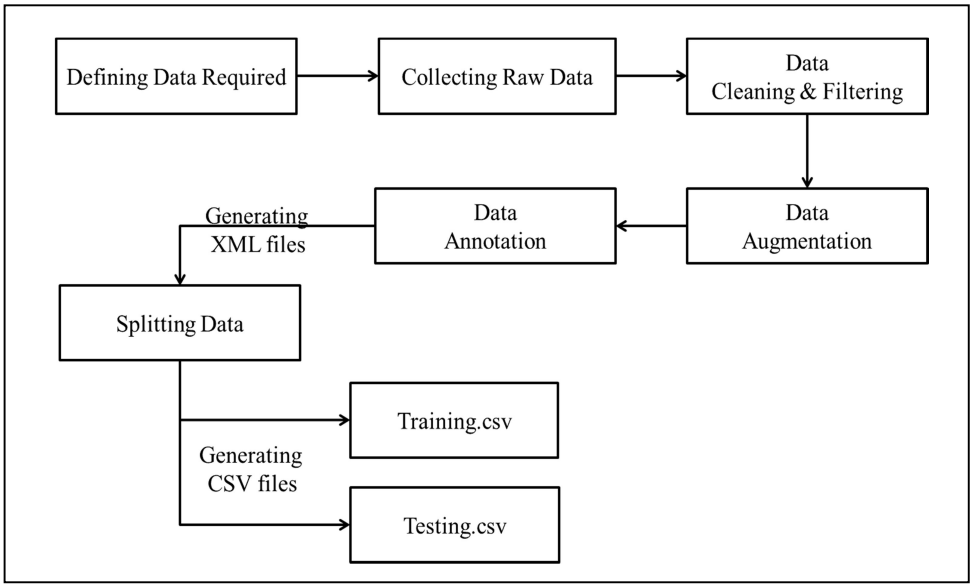


Fig 4.3: Block Diagram for Pre-processing

In pre-processing initially, we define the data that is required for processing. Then images are collected from websites and also manually from real world. These images are cleaned and filtered, images that are blur or do not have shellfish are deleted.

After filtering images are augmented that is rotation and flips are done to each image. In Data Annotation each image is labelled using LabelImg and XML files are generated for each image. After annotation XML files and images are split into test and train dataset and the .CSV files are generated for the test and train dataset.

4.3.2 SHELLFISH CLASSIFICATION MODEL

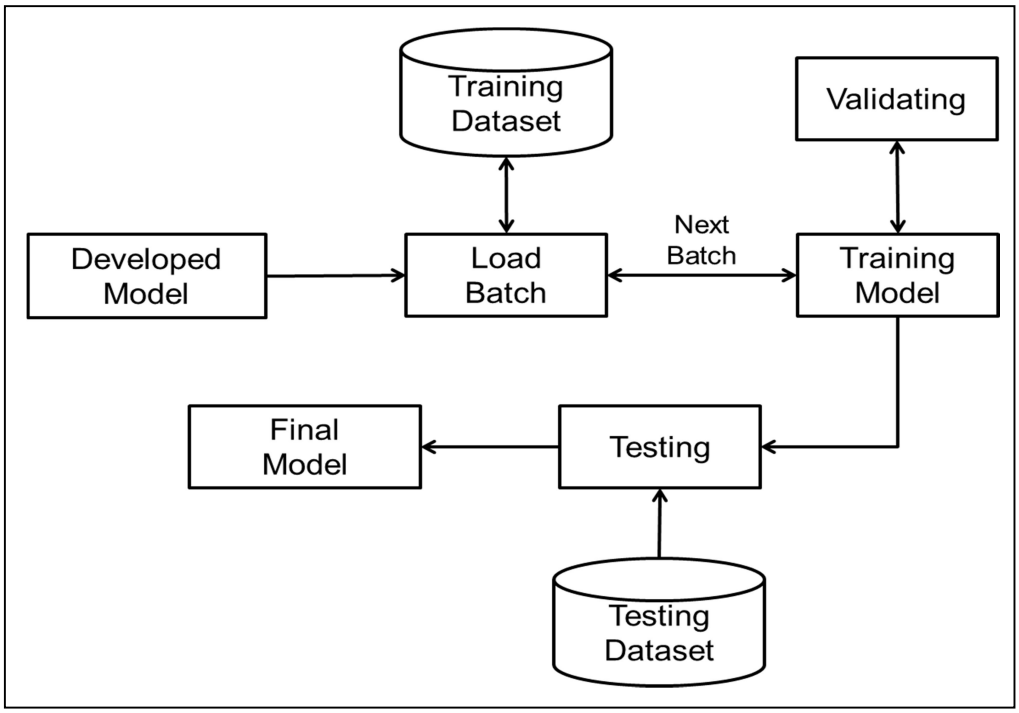


Fig 4.4: Block Diagram for Shellfish Classification Model

Architecture of shellfish classification model is first developed, using single shot multibox detector (SSD) to predict shellfish species. Feature extractor of the model is optimized to enhance efficiency of the classification process. Developed model architecture is then used for training using the training dataset. Training dataset contains over 8000 images which are loaded in batches into the model for training.

Training is performed with each batch of images one by one, once model is trained with a batch is then validated and its weights and biases are updated to improve accuracy. This is continued for all the batches and model is trained properly. Once training is done, we use testing dataset to test efficiency and accuracy of the system. If system after testing produces required results, then the final shellfish classification model is employed.

4.3.3 PREDICTION

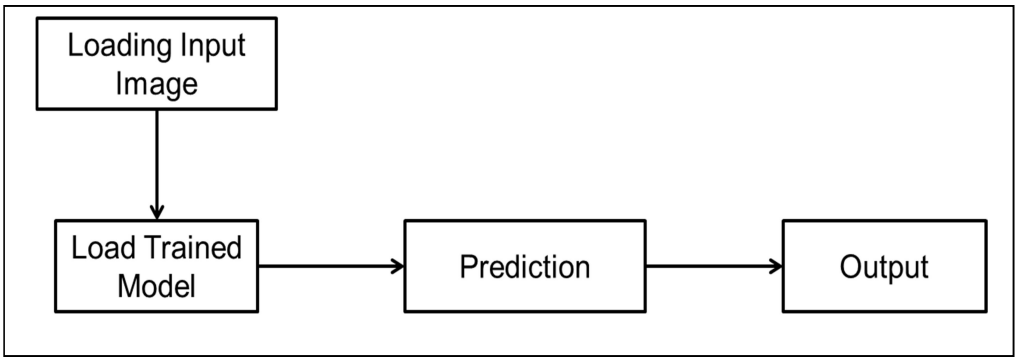


Fig 4.5: Block Diagram for Prediction module

We load the image that has arrived as input. Then we load the trained model and give the image as input to it. Next prediction of the object class in the image is been done. Finally, we pass the predictions to the output module.

4.3.4 DATA FLOW DIAGRAM

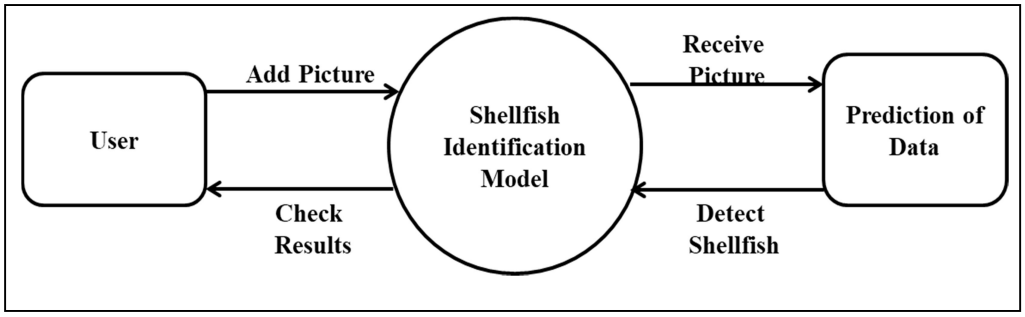


Fig 4.6: Level 0 Data Flow Diagram

In the DFD of our proposed system, the user captures the picture and sends to the shellfish identification model. The model, once the picture is received, predicts the result of the data. The result is sent from the prediction model to shellfish identification model to verify the results. Once, the results are verified the user receives the results and is displayed on the system. Also, some additional information is also displayed with respect to the result.

4.3.5 USE CASE DIAGRAM

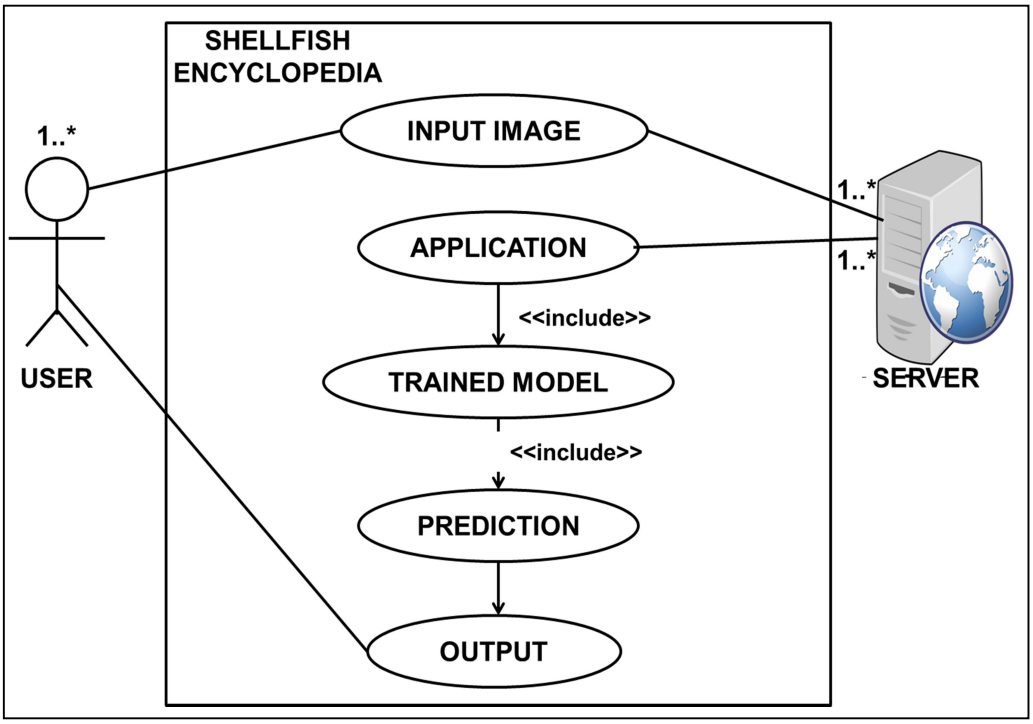


Fig 4.7: Use case diagram of the System

The purpose of the case diagram to use is to take a powerful feature of the system. Use case diagrams used to collect requirements, especially design requirements, for the system including internal and external influences.

As can be seen in the Use Case diagram above, the User first captures and sends the image to the server. The server will contain all the main models and processing units of our proposed system. The image is then provided with a professional model that predicts the shellfish category and the release of the required information about the target shellfish shown to the user.

CHAPTER 5:

IMPLEMENTATION

5.1 OVERVIEW OF THE TECHNOLOGIES USED

5.1.1 TENSORFLOW

It is the platform widely used for machine learning and deep learning. It makes use of python which provides a front-end API for creation of applications.

5.1.2 KERAS

It is a high-density python-based network SDK and can be run on TensorFlow. It enables prototyping rapidly and efficiently and embraces all convolutionary networks and repetitive networks and variations of both.

5.1.3 OPENCV

A set of program functions that speak highly of the machine's vision in real time. Made by Intel.

5.1.4 NUMPY

It is a python programming language framework that supports various arrays and matrices as well as a wide range of high-level mathematical functions that can operate on arrays.

5.1.5 DJANGO

Django is a free and source Python-based web framework that follows the model - template - looks at the build pattern. Allows you to quickly and successfully create a high-quality Web application, and is suitable for both front and backend.

5.1.6 ANDROID

Android offers a rich application framework that allows us to build new mobile devices using Android Studio IDE in the Java and Kotlin language areas.

5.2 PREPROCESSING

5.2.1 IMAGE AUGMENTATION

This technique allows us to artificially expand the data-set. It allows us to make multiple images from a single image by rotating, zooming, shearing, flipping and so on. These helped us to collect a dataset of over 10,000 plus images.

