Ethiopian Banknotes Recognition Using Convolutional Neural Network

Team Members

Dr. Dereje Tekilu, Principal Investigator

Dr. Harish Kalla, CO-PI

School of Electrical Engineering and Computing,

Department of Electronics and Communication Engineering

This project report is an authentic record of Department of Electronics and Communication Engineering, School of Electrical Engineering and Computing

Adama Science and Technology University
Adama, Ethiopia

Adama, Ethiopia
March 2021

ABSTRACT

Money transactions can be performed by automated self-service machines like ATM for money deposits and withdrawals, banknote counters and coin counters, automatic vending machines, and automatic smart card charging machines. These devices must be equipped with four essential functions: banknote recognition, counterfeit banknote detection, serial number recognition, and fitness classification. Therefore, we need robust system that can recognize banknotes and classify into denominations that can be used in these automated machines. However, most widely available banknote detectors are hardware systems that uses optical and magnetic sensors to detect and validate banknotes. These banknote detectors are usually designed for specific country banknotes. Reprogramming such system to detect our country, Ethiopia, banknotes is very difficult. In addition researchers have developed Ethiopian banknotes recognition system using Deep Learning Artificial Intelligence technology like CNN, and R-CNN. However, in these systems dataset used for training is relatively small, and accuracy of banknotes recognition is found smaller. The existing systems also not included implementation using embedded system. In this research work, we collected various Ethiopian currencies with different ages and conditions and applied various optimization techniques for CNN architects to identify the best optimization technique and CNN model with best accuracy. Experimental analysis demonstrated, *MobileNet with RMSProp optimization technique in batch size 32* is a robust and reliable Ethiopian banknote detector with best accuracy. Selected model is implemented on Embedded system (Raspberry Pi 3 B+) and Web based User Interface (UI) is developed and verified.

Keywords: Convolutional Neural Network, Ethiopian banknotes, Optimization, Embedded System

ACKNOWLEDGEMENT

First and foremost, thanks and praise to the Our God, the Lord, the Almighty for his immense care, love, and blessings throughout our research program to successfully complete the research work. Thank you God for guiding, strengthening, and holding us tight at times of hardships and pouring out the required knowledge to bring out a good output for the research and in turn help my fellow beings in a better way.

We would like to express our sincere gratitude to Electronics and Communication Department and School of Electrical Engineering and Computing, Adama Science and Technology University, Adama, Ethiopia for giving opportunity and constant support to complete this research project.

We extend deepest gratitude to Mr. Tadesse Hailu, Associate Dean for Research and Technology Transfer (ADRTT), SoEEC, Adama Science and Technology University, Adama, Ethiopia for his supportive suggestions and fruitful guidelines and also the former ADRTT of SoEEC, Mr. Girma Debele for initial advises and guidelines.

We extend deepest gratitude to Dr. Satyasis Mishra, Associate Professor, Department of Electronics and Communication Engineering, SoEEC, Adama Science and Technology University, Adama, Ethiopia for his loyal suggestions.

We would like to express special thanks to Mr. Tesfaye Balcha, Branch Manager and Mrs.Tigist, Technical assistant, Cooperative Bank of Oromia, Deka Adi Branch, Adama, Ethiopia for all their help in collection of research data.

We must acknowledge the Vice President for Research and Technology Transfer office and Office of Research Affairs, for technical guidance and administrative support for pursuing our research work.

We would like to express our sincere gratitude to Procurement and Property Administration office, Adama Science and Technology University, Adama, Ethiopia for purchasing research inputs and constant support to complete this research project.

We must acknowledge the academic resource that we received from Adama Science and Technology University, Adama, Ethiopia giving us a comfortable and active environment for pursuing our research work.

CONTENTS

List of Figures

List of Tables

List of Acronyms

Adadelta	:	Adaptive learning rate method
Adagrad	:	Adaptive Gradient algorithm
Adam	:	Adaptive Moment Estimation
ANN	:	Artificial Neural Network
ATM	:	Automated Teller Machine
BPNN	:	Back Propagation Neural Network
CAD	:	Computer Aided Design
CNN	:	Convolutional Neural Network
CT-Scan	:	computerized tomography Scan
EEPCO	:	Ethiopian Electric Power Corporation
GGD	:	Generalized Gaussian Density
GLCM	:	Gray Level Co-occurrence Matrix
HOG	:	Histogram of Gradient
HSV	:	Hue, Saturation, Value
HWT	:	Haar Wavelet Transform
LBP	:	Local Binary Pattern
LDA	:	Linear Discriminate Analysis
LVQ	:	Linear Vector Quantization
MLP	:	Multi-Layer Perceptron
Nadam	:	Nesterov Momentum into Adam
PCA	:	Principal component analysis

CHAPTER-1

INTRODUCTION

1.1. BACKGROUND

In the digital world, money transactions play a vital role in day-to-day life. Thousands of organizations perform digital transactions every day. Moreover, in digital technology world, banking operations and authentication of currencies are mandatory in many applications such as smart card charging machines to pay electricity and transport, money exchange machines, Automated Teller Machine (ATM), vending machines for drinks, toll gate ticket-vending machines at highways, parking meters at shopping malls, and so on. These devices are user friendly and make the jobs easier and fast. These devices prevent users from authenticating and denominating banknotes. The feature extraction is the important method in the process of classification of banknotes.

The banknote authentication process of such machines consists of a series of operations. They are Fake currency identification, Banknote denomination recognition and serial number recognition. The paper currency denomination is distinguished by using the banknote images with machine learning algorithms. The result of this process is used in validation and testing. Most of the countries are using these machines adequately. These machines can be used to Ethiopia environment by changing a few algorithms in the software to identify and classify the Ethiopian currency (ETB). In this research, we presented the results to show an effective scheme for Ethiopian banknote recognition. We used the Convolutional Neural Network (CNN) architecture-based banknote recognition algorithms along with the embedded system hardware to verify and classify the Ethiopian banknotes denominations.