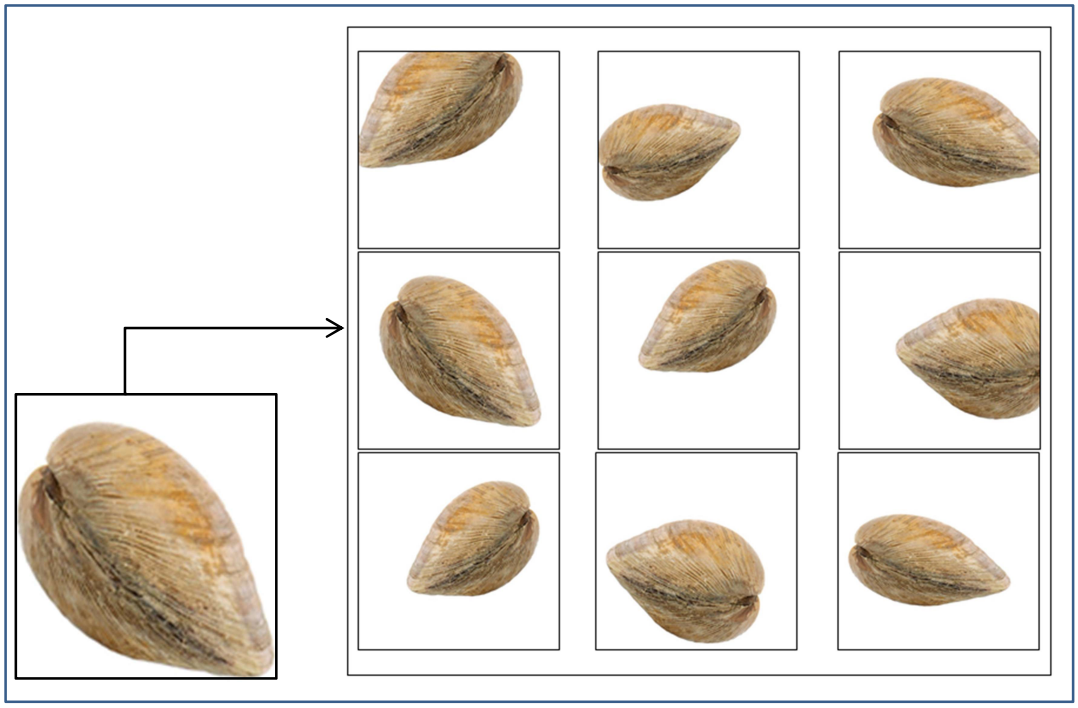


Fig 5.1 : Results of Augmentation

5.2.2 IMAGE ANNOTATION

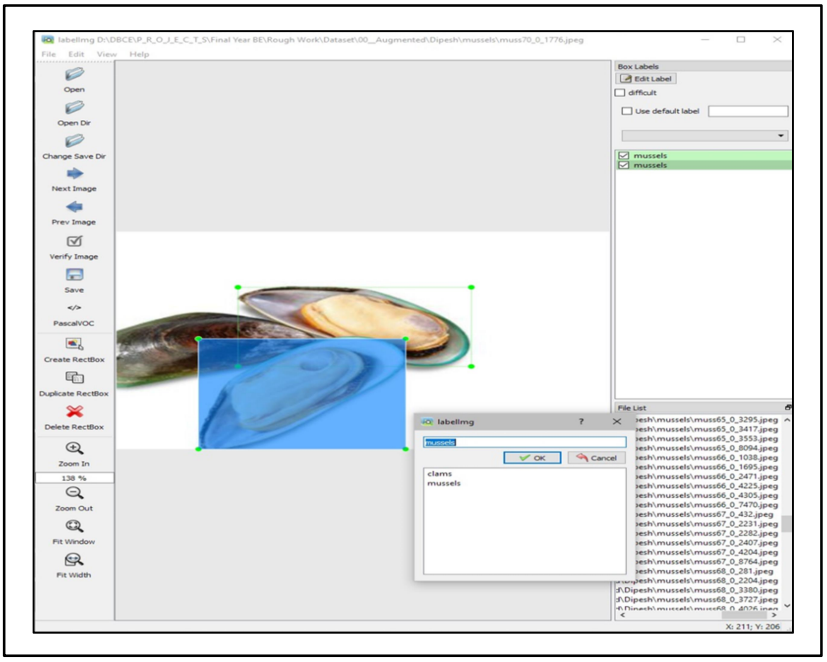


Fig 5.2: Annotations using LabelImg

In machine or deep learning, image annotation is the process of LabelImg or classifying the objects present in an image with the help of text, annotation tools or both in order to show the data features which will be recognized by the model on its own. It is sometimes called tagging, transcribing, or processing.

5.2.3 Comma Separated Values (CSV)

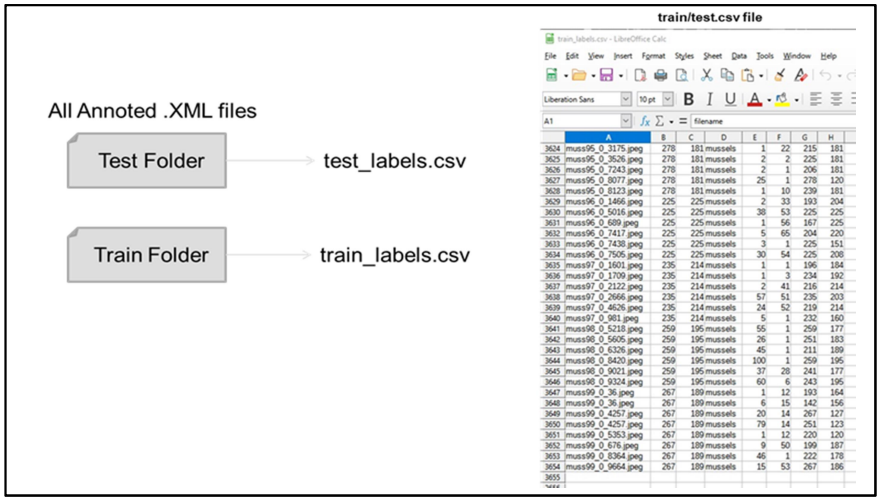


Fig 5.3: CSV files obtained

Comma Separated Value is a file format that allows us to store the data in tabular format. The Comma Separated Value File is saved using ".csv" extension. We can access a Comma Separated Value file with the help of spreadsheets like Google Spreadsheets or Microsoft Excel. If an image has multiple objects in it then csv helps us to determine how many objects are present in an image along with its label and location.

5.2.4 IMAGE PREPROCESSING

Images in the database are saved; Save the amount by which we will double the data before further processing. Our original images contain RGB coefficients at 0-255, but such values may be too high for our model to be processed (given the normal reading rate), so we point to values between 0 and 1 instead of 1. / 255.

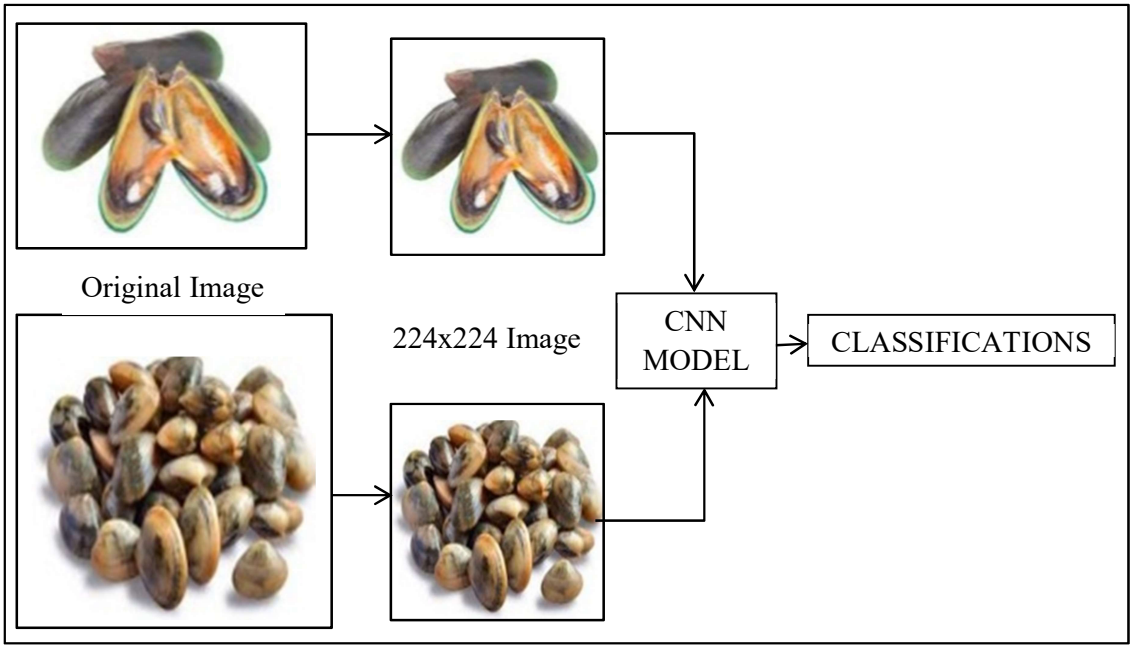


Fig 5.4: Image Resizing for the CNN model

Since neural networks receive inputs of the same size, all images need to be enlarged at a fixed size before uploading to CNN. The larger the dried size, the smaller the reduction. A slight decrease means a small defect of the symbols and patterns within the image. Images in the database were of different sizes; therefore, the images had to be resized before use as in modelling. Images have been edited to 224x224 for transfer as included in our split model.

5.3 IMPLEMENTATION OF PROJECT

5.3.1 ANDROID APPLICATION

This contains all the front-end part of the system. Here, the application is on android platform. It will ask for users' permissions for storage. Application will have different functions which will be interactive with the user and display various information. The application is designed in Android and Kotlin. Once the user uploads the image, it is sent to the server and the prediction of the input is done on the server and accordingly the output is displayed on the application. This will provide the user with an interface to view information about the shellfish. It will provide option to the user on selecting them he can either view the famous dishes or the nutrition and health benefits as per his requirements.