1.2. STATEMENT OF THE PROBLEM

In our country, Ethiopia, there are different public services that are being provided by government or organizations. These include; public transportation services, Banking Services and Energy meter billing system to mention some. When closely look these services there are problems related customer satisfaction and wastage of time. We investigate the following problems:

- We have light train services which currently providing transportation service for people in Addis Ababa, but the transportation fee is collected manually and

customers must make queue and waste their time to get tickets. As my information light train service provider is to upgrade transportation fee payment system to smart card technology. At this time, we need Ethiopian banknote detector system that will accept the currency and add to the smart card of the passenger.

- We have bus services (like Anbessa, Sheger, etc) which provide transportation services for a long time for people in Addis Ababa, but transportation fees are collected manually and customers must queue and waste their time to get tickets. Some sources are showing that Sheger bus service provider is to upgrade the fee payment system to smart card technology. Therefore, we need Ethiopian banknote detector system that will accept the currency and add it to the smart card of the passenger.
- Ethiopian Electric Power Corporation (EEPCO) has a smart Energy meter but recharging of smart cards creates long queues in front service provider offices thus the customer gets bored because of wastage of time and strong sunlight. If we have a banknote acceptor machine equipped with Ethiopian banknote detector system, then the user can charge their Energy meter card anywhere where the machine exists.
- We have banks that provide ATMs for customers, but customers also need intelligent ATMs that provide cash deposit to the account.

1.3. OBJECTIVES

1.3.1 General Objective:

The objective of this project is to design and implement an efficient Ethiopian banknote detector system using CNN architecture, embedded hardware system (Raspberry pi B+) and Web based app for user interface.

1.3.2 Specific Objective:

Particularly, this research has the following specific objectives:

- To collect various Ethiopian currencies with different ages and conditions.
- To identify the best architecture model from existing CNN architectures.
- Apply various optimization techniques for CNN architects to identify the best optimization technique.
- To develop Web based app(Software) for User Interface

- To implement a novel architecture with best optimization technique using Embedded system for classifying banknotes.

1.4. RESEARCH CONTRIBUTION

- In this research we analyzed classification of the Ethiopian bank notes using four different CNN architectures. Those are InceptionV3, MobileNet, XceptionNet and Resnet50.
- We applied various optimization techniques like Adam, SGD, RMSProp, Nadam, Adagrad and Adadelta to above mentioned CNN architectures.
- We applied various batch sizes 32, 64 and 128 to understand the performance of hyperparameters of each CNN architecture with different optimization techniques.
- We compared the training loss, training accuracy, validation loss, validation accuracy, training time and prediction time of all these models.
- We measured the performances with regards to Sensitivity, Specificity and Accuracy of different classifiers
- We selected the best model and implemented using embedded hardware system based upon all above these results to classify the banknotes.

1.5. SIGNIFICANCE OF THE STUDY

In Ethiopia, smart card recharge for the household and industries for the Electrical Energy meters is a very big challenge. Most of the time, the consumers used to wait in a long queue at the counters. Similar kind of situation exists in public transportation as well to pay for the ticket money. Automatic Smart card recharge systems or automatic ticket-vending machines required for these kinds of applications which requires recognition of banknotes for classification and charging the smart cards automatically. There are various machines used to handle paper currencies, including ATMs, vending machines, transportation ticket-vending machines and so on. In practice, we used to touch and count the notes in almost all the situations while performing transactions. Most of these machines are available at banks as ATMs for cash transactions, bank teller counters, automated vending machines etc., The banknote validation system consists of classification of banknote denominations. As there are few features used to differentiate different denominations, the system designed for this purpose must produce precise outcomes as they involve money. At the same time, the cost involved in the design, implementation and preparation of the

prototype should be low since high end hardware embedded processors like RISC processors are involved. Apart from the design and implementation of Ethiopian banknotes recognition system using best performance CNN architecture on embedded hardware system, we have also come up with the same identification and verification of banknote denominations mechanism by web-based system.

1.6. RESEARCH PROJECT REPORT ORGANIZATION

The research project report will be organized into five chapters. Chapter one introduces the background of the study, statement of the problem, objectives, significance, beneficiaries, scope, and limitations. Chapter two discussed the existing methods and techniques to authenticate and classify the Ethiopian banknotes and various countries' currencies. Also, discussed extensive literature reviews related to Ethiopia bank note detection and classification systems by improving the main findings of the studies and for knowing the main reasons behind the limitations. Briefly explained the details of different types of image processing techniques and classification models in chapter three. In chapter four explained the proposed Ethiopian banknote classification using pretrained CNN architectures such as InceptionV3, XceptionNet, MobileNet and ResNet50 models. All these architectures used the six optimization techniques with batch sizes of 32, 64 and 128. Experimental setup also explained in this chapter. The obtained results are discussed and shown in chapter five.

CHAPTER-2

LITERATURE REVIEW

2.1. LITERATURE SURVEY

Banknotes have been effectively used in almost all the money transactions like electronic currencies daily. Currency recognition and detection whether original or fake is an important issue in computer vision. Even though various types of digital currency transaction systems exist, cash is still the favoured choice in daily transactions such as vending machines, banks, shopping complexes, railway station counters, automatic teller machines, and forex. Moreover, there may be situations where the monies drawn have fake currencies. This issue is still unavoidable, and the banks are not able to supply the required fake currency detection system in all the branches. [1].