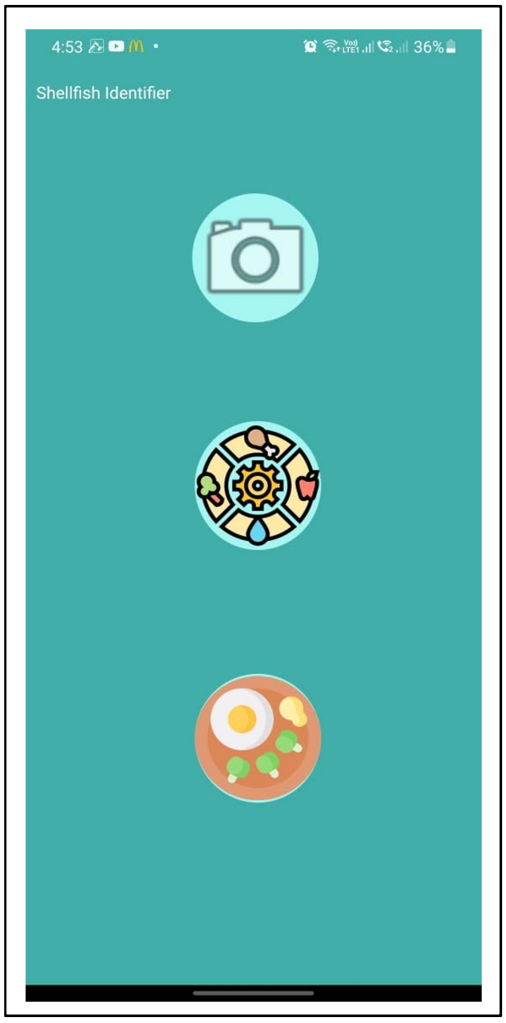


Fig 5.5: Application Home Screen

Fig 5.6: Application Dishes and Recipes Page

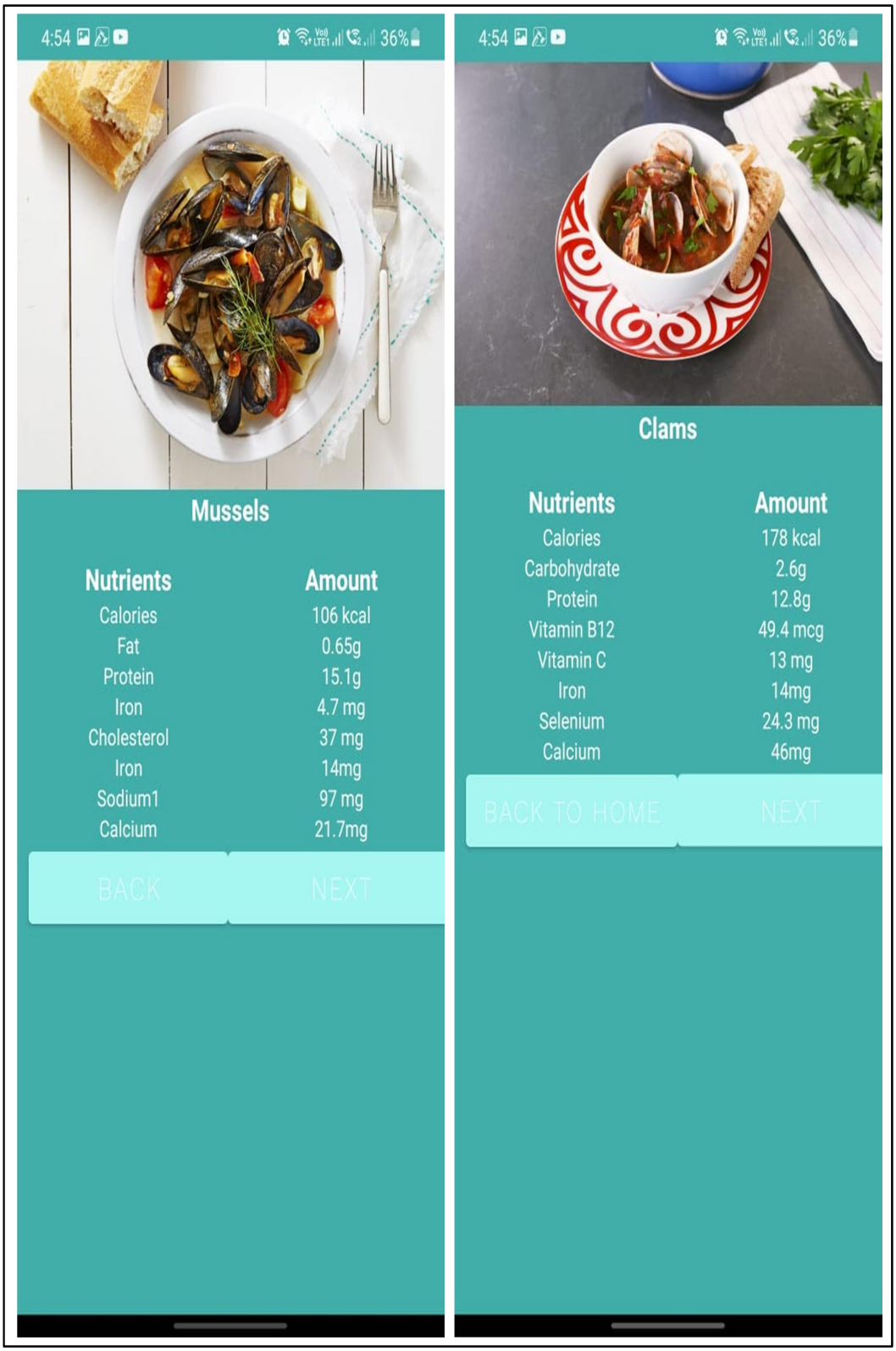


Fig 5.7: Application Nutrition Page

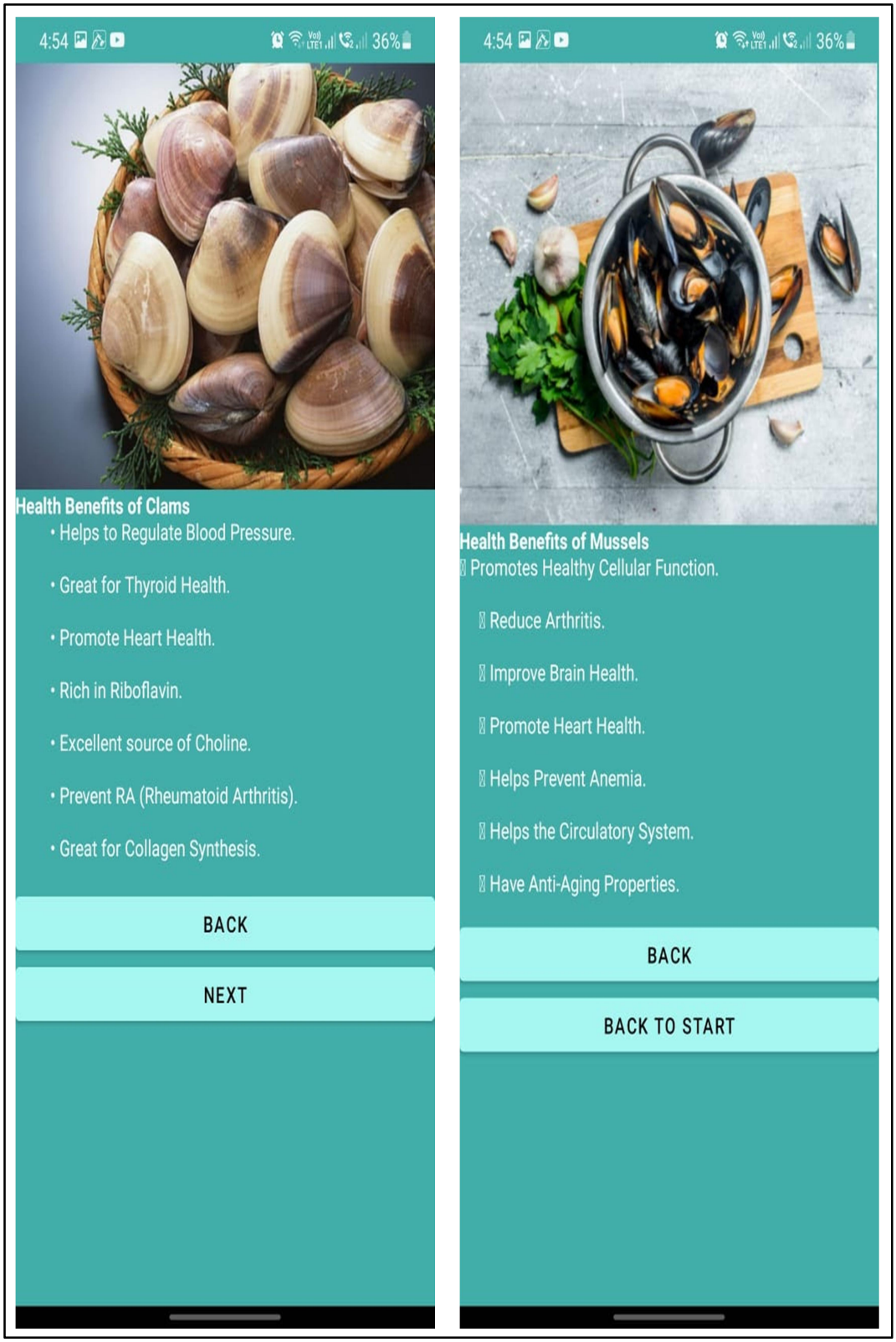


Fig 5.8: Application Health Benefits Page.

5.3.2 WEB APPLICATION

SSD Mobile-Net classification model is fully trained and adjusted to achieve maximum performance of our dataset. This model is then uploaded onto a web app which is designed using Django. Web app is hosted onto a local server from where user can use it on the android application upload the image on server and get classification results.

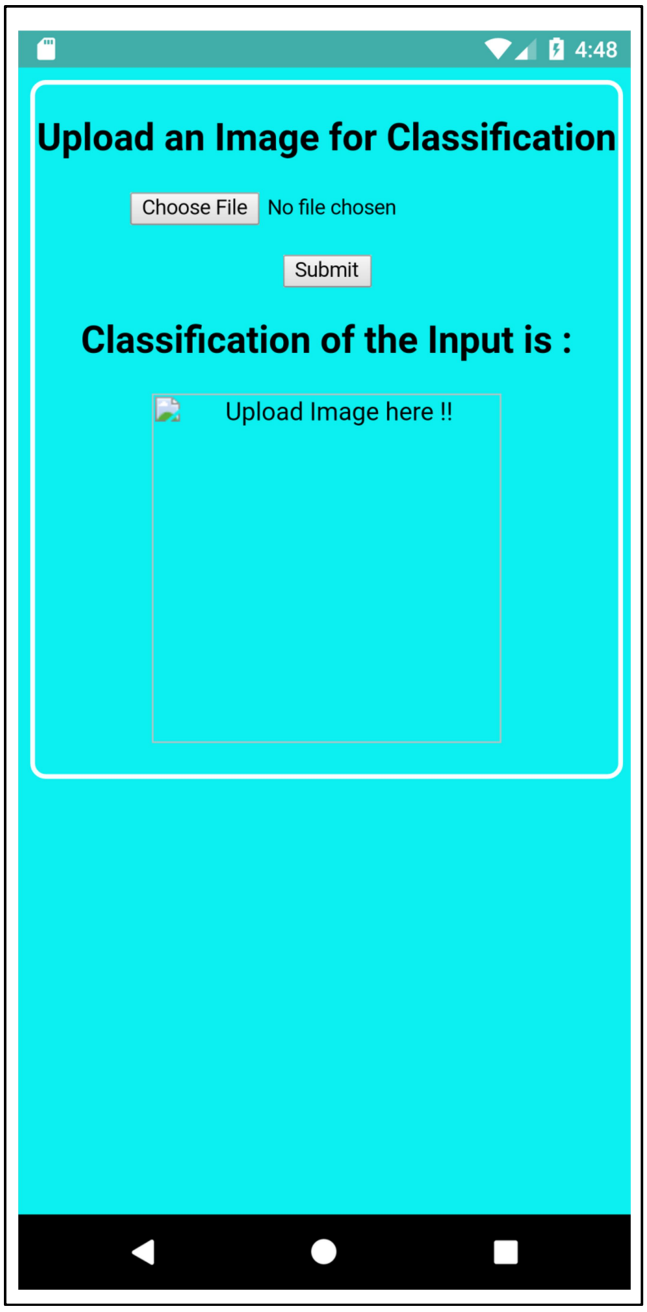


Fig 5.9: Web Application Home screen

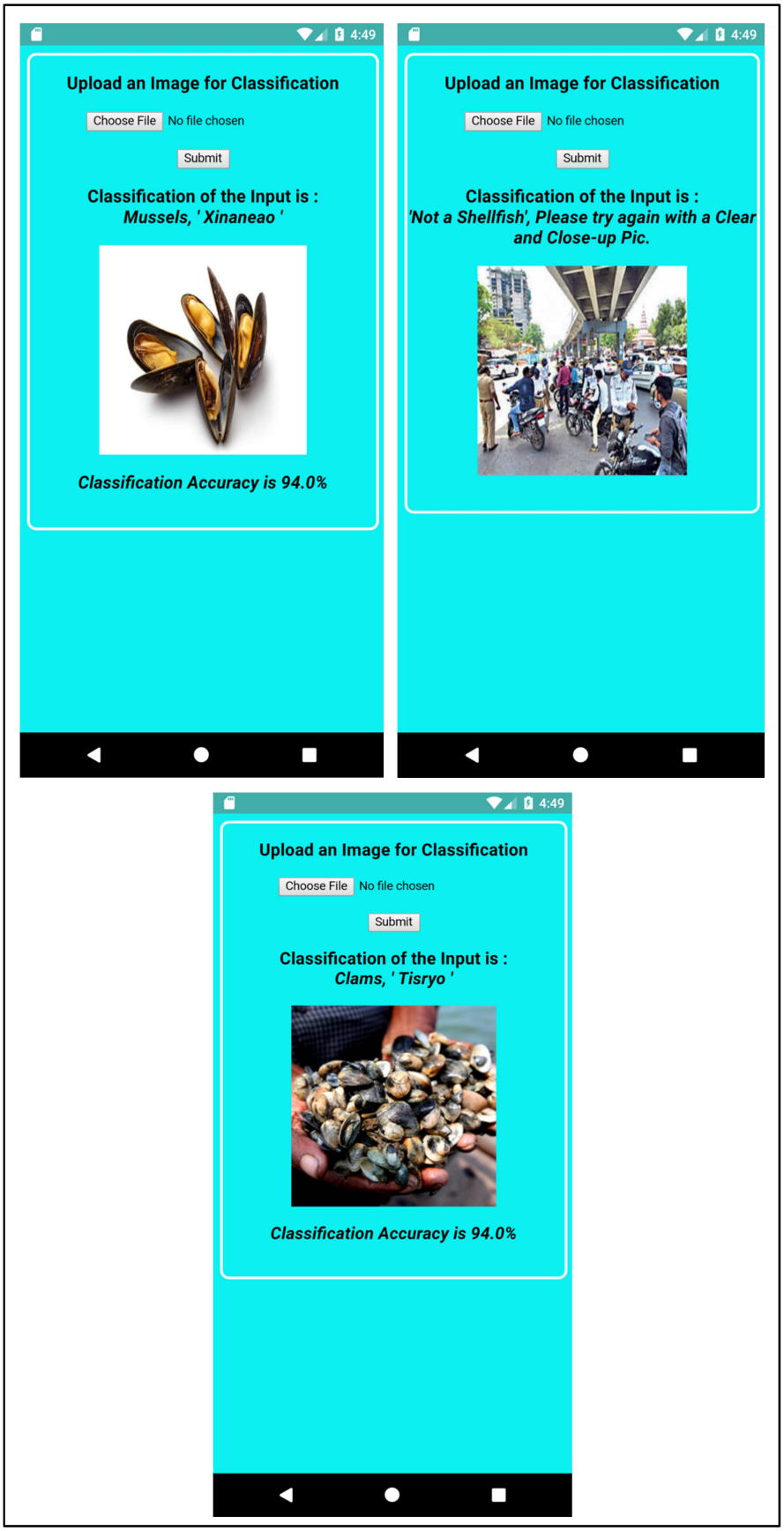


Fig 5.10: Web Application Classification Results

CHAPTER 6:

TESTING

Testing was carried out in multiple neural network models which utilized the fundus images as their common input.

Parameters taken under consideration while comparing different models are as follows:

Accuracy:

Accuracy is the ratio of correct predictions to the total amount of predictions that a model has generated during its run time.

This Accuracy is further sub-divided into three sections:

1. Train accuracy: The accuracy obtained after applying the model to the training data set is called the training accuracy model.

2. Validation accuracy: This precision is determined on the basis of a data set not used for testing but used (during the testing process) to verify (or 'test') the model's generalization capability.

3. Complete set accuracy: Provides overall model accuracy by combining train accuracy and previously obtained validation accuracy.

Loss:

Loss determines how much the values predicted in training data deviate from the true value.