2.2. PRIOR WORK ON ETHIOPIAN CURRENCY DETECTION

The four characteristic features of the banknotes such as the dominant color, the distribution of the dominant color, the hue value, and speeded up robust features (SURF) were extracted as the preferential features of banknotes in order to deduct the counterfeit currencies [2]. The above four features in combination with local feature descriptors were considered in a four-level classification process. During the execution, among the four, one of the four features was extracted as a classification task. The correlation coefficient-based template matching was used for classification. To check the originality of the currencies, final verification tasks were performed by segmenting the thin golden vertical strip which is on the currency denomination of ETB notes of 50, and 100. Test results showed that the proposed design had an average detection rate of 90.42 % for genuine currency, with an average prediction time of 1.68 seconds per banknote (Jegnaw and Yaregal, 2016). Though the researchers have used a suitable method for classification as well as fraud recognition, the techniques suffer precision issues such as classifying old genuine bills whose color become fading. The fading of color of the currencies might be due to hard- and - fast circulation of the currencies by careless peoples. The spatial information of the bank note relates to the color present in the note but not the color density. During the image acquisition process illumination conditions, it is expected that it would not

vary, but the notes may lose their intensities because of the worn-out or they may have dirt.

The bank note recognizing process is sufficient in using RGB space density. The Ethiopian currency may have similar color features because of this it is not able to classify accurately the denomination based on color features. In addition, the template matching classifiers are variant to intensity value change (Gupta and Goswami, 2017).

Some researchers proposed the following methods to enhance the feature extraction of the images (Asfaw Sheferaw and Million Meshesha). SIFT, GLCM, color momentum, CNN and combination of SIFT, GLCM and color momentum techniques. Another researcher proposed and improved the feature extractions using Feed-Forward Artificial Neural Network as a classifier for the design of Ethiopian paper currency recognition system [3].

The major image processing phases such as image acquisition by using a scanner, pre-processing which is responsible to remove noise, convert RGB to grayscale and normalized the size of the input banknote image to reduce the influence of the noise to the recognizer, feature extraction which is responsible to extract the descriptive features from the given banknotes by using Convolutional Neural Network (CNN) model and classification which was performed by using feed-forward artificial neural network classier was followed by this research work. In this research work to train and test the proposed model a total of 2400 banknotes images were collected through the scanner and 70%, 15%, and 15% was used for training, validating and testing respectively.

A CNN based Ethiopian currency recognition model for vision-impaired peoples have been developed in which they used pre-trained models such as Tensorflow Object Detection API and the pre-trained Faster R-CNN (Region-based CNN), Inception, and SSD (Single-Shot multibox Detection) MobileNet models with CNN architecture and tested them in real time scenarios. They predicted the outcomes from the system by applying various input methods such as partially damaged banknotes, fully damaged banknotes and folded currencies. This model showed the average accuracy value of 91.8% which is considered as very less when compared to our proposed method. In this approach, for every 100 transactions, approximately 8 transactions are found to be abnormal. Hence, this approach has not been considered for commercial usage. Moreover, one of the drawbacks of their approach is that they haven’t come up with

the procedures to calculate the total processing time for the prediction of banknotes [4].

2.3. TEXTURE-BASED BANKNOTE RECOGNITION

The traditional approach of paper money recognition is the identification of visible features found on paper currency like the historical people images like Father of nation, freedom fighters etc., landmark buildings like Parliament buildings, numbers, color, texture, signature and size of banknote. These textures are extracted as features using texture-modeling techniques. The disadvantage of this approach is that it is difficult to recognize the old age of banknotes torn due to their regular usage [5]. The researcher predicted the denomination classification based on geometric features of banknotes [6]. This approach is applicable only for countries that have banknote denominations with various sizes like China and Italy. It was unsuitable for US dollars, which had similar size for various denominations. Moreover, most of the country's banknotes' denomination colors are different. In addition, most previous studies on banknote detection using deep learning have used databases with simple backgrounds or with the application of a slight rotation such that the objects can be easily recognized. Thus, the studies that examine the detection performance using the images captured in various conditions are lacking [7]. The color of Indian paper currencies is different for all denominations. Using these various colors, the author developed a Indian banknote denomination classification [8]. Similarly, another researcher also investigated the currency recognition system using color conversion techniques. The researcher extracted the color features of Indian currencies by using HSV (Hue, Saturation, Value) color space [9]. Furthermore, a researcher urged on extracting the Euro currency note features using unique characteristics of image parts. Similarly, contributed research on features extraction techniques using coordinate data size possesses the same characteristics of image color [10]. Likewise, a new approach called the visible light spectrum method is employed for classification of Korean Won notes. The wavelet domain features of Korean Won bill images were extracted using the Sobel filter. Additionally, to reduce the features of image space coordinates employed the canonical analysis. Finally, it is required to send the output image to the K-nearest neighbor method to recognize the Korean won currency notes [11].

2.4. COUNTRY SPECIFIC BANKNOTE RECOGNITION

Modern countries have been investigating to develop innovative technologies for authenticating their banknotes. A Canadian dollar scanner has been developed with pre-recorded voice output. The user can keep the banknote under the scanner a pre-recorded voice output informing the denomination value. This scanner works only for Canadian dollars [12]. A Sri Lankan (LKR) rupee classification system has been designed by utilizing a unique linear transformation function, edge detection and back propagation Neural Networks [13].

Likewise, [14], banknote recognition was conducted using AdaBoost and SURF-based methods. However, detection and classification processes using SURF are inefficient for finding the desired objects in images with complicated backgrounds or environments. The reason is that the SURF algorithm involves transforming color images into grayscale images to find features; consequently, important information is lost when color images captured by smartphone cameras are used. Moreover, even when the object and background in color images are different, SURF transforms the image into grayscale and recognizes the object and background as similar features, which leads to false detection of objects.

The front side and backside of Myanmar currencies (kyats) in three denominations with image processing techniques are applied in [15]. For the purpose of feature extraction, Zernike moments were used, and classification is done using the k nearest neighbor algorithm. The neural network is also used in [16] to solve these kinds of problem for visually impaired. Their study suggests that cognition frameworks and neural activities can result in more significant research to do these kinds of tasks.

Furthermore, an image processing-based Bangladesh bills recognition technique has been designed in which the researcher extracted the features from Latent Image, watermark, and micro-printing of the bill using Histogram of Gradient (HOG) method. To classify the Bangladesh bills, they utilized the SVM (super Vector Machine) algorithm. This design was found to provide 100% of classification accuracy. However, the designed system tested only on the 1000 and 500 Bangladesh banknote denominations [1]. In the same view, the portable system for blind people for Euro banknotes detection and recognition is presented in [17]. Finally, a researcher presented a currency recognition system for six different types of US dollars in which the banknote images are processed by utilizing the line sensor. To extract the banknote image features adopted the Principal Component Analysis (PCA)

algorithm and Linear Vector Quantization (LVQ) network was utilized for classification [18].

2.5. FEATURE DESCRIPTOR BANKNOTE RECOGNITION

Several researchers have recommended server-client model based online banknote recognition systems for Slovak EURO currency. This model employed a SIFT detector. However, this model had a problem with long recognition time [19]. The proposed task was made to handle small data problems and unable to handle real-time processing due to high computation. A currency recognition system for Ethiopian Banknotes using a support vector machine is discussed in [20], which recognized the front part of the currency well. It was not trained on backside of the banknotes. Another researcher proposed an Indian banknote detection method using side invariant technique. In this technique, the system used a template matching method. It compared the edges of the input test images and reference images from the database. Finally, the system classified the paper currency depending on the generated match score. Due to the limitations of the proposed system, it showed a low accuracy rate because of the positioning of the input currency image, input image brightness and image quality system [21]. Furthermore, a Chinese Yuan Reminbi currency classification employing a robust method called Block Local-Binary pattern was investigated by researchers [22]. Similarly, a multi-currency detection system for Chinese Yuan and Euro paper currencies has been designed and tested where they followed the Generalized Gaussian Density (GGD), Quaternion Wavelet Transform (QWT) and a Back Propagation Neural Network (BPNN) to detect the multi-currency notes [23]. Additionally, a Persian banknote detection is projected by using Haar Wavelet Transform (HWT) and Multilayer Perceptron (MLP) neural network. The benefit of HWT is as a change in the input signal which can lead to an unpredictable change in the transform coefficient. The drawback of this method is that it generates a spatial-domain feature in geometrical positions [24]. Similarly, a new approach, grid features of image, was invented to classify the banknote images. In this approach, the banknote image feature vector is calculated by using the average gray value of partitioned m x n divisions of the input image. The system also had other flaws such as determining the best m x n partition size and number of features for efficient classification [25]. The banknotes classification for Turkish lira using deep

convolutional neural networks that is implemented and trained on the DenseNet-121 architecture [26]. Finally, the SVM machine learning algorithm based Nigerian currency classification system was proposed and used a genetic algorithm for parameter optimization. A predefined decision function was applied in the SVM machine learning algorithm. However, an efficient set of features was not calculated for Nigerian banknote classification. Further, the system wasn't evaluated using other global banknotes [27].

2.6. MOBILE BASED BANKNOTE RECOGNITION

Globally, various countries' banknote recognition systems have been designed to help visually impired persons. Few of these systems are LookTel Money Scanner and IDEAL Currency Identifier to predict any denomination of US dollars. Apart from this, there are dedicated mobile readers namely Money Talker, Note TEller 2 and K-NFB designed for Australian and US dollar currency recognition [28]. A mobile phone-based US banknote detection system was employed with background subtraction and perspective correction algorithms. These banknote images are trained using an efficient Ada-boost algorithm. This approach works on various mobile operating systems like Windows and Symbian. However, though the mobile phone-based system used the efficient algorithms, they took more processing time which needed to be improved for real time scenarios [29]. Similarly, a technique proposed for determining the US Dollar employed an image identification approach called Eigenfaces based on the Principal Component Analysis (PCA). The system analysed the banknote images on two different Nokia Smartphones. It is found that the system had a precision rate of 99.8%. The system had limitations such as image brightness and segmenting the background of banknote images [30]. Correspondingly, a bionic eyeglass was implemented to recognize the banknotes. Bionic eyeglass was interfaced with a Samsung Galaxy S mobile phone and this experiment was tested with five visually challenged persons. This system utilized K-means and CNN architecture for classification of banknote images. Bionic eyeglass has an integrated micro camera to capture the banknote image. After that image is transmitted to the computer system by using server/client architecture for recognition. Here, Samsung mobile acts as an interface device just to send the images to the computer system through the Bionic eyeglass with no processing capability [31]. Finally, a Jordan paper currency recognition system for both colored and gray scaled images with a dataset size of 400

has been developed. This system used to test on both grayscale and color images. The SIFT features were extracted from both colored and grayscale images and compared with the classification rate. However, the system faced the same issues with the illumination difficulty and system was comparatively slow when interfaced on a cell phone [32].

2.7. OPTIMIZATION METHODS

Optimization is one of the core components of machine learning. The essence of most Machine Learning algorithms is to build an optimization model and learn the parameters in the objective function from the given data. In the era of immense data, the effectiveness and efficiency of the numerical optimization algorithms dramatically influence the popularization and application of the machine learning models. A researcher evaluated the performance using Convolution Neural Network (CNN) with RMSprop (Root Mean Square Propagation) Optimization and different filters are used on a standard a UCI Machine Learning repository dataset [33]. A technique the PSO-SGD proposed by Albeahdili has been proved to be superior to the traditional method in training, and this study proposes a new hybrid algorithm, iSSO-SGD, based on that method [34]. A researcher to boost the classification performance, four different optimizers are used with the TF-CNN, Adagrad, Proximal Adagrad, Adam, and RMSProp. Seven classification metrics were measured and the time consumed for each optimizer has been estimated [35].