6.1 ARCHITECTURE OF TRAINED MODELS USED

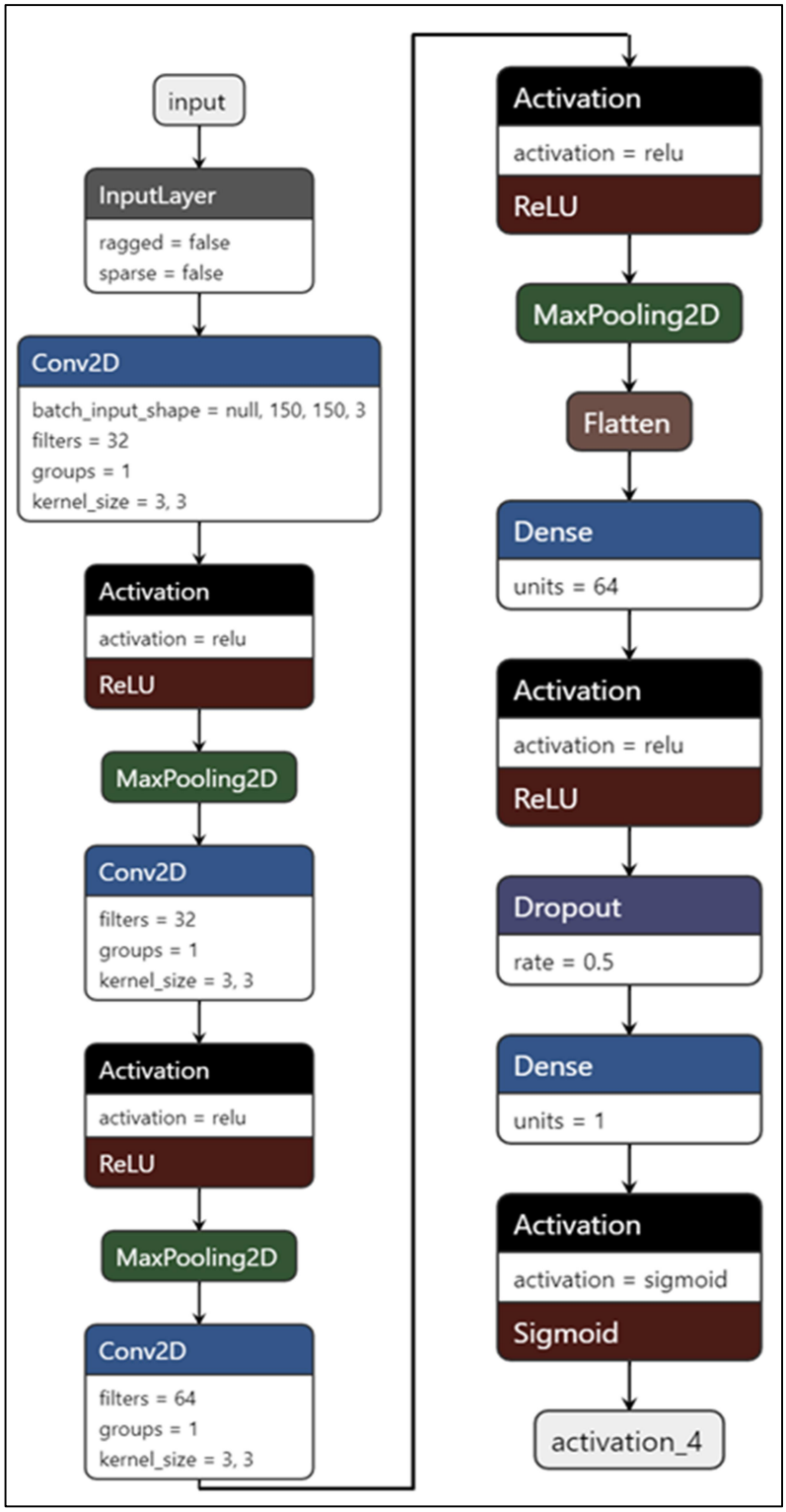


Fig 6.1: Basic CNN Architecture

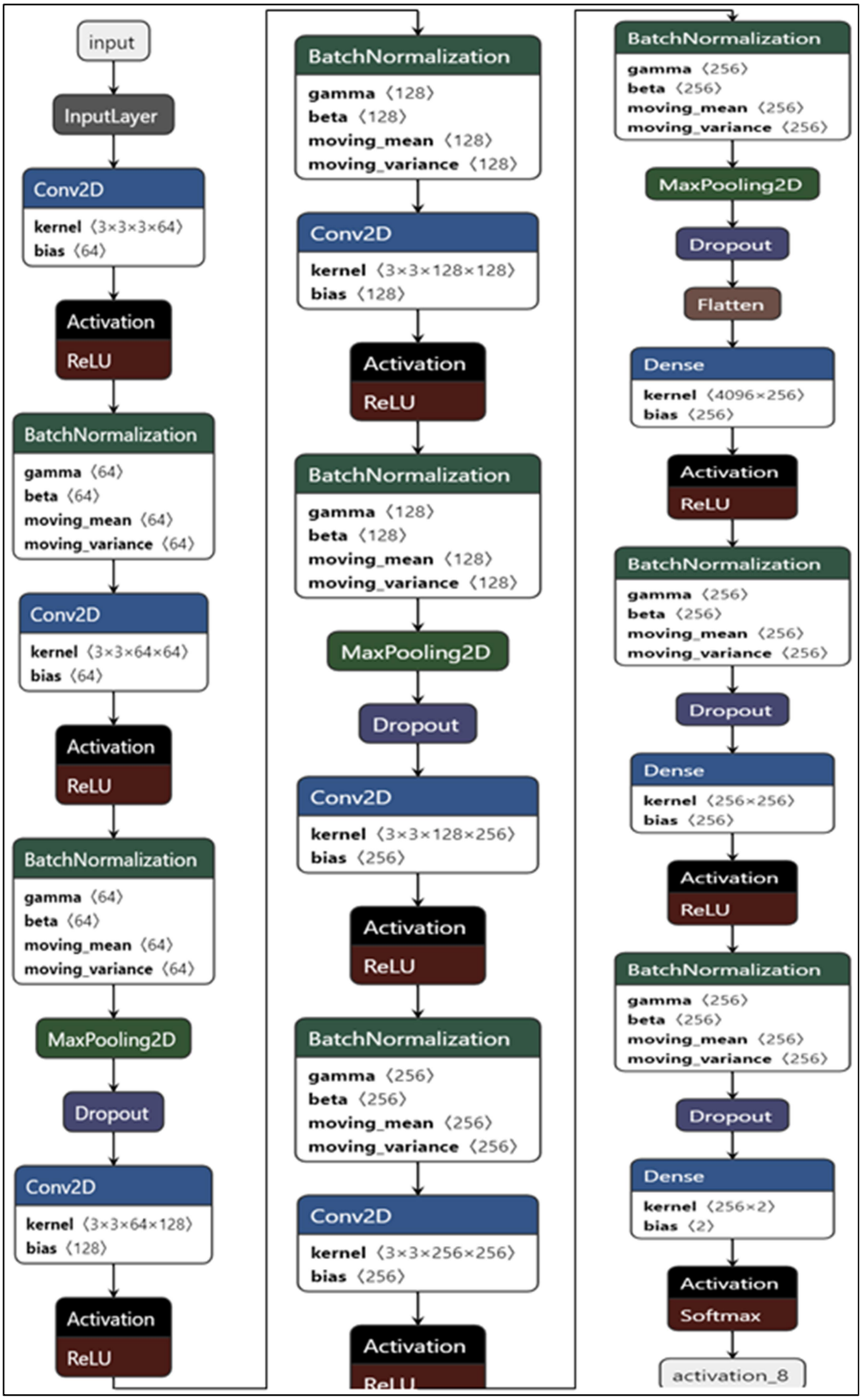


Fig 6.2: Little VGG Architecture

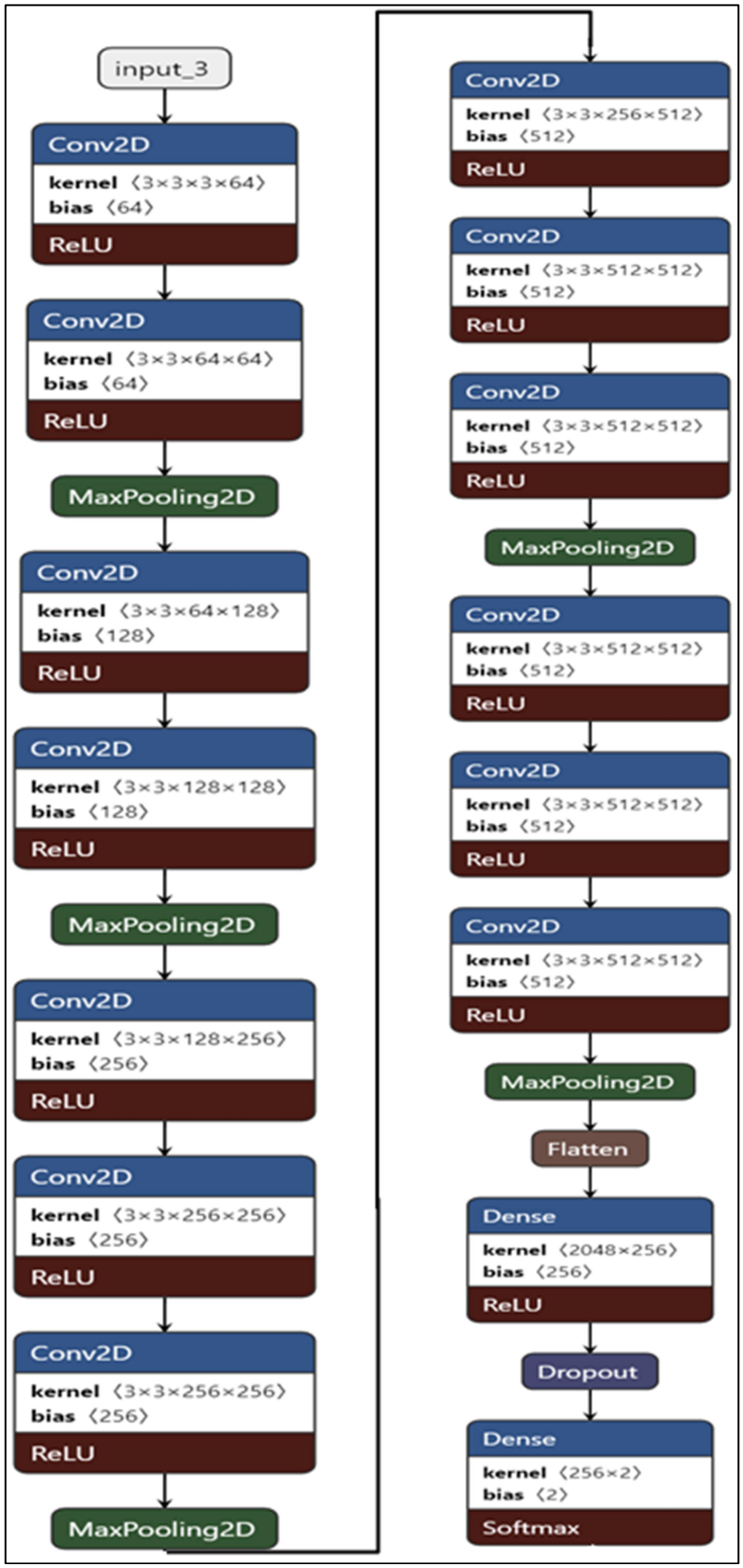


Fig 6.3: VGG16 Architecture

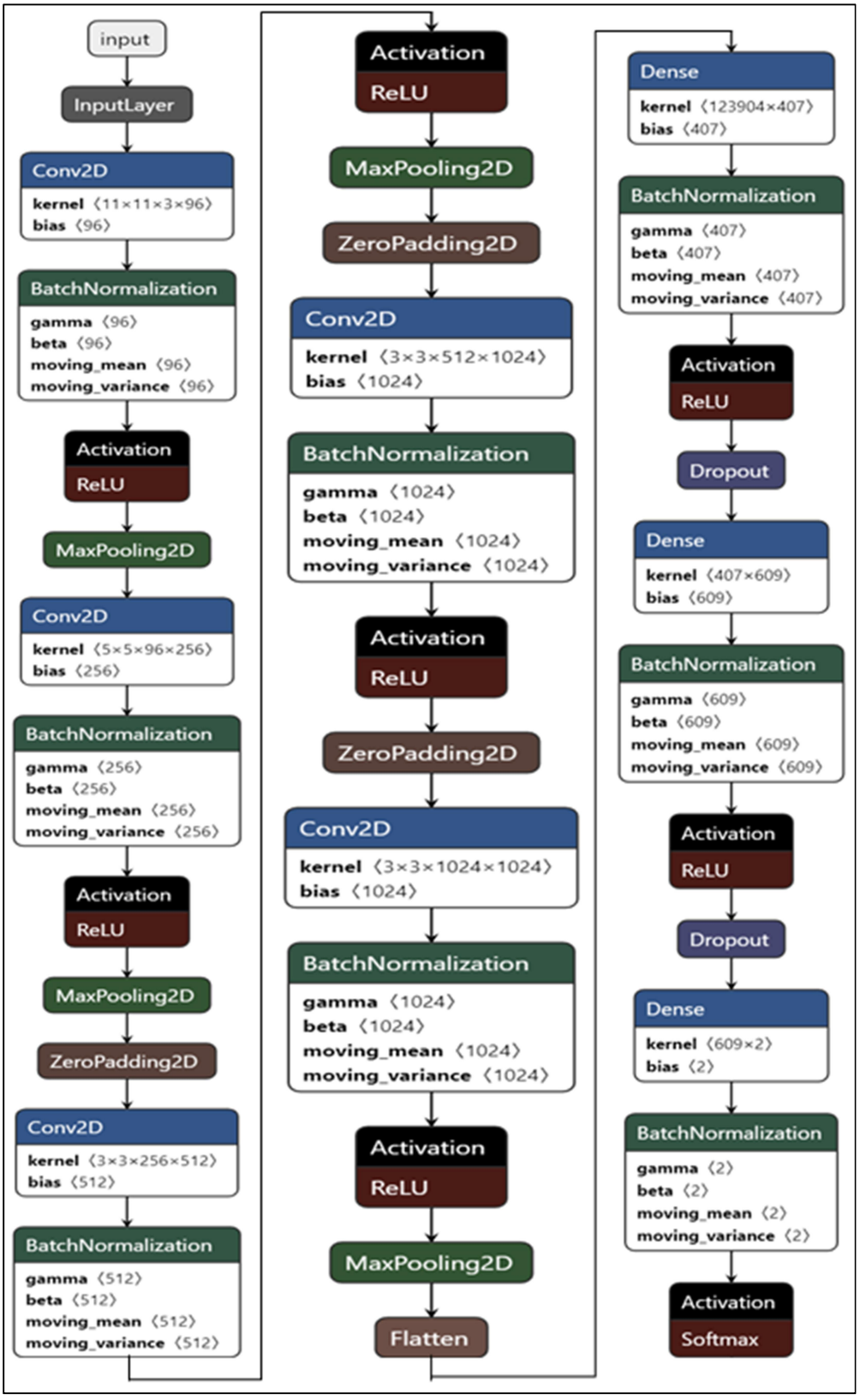


Fig 6.4: Alex Net Architecture