2.8. RESEARCH GAP

Banknote recognition and classification systems play a vital role in automatic money transaction machines. Most of the reviewed research in the above-mentioned section played a substantial role in the problem area of banknote recognition and classification. Even if they have their limitations/gap such as small datasets, they require special knowledge on the usage of systems and moreover they are found to be expensive. Some of the research needs more processing time to see the results because of more computational algorithms and internal architecture layers. In addition, other researchers are expected to come up with issues like folding, lighting conditions, more processing time, paper currency quality and worn currency notes. Moreover, all the above research works are concentrated on different neural network architectures and not considered the effect of optimization techniques.

Most of the researchers concentrated on various techniques and algorithms but they did not mention the type and specifications of the computer system to run these algorithms efficiently. Though most of the researchers do analysis on high-speed computers there are limitations like the cost and complexity of the system to use and operate. Hence, apart from the software programs, algorithms, and architectures, it is also necessary to implement it on sophisticated embedded systems with light weight architectures in order to provide high processing speed with less validation time.

Hence, there is a need to come up with a development of the best accuracy currency recognition models using CNN classifier with a good optimization technique on sophisticated embedded systems. Therefore, in this research project we designed and implemented an efficient Ethiopian banknote detector system using CNN architecture, embedded hardware system (Raspberry pi B+) and Web based app for user interface.

CHAPTER-3

METHODOLOGY

3.1. IMAGE PREPROCESSING

In recent times, the Convolutional Neural Networks have become the most powerful method for image classification. Various researchers have shown the importance of network architecture in achieving better performances by making changes in different layers of the network. Some have shown the importance of the neuron's activation by using various types of activation functions. Raw data if applied to any classification methods does not produce good accuracy. The preprocessing techniques play a vital role in achieving the state of art on any dataset. The aim of this process is to improve the image data (features) by suppressing unwanted distortions and enhancement of some important image features so that our classification models can benefit from this improved data to work on. Steps for image pre-processing includes reading image, resizing image, and data augmentation.

Data augmentation in machine learning refers to the techniques that synthetically create new examples from a data set by applying possibly stochastic transformations on the existing examples. In the image domain, these transformations can be, for instance, slight translations or rotations, which preserve the perceptual appearance of the original images, but significantly alter the actual pixel values.

We performed the experiments on the Ethiopia bank note image data sets. We resized the 1.3 M images into 227 x 227 pixels, as a compromise between keeping a high resolution and speeding up the training.

3.1.1. Need for Image-Preprocessing

Computers are able to perform computations on numbers and is unable to interpret images in the way that we do. We have to somehow convert the images to numbers for the computer to understand. The aim of pre-processing is an improvement of the image data that suppresses unwilling distortions or enhances some image features important for further processing.

3.2. STEPS FOR IMAGE PRE-PROCESSING

1. Read image
2. Resize image
3. Data Augmentation

a. Gray scaling of image
b. Reflection
c. Gaussian Blurring
d. Histogram Equalization
e. Rotation
f. Translation

Step 1: Reading Image

In this step, we simply store the path to our image dataset into a variable and then we create a function to load folders containing images into arrays so that computers can deal with it.

Step2.Resize image

Some images captured by a camera and fed to our AI algorithm vary in size, therefore, we should establish a base size for all images fed into our AI algorithms by resizing them.

Step3. Image Data Augmentation:

Image data augmentation is an integral process in deep learning, as in deep learning we need large amounts of data. Some cases, it is not feasible to collect millions of images, so data augmentation comes to the rescue. It helps us to increase the size of the dataset and introduce variability in the dataset. It is a technique of applying different transformations to original images which results in multiple transformed copies of the same image. Each copy, however, is different from the other in certain aspects depending on the augmentation techniques you apply like shifting, rotating, flipping, etc.

Applying these small amounts of variations on the original image does not change its target class but only provides a new perspective of capturing the object in real life. These image augmentation techniques not only expand the size of your dataset but also incorporate a level of variation in the dataset which allows model to generalize better on unseen data. Also, the model becomes more robust when it is trained on new, slightly altered images.

3.3. TRAIN TIME IMAGE AUGMENTATION

Augmenting the images just before training is one of the most common approaches. Train time image augmentation works well but there is a catch to it. The image dataset actually does not expand. Rather the augmented images replace the original images.

And then these augmented images are used for training. So, our neural network model does not train on a greater number of images. It trains on the same number of images. It's just that each of the images is augmented. Depending on the augmentation methods used, the images may be scaled, shifted, or flipped. Instead of the original images, these augmented (but same amount) of images are used for training.

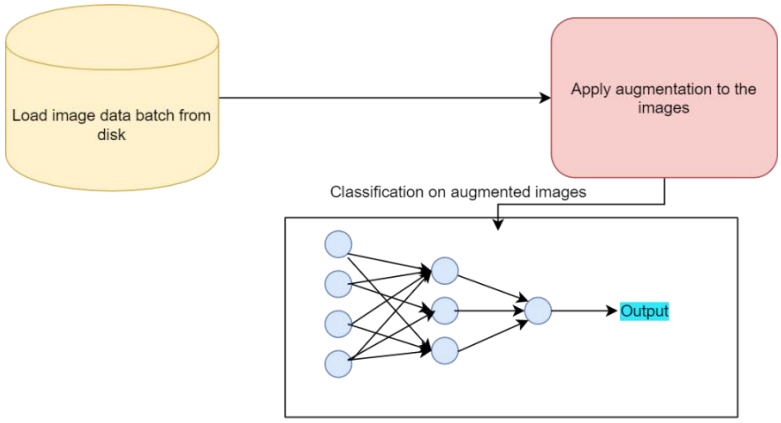

Figure 3.1 Image showing the working of train time augmentation in deep learning

Now, augmenting the images brings some variety to the dataset and the neural network model gets to see different types of those images. This helps as it increases the generalization power of the neural network model.