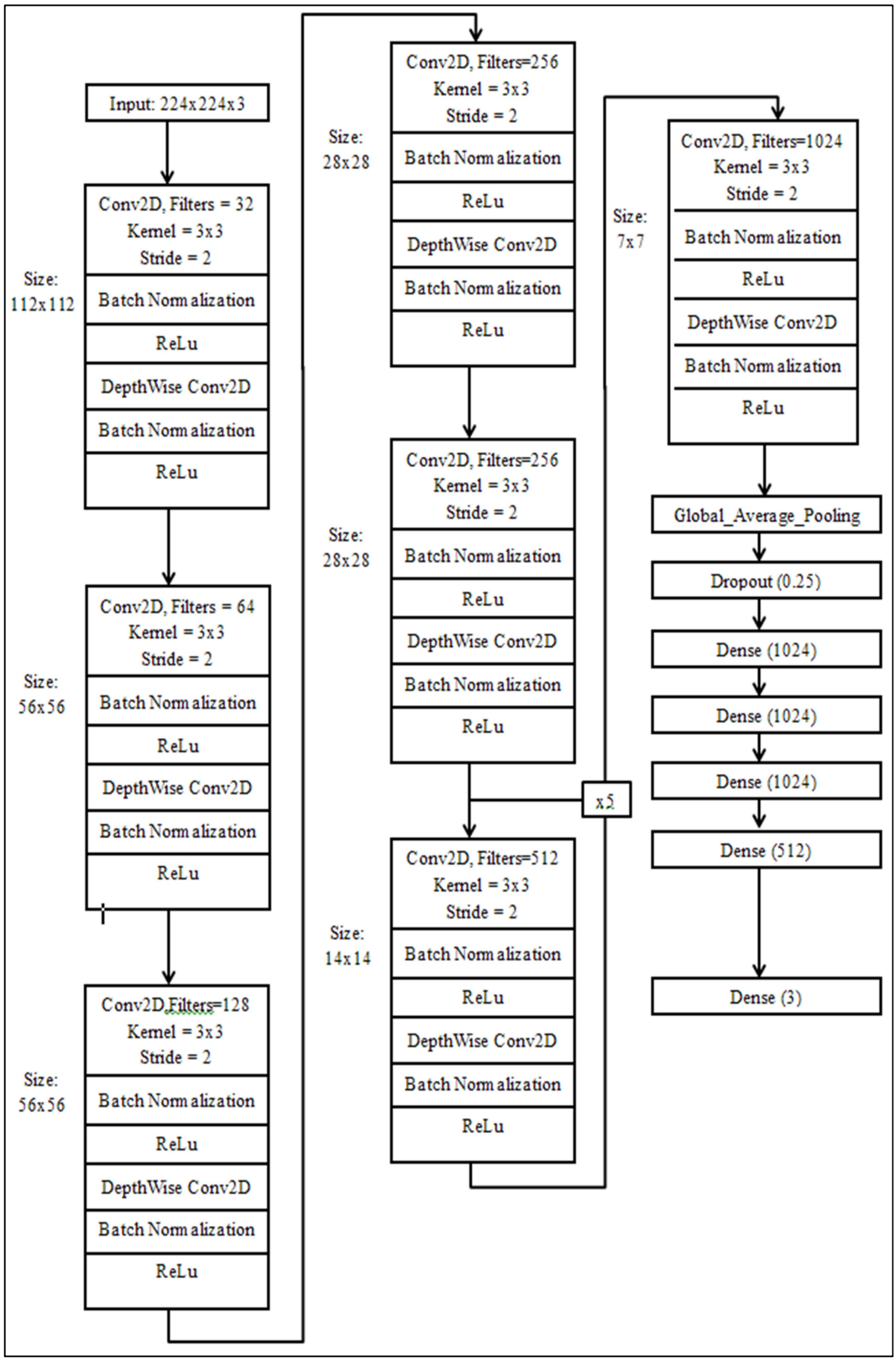


Fig 6.5: Mobile Net Architecture

6.2 OBSERVATIONS

6.2.1 BASIC CNN

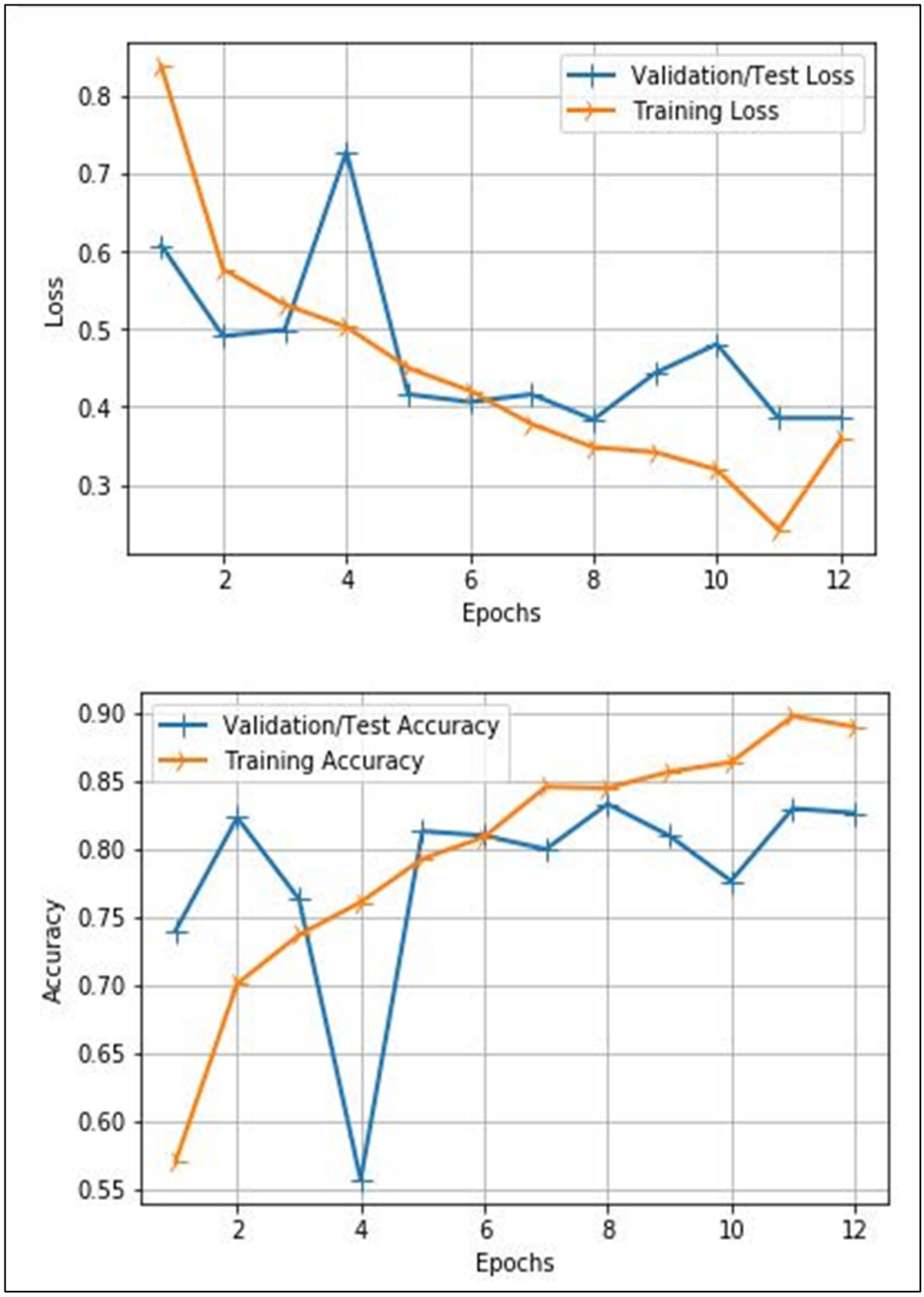


Fig 6.6: Basic CNN Training and Validation Accuracy and Loss on Un augmented Dataset

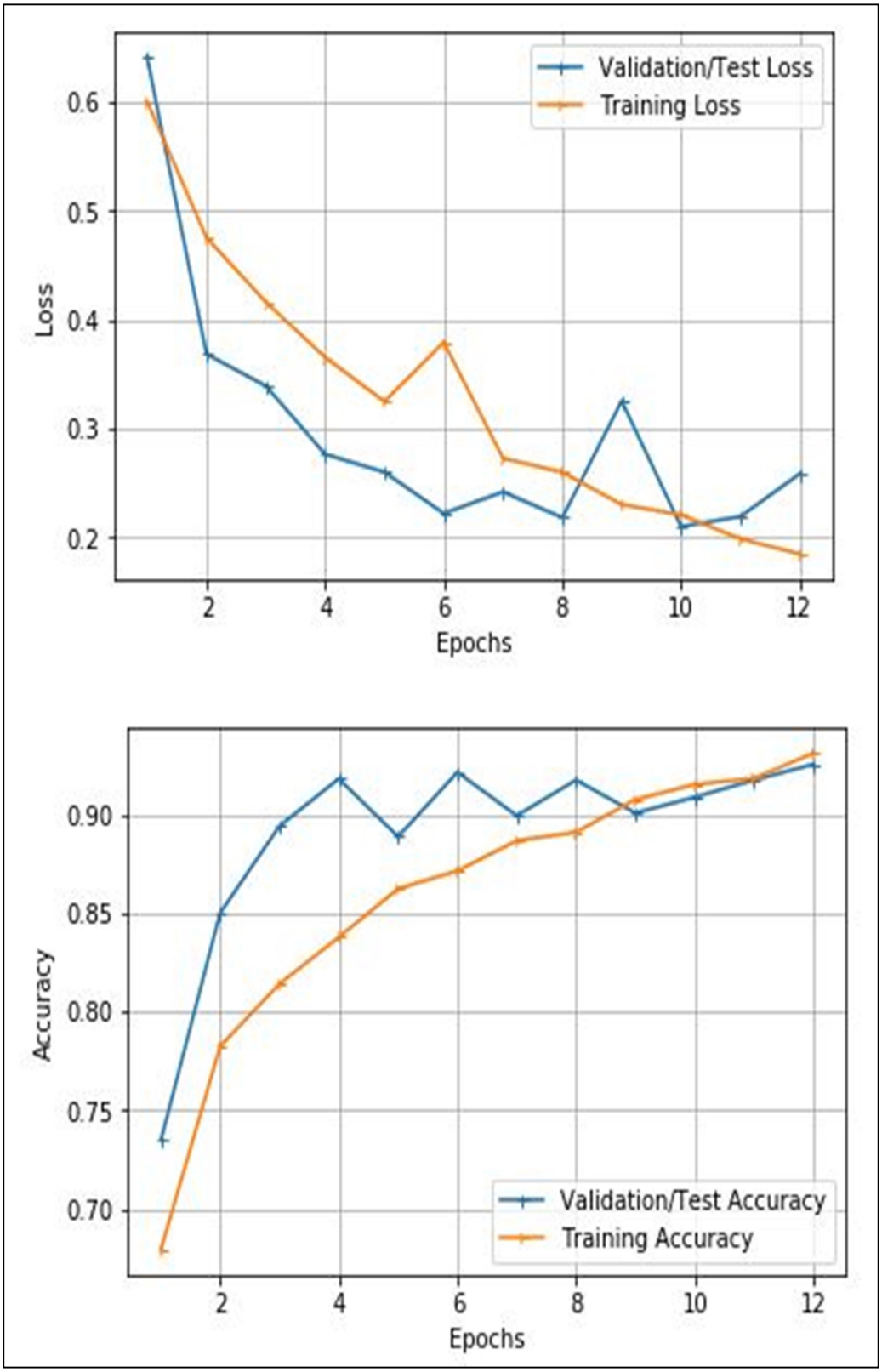


Fig 6.7: Basic CNN Training and Validation Accuracy and Loss on Augmented Dataset

6.2.2 LITTLE VGG

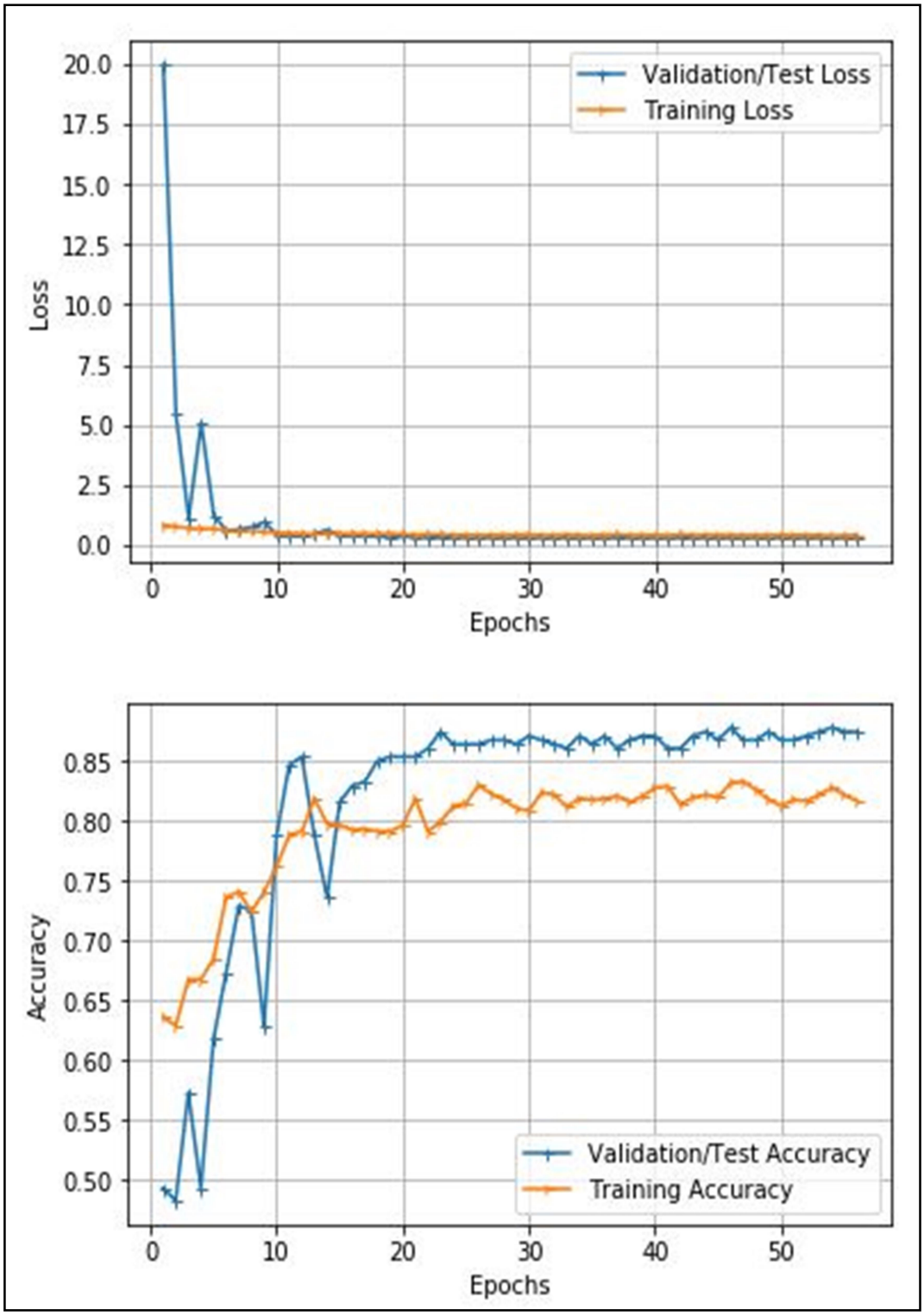


Fig 6.8: Little VGG Training and Validation Accuracy and Loss on 64 Epochs

6.2.3 VGG 16

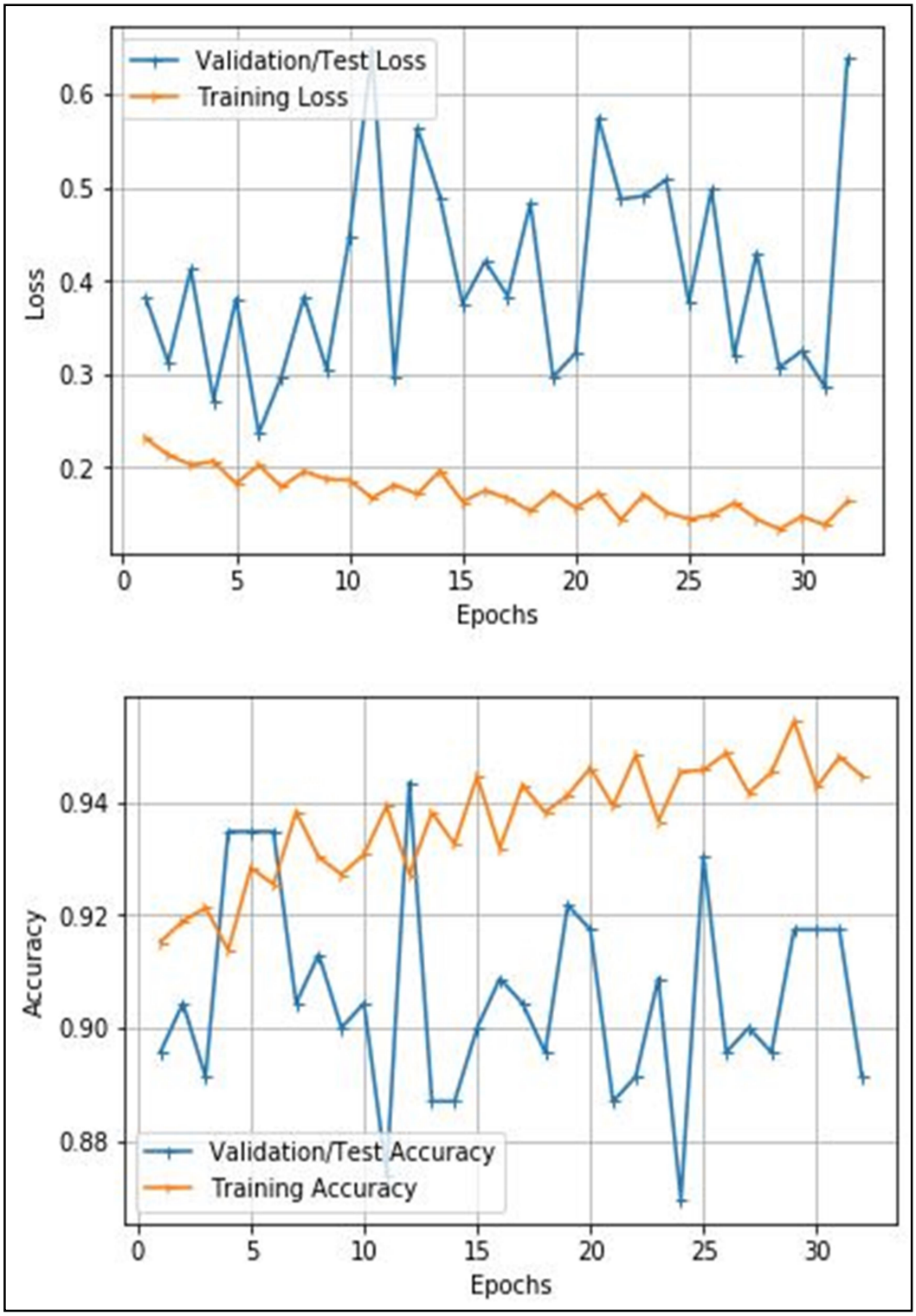


Fig 6.9: VGG 16 Training and Validation Accuracy and Loss without Early Stop

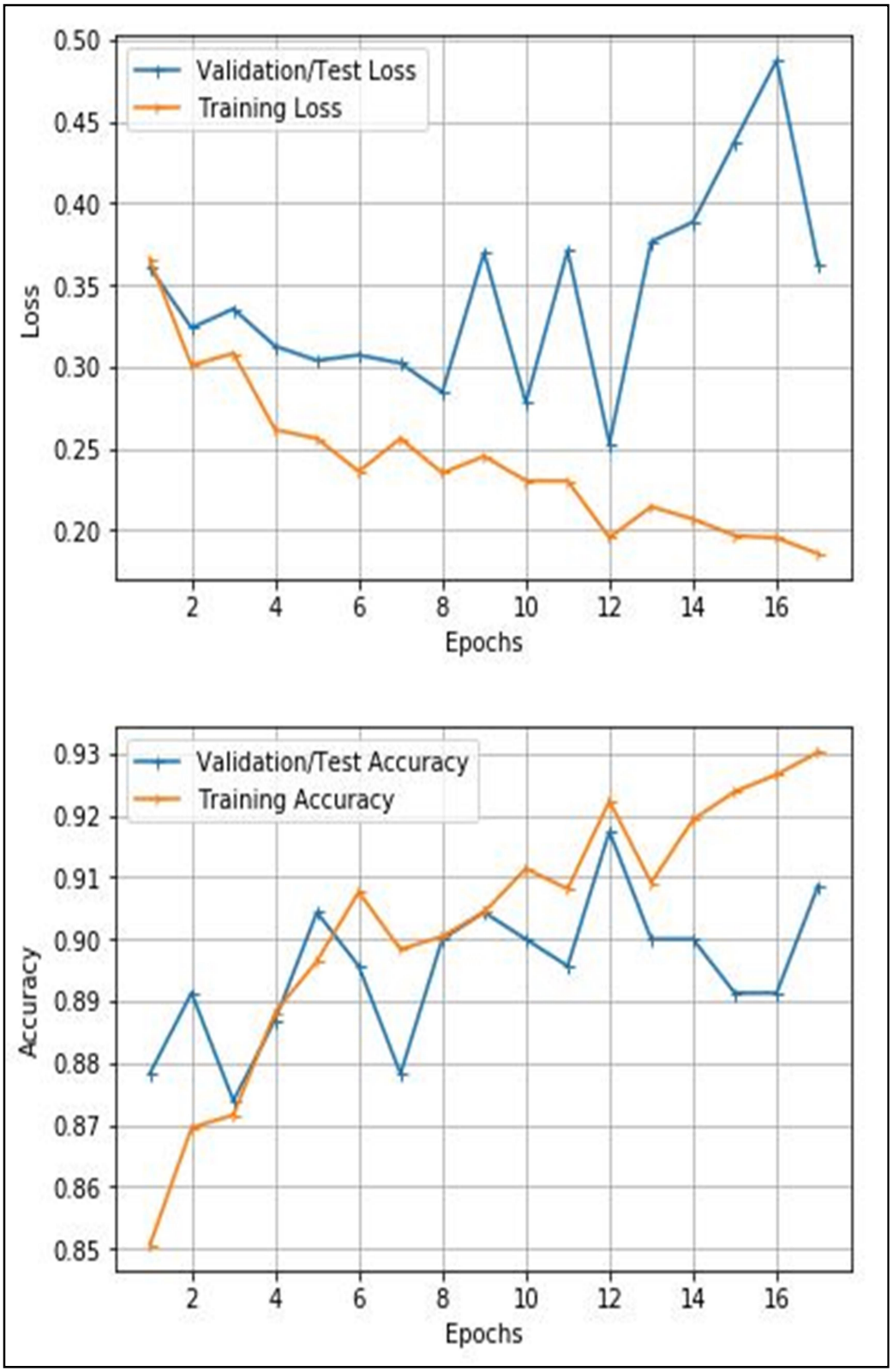


Fig 6.10: VGG 16 Training and Validation Accuracy and Loss with Early Stop

6.2.4 ALEX NET

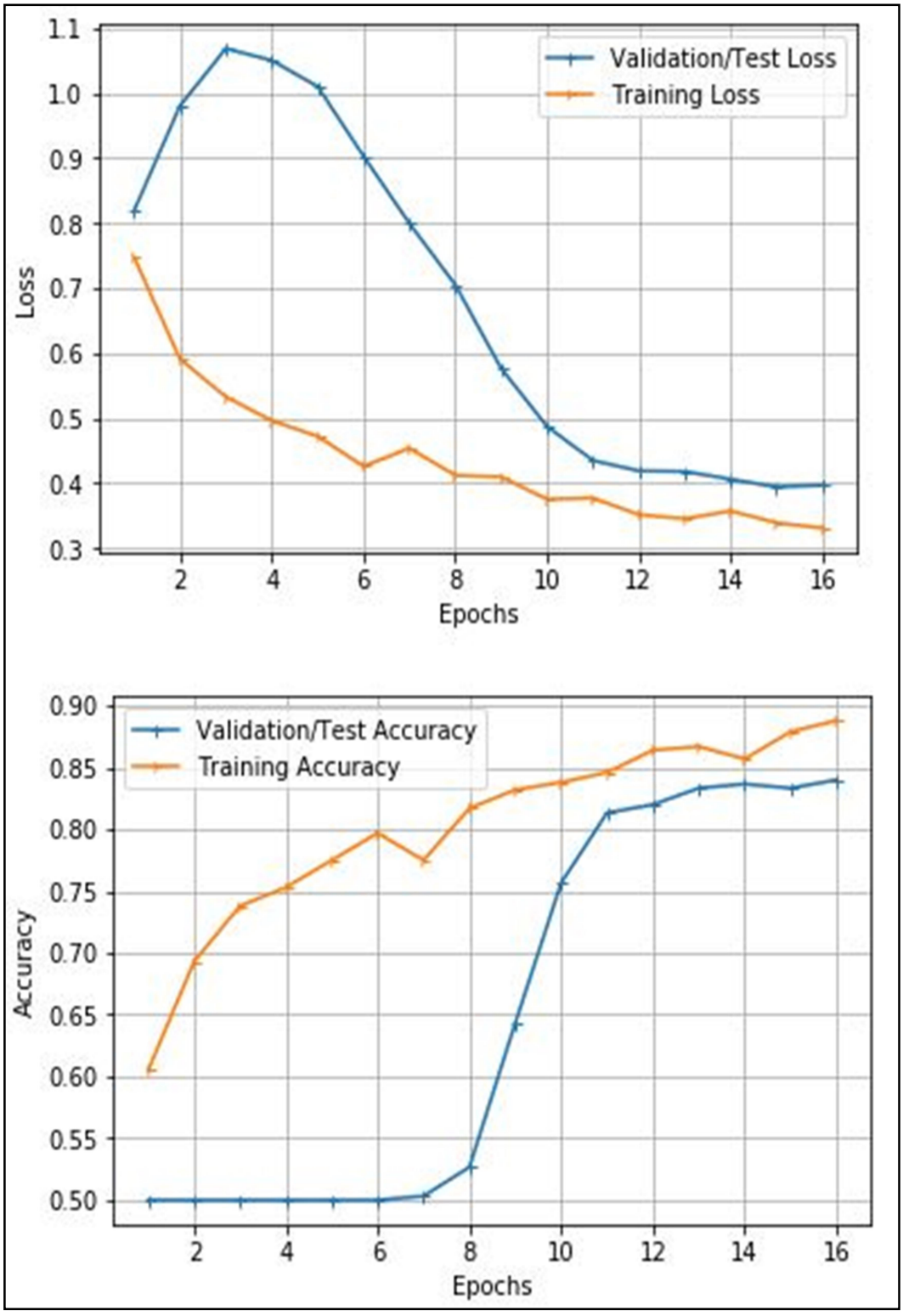


Fig 6.11: Alex Net Training and Validation Accuracy and Loss on 16 Epochs

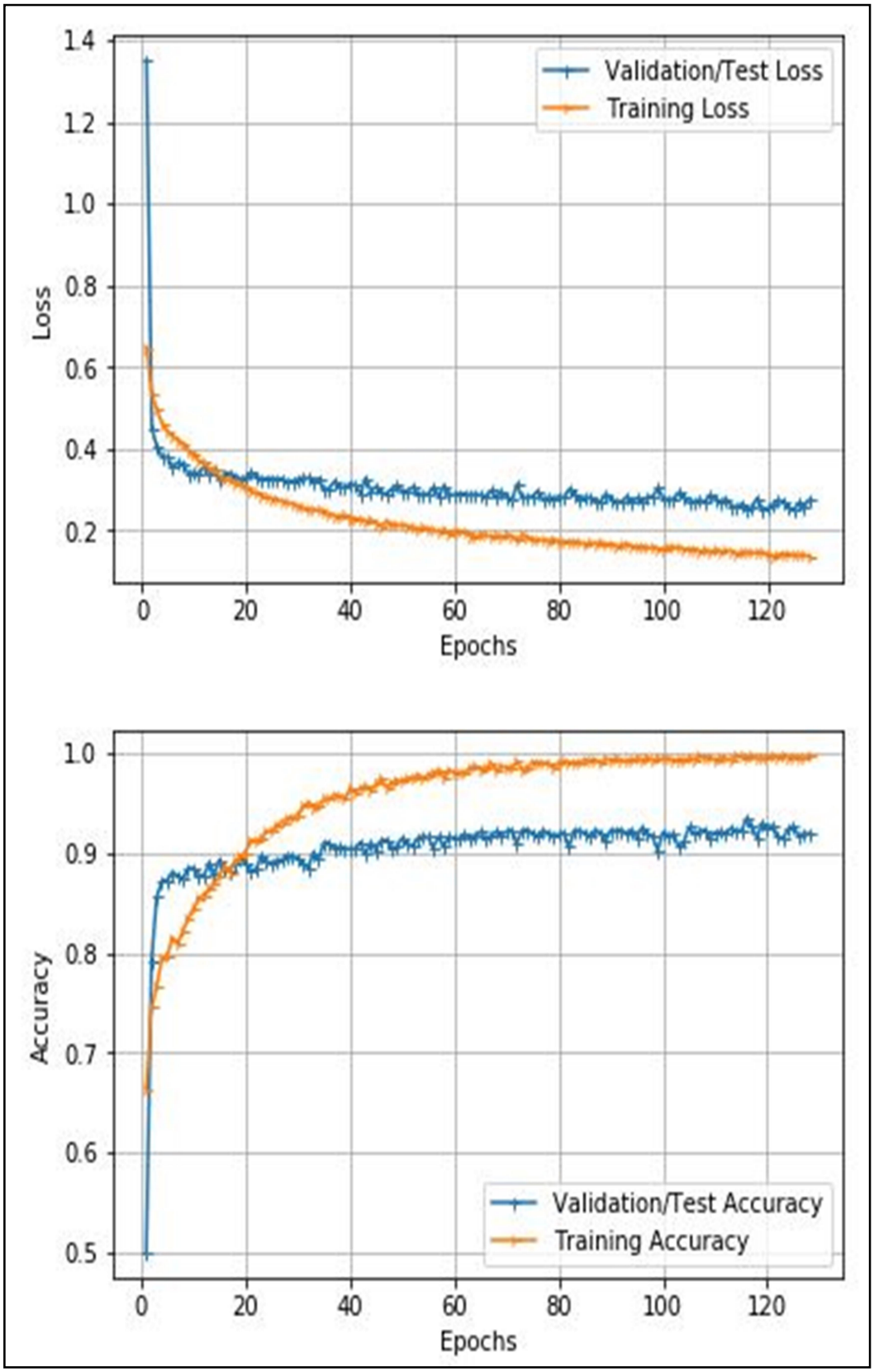


Fig 6.12: Alex Net Training and Validation Accuracy and Loss on 128 Epochs

6.2.5 MOBILE NET