One of the other, less used, yet highly effective methods is expanding the image dataset using image augmentation. In this method, we use the original images as well as the augmented images for training. So, while training the neural network model will get to see the original images and the augmented images. Using this method, we can increase the size of the image dataset substantially.

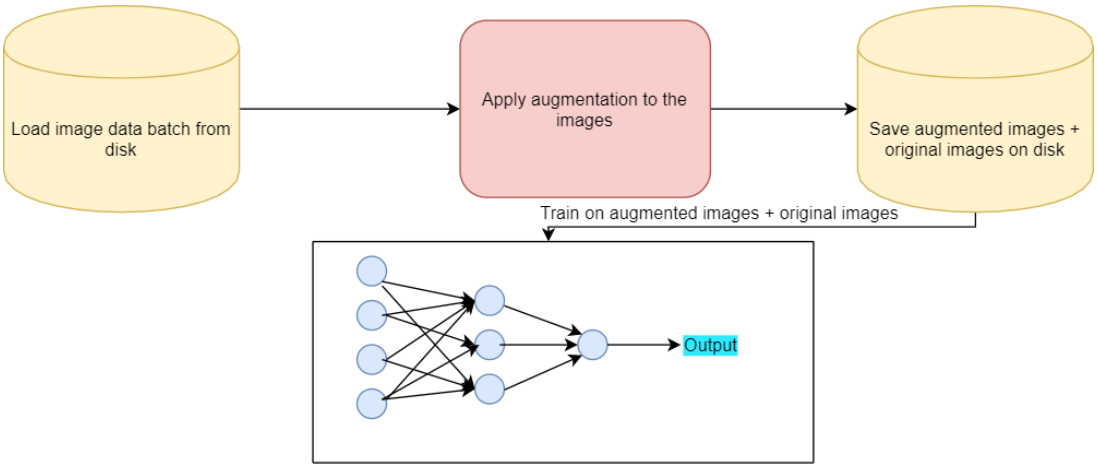

Figure 3.2 Training a deep neural network on both augmented images and the original images

Also, when used correctly, each image can be made to look very different from the original images. Using a combination of flipping, scaling, rotating, and shifting usually yields the best results.

3.4. DETAILS OF IMAGE CLASSIFICATION TECHNIQUES

We will start with some statistical machine learning classifiers like Support Vector Machine and Decision Tree and then move on to deep learning architectures like Convolutional Neural Networks. To support their performance analysis, the results from an Image classification task used to differentiate lymphoblastic leukemia cells from non-lymphoblastic ones have been provided. The features have been extracted using a convolutional neural network, which will also be discussed as one of our classifiers. This is because deep learning models have achieved state of the art results in the feature extraction process.Different classifiers are then added on top of this feature extractor to classify images.

3.4.1. Support Vector Machines

Support vector machines (SVM) are powerful yet flexible supervised machine learning algorithms which are used both for classification and regression. Support vector machines have their unique way of implementation as compared to other machine learning algorithms. They are extremely popular because of their ability to handle multiple continuous and categorical variables. Support Vector Machine model is basically a representation of different classes in a hyperplane in multidimensional space. The hyperplane will be generated in an iterative manner by support vector machine so that the error can be minimized. The goal is to divide the datasets into classes to find a maximum marginal hyperplane. It builds a hyper-plane or a set of hyper-planes in a high dimensional space and good separation between the two classes is achieved by the hyperplane that has the largest distance to the nearest training data point of any class. The real power of this algorithm depends on the kernel function being used. The most commonly used kernels are linear kernel, gaussian kernel, and polynomial kernel.