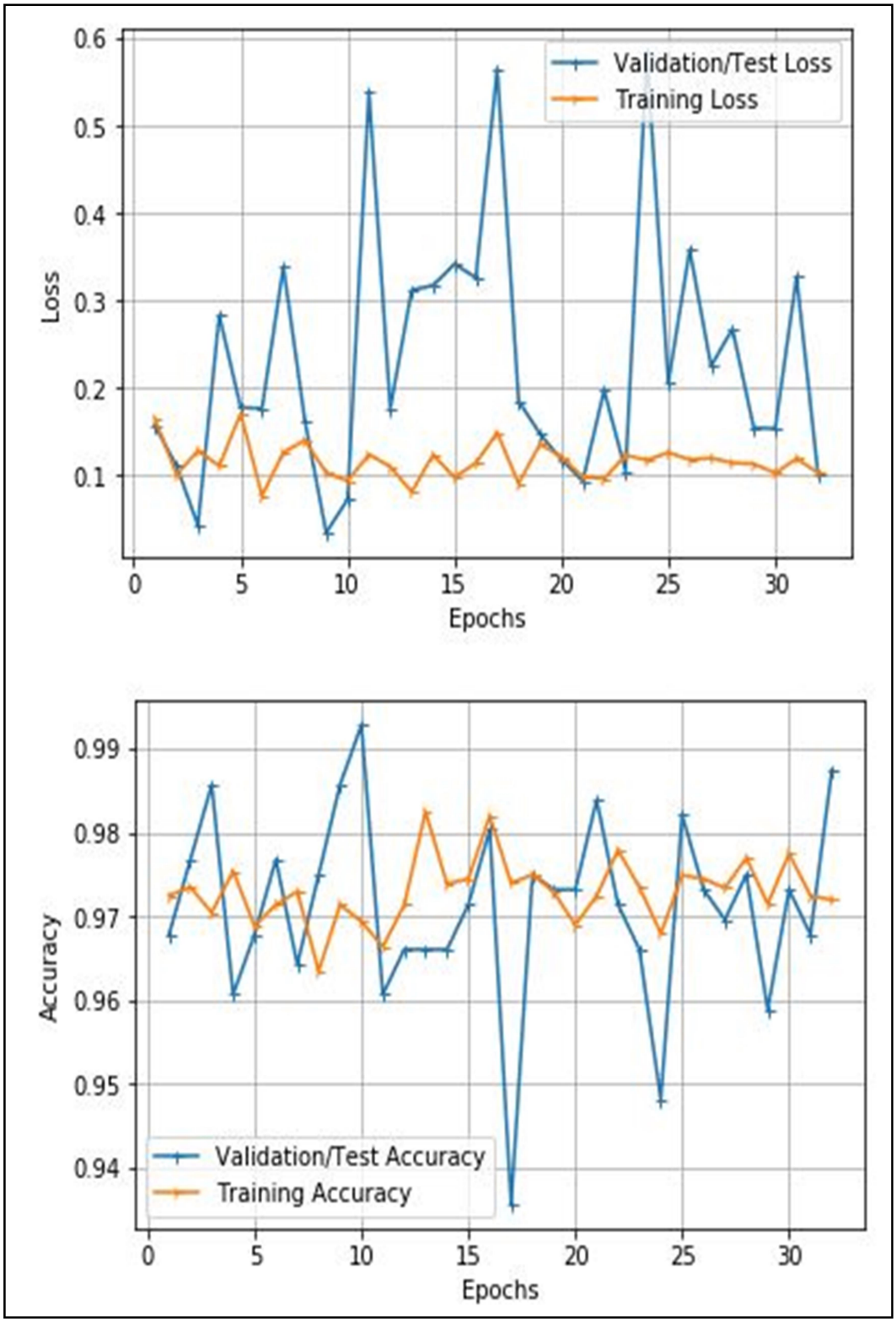


Fig 6.13: Mobile Net Training and Validation Accuracy and Loss on 32 Epochs

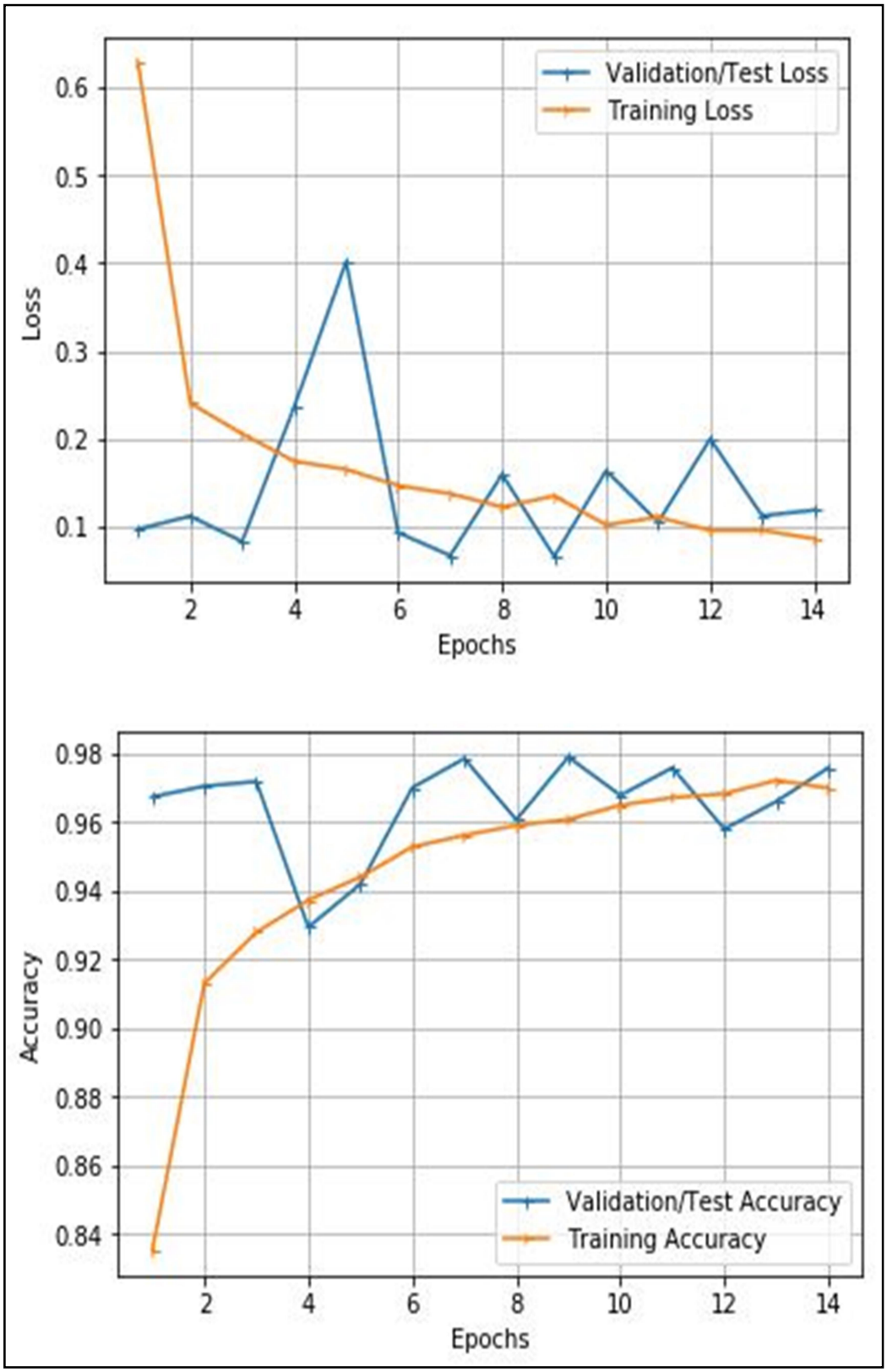


Fig 6.14: Mobile Net Training and Validation Accuracy and Loss with Early Stop

6.3 COMPARISONS

6.3.1 PERFORMANCE DURING TRAINING

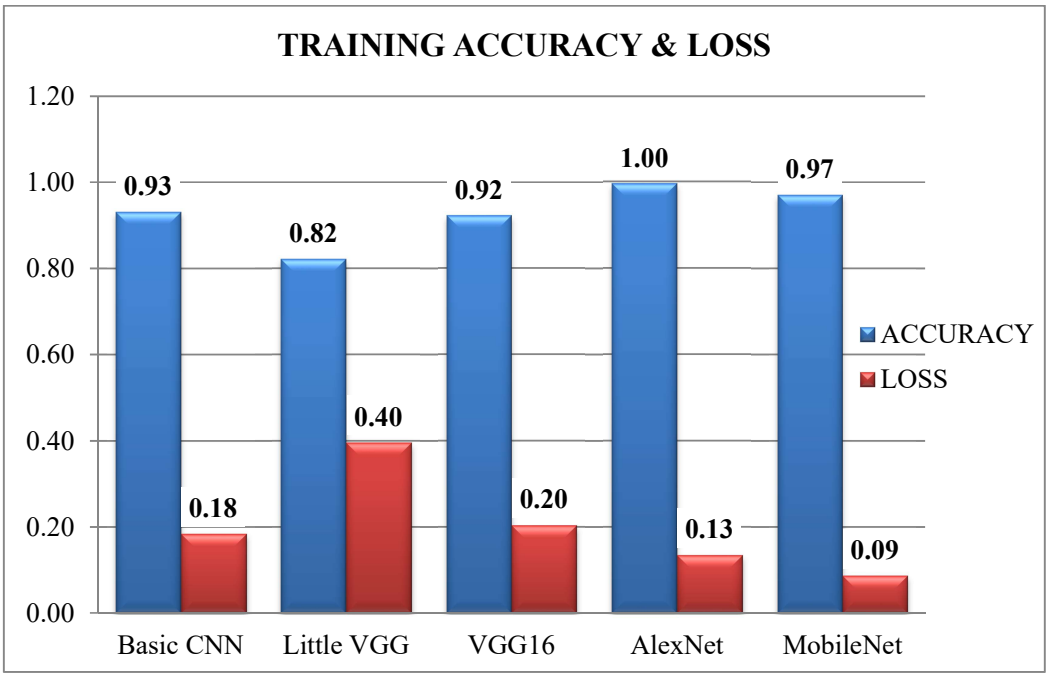


Fig 6.15: Training accuracy and loss of each model

6.3.2 PERFORMANCE DURING TESTING

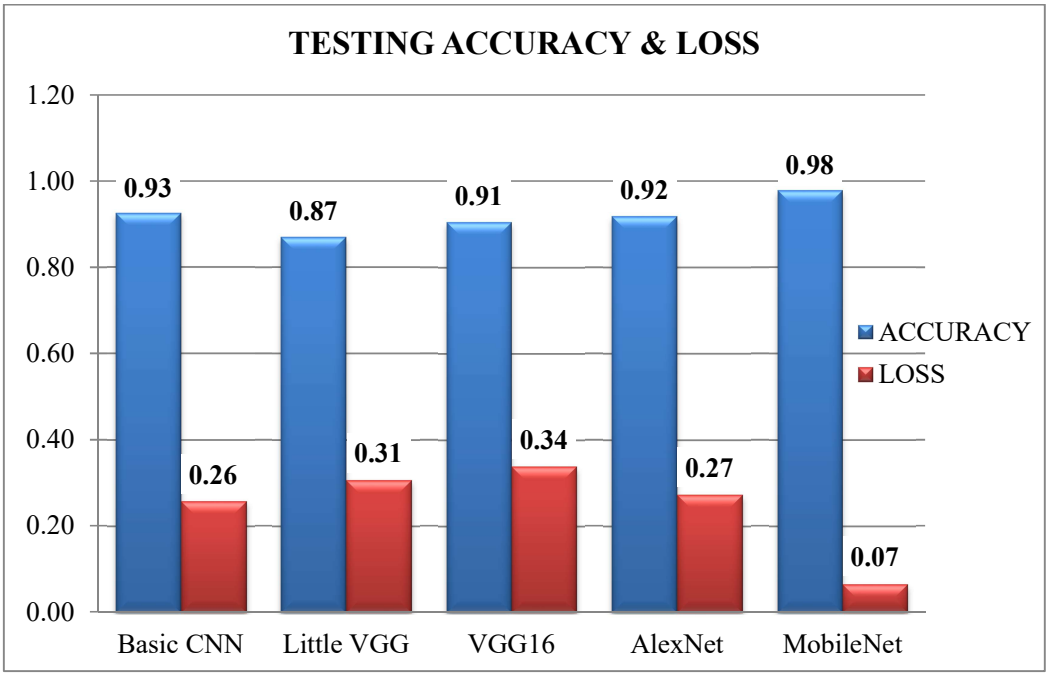


Fig 6.16: Testing accuracy and loss of each model

6.3.3 ACCURACY AND LOSS

	ACCURACY	LOSS
Basic CNN	0.9253	0.2585
Little VGG	0.8700	0.3072
VGG16	0.9060	0.3396
Alex Net	0.9193	0.2740
Mobile Net	0.9786	0.0665

Table 6.1: Final accuracy and loss of each model

6.3.4 PARAMETERS

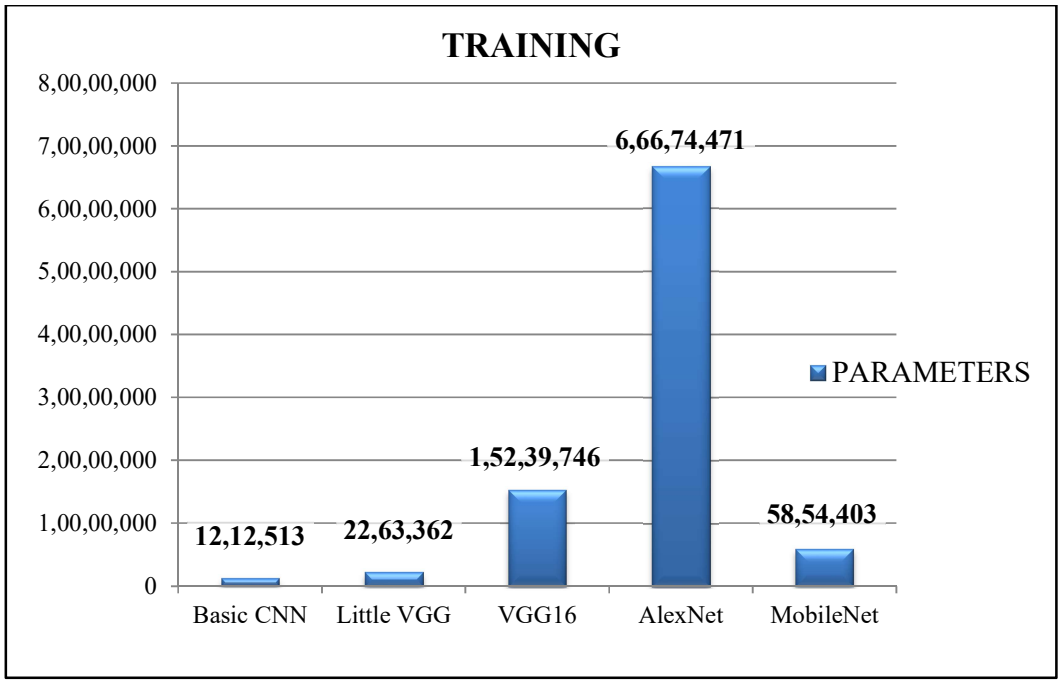


Fig 6.17: Trainable Parameters

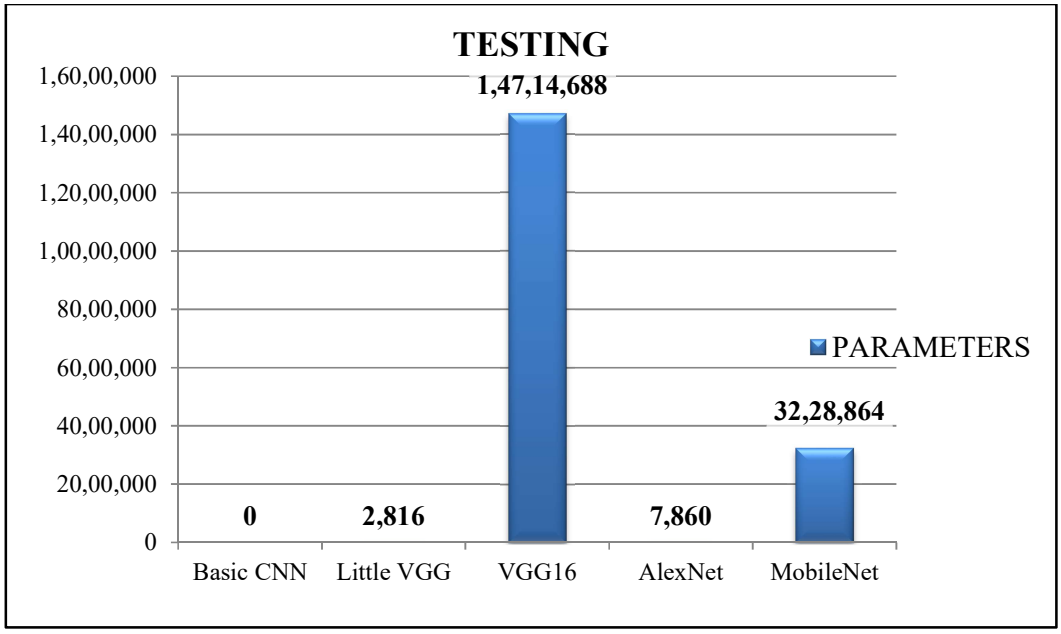


Fig 6.18: Testing Parameters

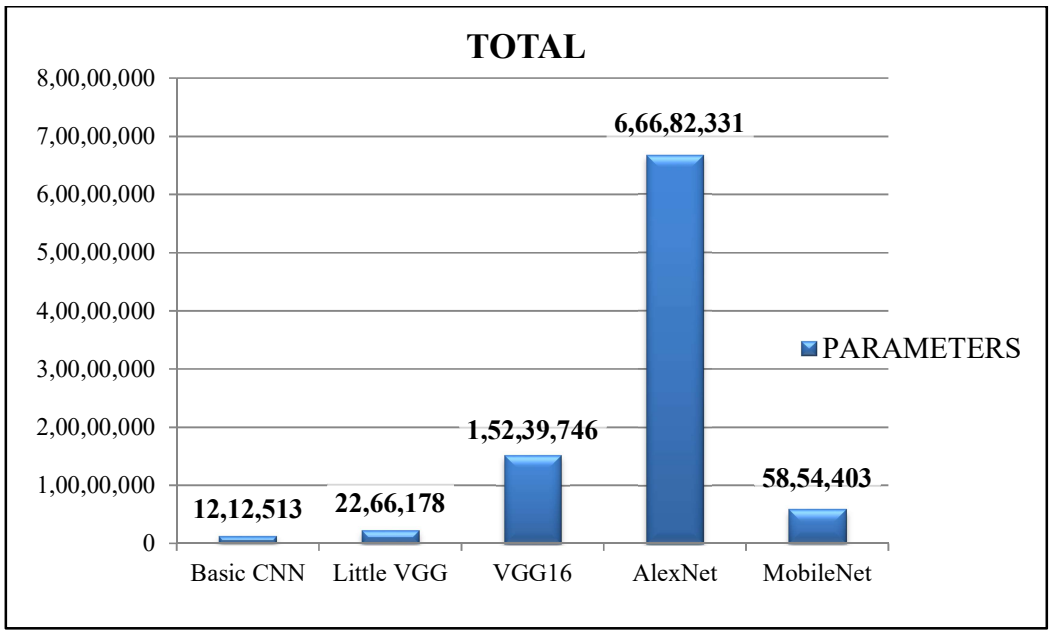


Fig 6.19: Total Parameters

CHAPTER 7:

CONCLUSION

This project presents an application for guiding user to know about the shellfish. Individuals who don't have proper knowledge about shellfish may find it difficult to recognise a shellfish. Especially Goa being a tourist destination, lot of foreigners visit Goa. This application will be very efficient for them to recognise the shellfish sold in local markets. It will help them to identify species and have complete information about the shellfish.

The application uses image classification and deep-learning algorithms to identify the shellfish species. Shellfish classification is achieved using Convolutional Neural Networks consisting of Single Shot Multi Box Detector architecture with Mobile-Net as its feature extractor as this proved to have a better efficiency and performance for devices with low hardware specification like smartphones. This was achieved using transfer learning that is 97.8% accurate having ReLu and SoftMax activation layer and Dropout as 25%.

This configuration was trained against 8058 images and the trained model was tested against 2250 images from the dataset. The android application uses trained model present on the server and takes input image of shellfish provided by user and successfully classifies in real-time into "Mussels" or "Clams."

The future scope would be to further improve the accuracy of the application and decrease the loss by experimenting with more convolutional neural networks using different alterations and combinations of different layers and inputs. Another aspect to improve is increasing the species of shellfish that it can identify successfully and provide user with variety of classifications. Also, enhancements can be given to user interface with more features.

REFERENCES

[1] T. Malisiewicz, A. Gupta and A. A. Efros, "Ensemble of exemplar-SVMs for object detection and beyond," 2011 International Conference on Computer Vision, Barcelona, Spain, 2011, pp. 89-96, doi: 10.1109/ICCV.2011.6126229.

[2] Felzenszwalb, P. F. and Girshick, R. B. and McAllester, D. and Ramanan, D. "Discriminatively trained deformed part models" Version5.

[3] Xiaofeng Ren and Deva Ramanan, "Histograms of Sparse Codes for Object Detection." 2013 IEEE Conference on Computer Vision and Pattern Recognition.

[4] Amr Suleiman and Vivienne Sze, "Energy-Efficient HOG-based Object Detection at 1080HD 60 fps with Multi-Scale Support". Massachusetts Institute of Technology, IEEE International Workshop on Signal Processing Systems (SiPS)

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 2014, pp. 580-587, doi: 10.1109/CVPR.2014.81.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, no. 9, pp. 1904-1916, 1 Sept. 2015, doi: 10.1109/TPAMI.2015.2389824.

[7] Keiron O'Shea and Ryan Nash, "An Introduction to Convolutional Neural Networks." Department of Computer Science, Aberystwyth University, Ceredigion, SY23 3DB. arXiv:1511.08458v2 [cs.NE] 2 Dec 2015.

[8] F. Sultana, A. Sufian, and P. Dutta, "A Review of Object Detection Models Based on Convolutional Neural Network." Springer Nature Singapore Pte Ltd. 2020.

[9] Wei Liu Dragomir Anguelov Dumitru Erhan Christian Szegedy Scott Reed, Cheng-Yang Fu, Alexander C. Berg, "SSD: Single Shot MultiBox Detector."

[10] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam, ""MobileNets:

Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv:1704.04861.

[11] Suxia Cui, Yu Zhou, Yonghui Wang and Lujun Zhai, “Fish Detection Using Deep Learning.” Hindawi, Applied Computational Intelligence and Soft Computing.

[12] P. Kaveti and H. Singh, "Towards Automated Fish Detection Using Convolutional Neural Networks," 2018 OCEANS - MTS/IEEE Kobe Techno-Oceans (OTO), Kobe, Japan, 2018, pp. 1-6, doi: 10.1109/OCEANSKOBE.2018.8559068.

[13] https://towardsdatascience.com/understanding-ssd-multibox-real-time-object-detection-in-deep-learning495ef744fab

[14] https://machinethink.net/blog/googles-mobile-net-architecture-on-iphone/

Plagiarism Checker X Originality Report

Similarity Found: 14%

Date: Saturday, July 31, 2021
Statistics: 903 words Plagiarized / 6549 Total words
Remarks: Low Plagiarism Detected - Your Document needs Optional Improvement.

CHAPTER 1: INTRODUCTION INTRODUCTION TO PROJECT Fish is apparently one of the most widely consumed foods on earth today. It is a staple food in many countries, with India ranked fifth in the world for fish consumption. Tripura, Lakshadweep, and Goa have been identified as the most fish-rich areas in India. Fish has been a major source of human protein and nutrients throughout history.

Shellfish are a major part of the fish above. Shellfish is a flexible and delicious source of nutritious food. Clams, Prawns, Oysters and Crabs are just a few of the famous fish. Shellfish are divided into two common categories that include Crustaceans and Mollusks. Crustaceans are easily identified by their strong, fragile bodies.

Mollusks are divided into single-shell gastropods like snails; bivalves with interlocking shells like clams; and octopus-like cephalopods have no shells. The mussel variety makes buying, building, and building fish a bigger challenge than working with meat and poultry. Adding to this challenge is the common problem of poorly labeled fish being sold at a high price in the market.

The proposed system will take image from the user, detect if there is any shellfish and if present identify species of that shellfish and will tell user its species along with local name. Also provide additional information like its nutrition facts, health benefits and special dish related to that shellfish. PURPOSE OF THE PROJECT The main purpose of our project is to develop an app that will help user to recognize shellfish by just capturing its image.

If an individual is not able to identify the shellfish, he must either ask someone knowledgeable about the shellfish or use the internet to manually compare and try to

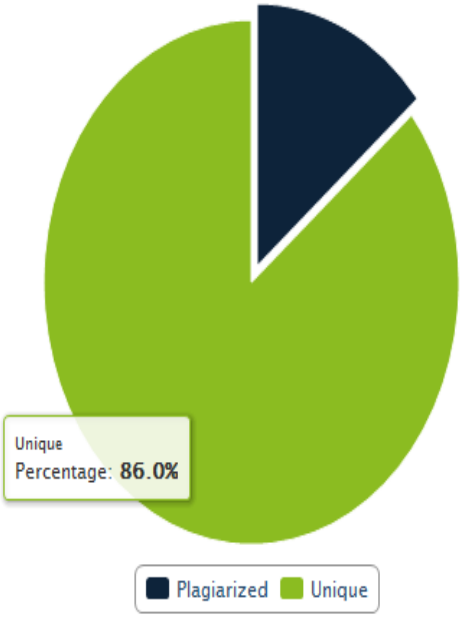
Summary Report

Print Save

Share Score with friends

Share Tweet Share

PlagiarismCheckerX Summary Report



Plagiarized Unique

Date	Saturday, July 31, 2021
Words	903 Plagiarized Words / Total 6549 Words
Sources	More than 104 Sources Identified.
Remarks	Low Plagiarism Detected - Your Document needs Optional Improvement.