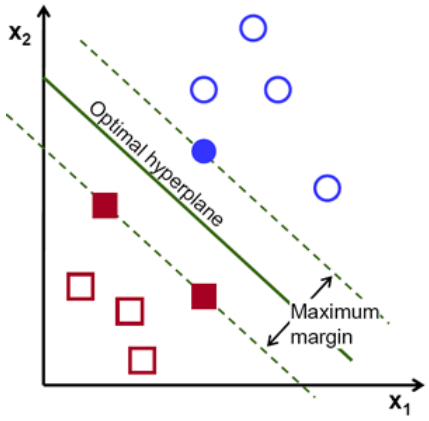

Figure 3.3 Super vector machine Possible hyperplanes

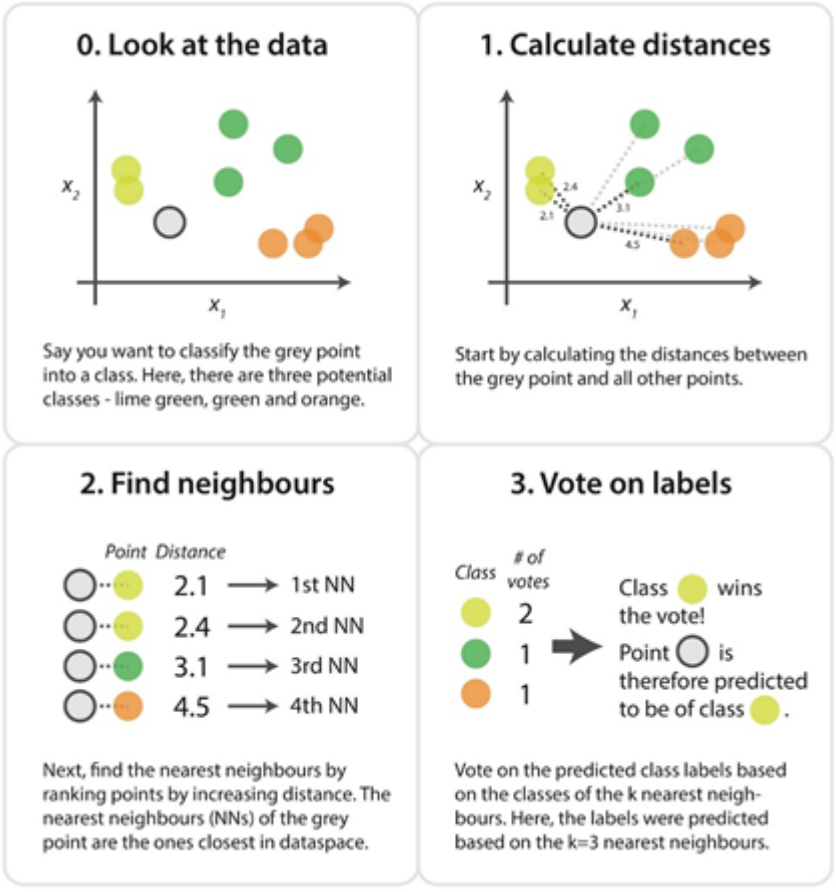

Figure 3.4 K-Nearest Neighbor training examples

3.4.2. K-Nearest Neighbor

K-Nearest Neighbor is a non-parametric method used for classification and regression. In both cases, the input consists of the *k* closest training examples in the feature space. It is by far the simplest algorithm. It is a non-parametric, lazy learning algorithm, where the function is only approximated locally and all computation is deferred until function evaluation. This algorithm simply relies on the distance between feature vectors and classifies unknown data points by finding the most

common class among the k-closest examples. In order to apply the k-nearest Neighbor classification, we need to define a distance metric or similarity function, where the common choices include the Euclidean distance and Manhattan distance. The output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor. Condensed nearest neighbor (CNN, the Hart algorithm) is an algorithm designed to reduce the data set for K-Nearest Neighbor classification.

3.4.3. Artificial Neural Networks

Inspired by the properties of biological neural networks, Artificial Neural Networks are statistical learning algorithms and are used for a variety of tasks, from relatively simple classification tasks to computer vision and speech recognition.

ANNs are implemented as a system of interconnected processing elements, called nodes, which are functionally analogous to biological neurons. The connections between different nodes have numerical values, called weights, and by altering these values in a systematic way, the network is eventually able to approximate the desired function.

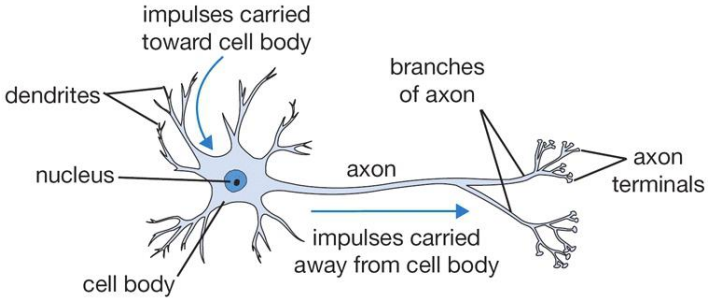

Figure 3.5 Single biological neuron

The hidden layers can be thought of as individual feature detectors, recognizing more and more complex patterns in the data as it is propagated throughout the network. For example, if the network is given a task to recognize a face, the first hidden layer might act as a line detector, the second hidden takes these lines as input and puts them together to form a nose, the third hidden layer takes the nose and matches it with an eye and so on, until finally the whole face is constructed. This hierarchy enables the network to eventually recognize very complex objects.

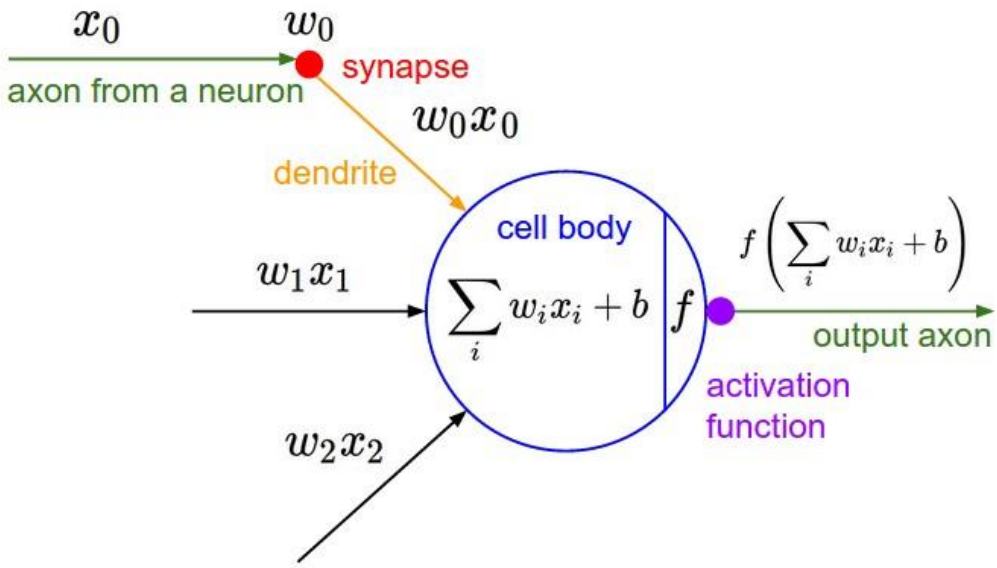

Figure 3.6 Basic structure of an artificial neuron

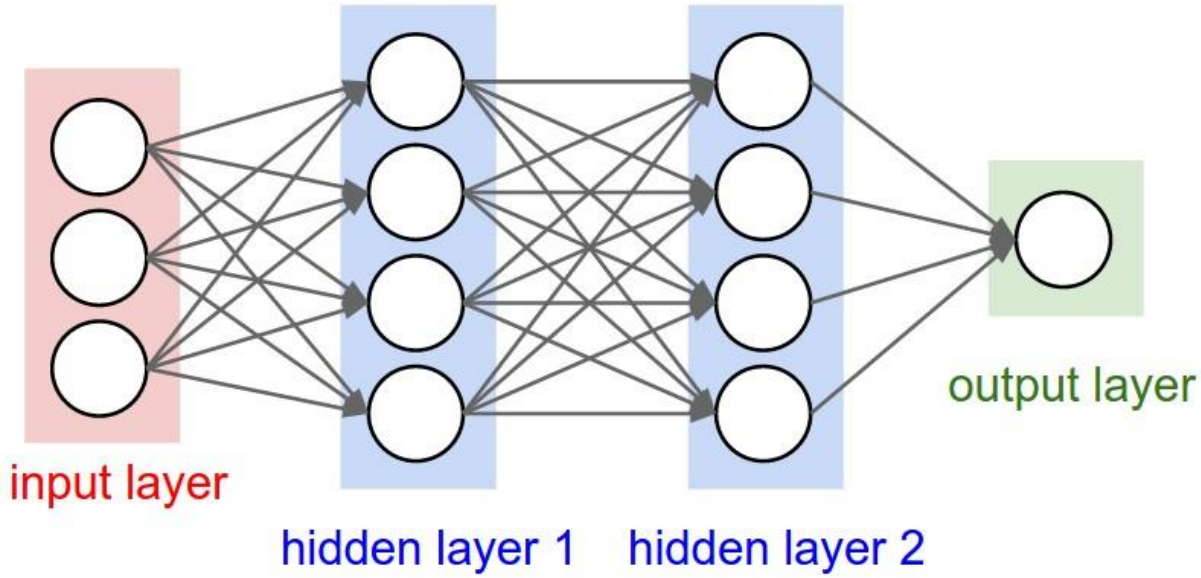

Figure 3.7 Architecture of an artificial neuron

3.4.4. Convolutional Neural Network

Convolutional Neural Network (CNN, or ConvNet) are a special kind of multi-layer neural networks, designed to recognize visual patterns directly from pixel images with minimal pre-processing. It is a special architecture of artificial neural networks. Convolutional neural network uses some of its features of visual cortex and have therefore achieved state of the art results in computer vision tasks. Convolutional neural networks are comprised of two very simple elements, namely convolutional layers and pooling layers. Although simple, there are near-infinite ways to arrange these layers for a given computer vision problem. The elements of a convolutional

neural network, such as convolutional and pooling layers, are relatively straightforward to understand. The challenging part of using convolutional neural networks in practice is how to design model architectures that best use these simple elements. The reason why convolutional neural network is hugely popular is because of their architecture, the best thing is there is no need of feature extraction. The system learns to do feature extraction and the core concept is, it uses convolution of image and filters to generate invariant features which are passed on to the next layer. The features in next layer are convoluted with different filters to generate more invariant and abstract features and the process continues till it gets final feature/output which is invariant to occlusions. The most commonly used architectures of convolutional neural network are LeNet, AlexNet, ZFNet, GoogLeNet, VGGNet, and ResNet.

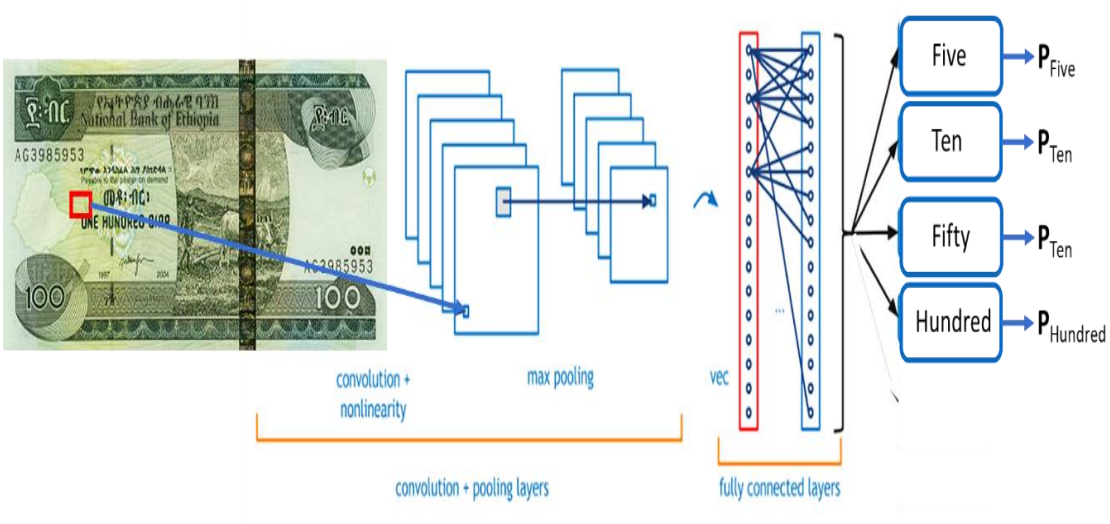

Figure 3.8 Convolutional Neural network working flow

3.5. ETHIOPIA BANKNOTE CLASSIFICATION USING PRETRAINED CNN ARCHITECTURES

Deep convolutional neural network has greatly promoted the development of computer vision, especially in object recognition, object detection and semantic segmentation. CNN has a very special structure in a deep layered network composed by several different functional neurons. They are the convolutional layer, pooling layer and fully connected layer. CNN can be utilized with a pretrained feature set as an initial weight vector. The CNN basic structure is described in Figure 3.9.

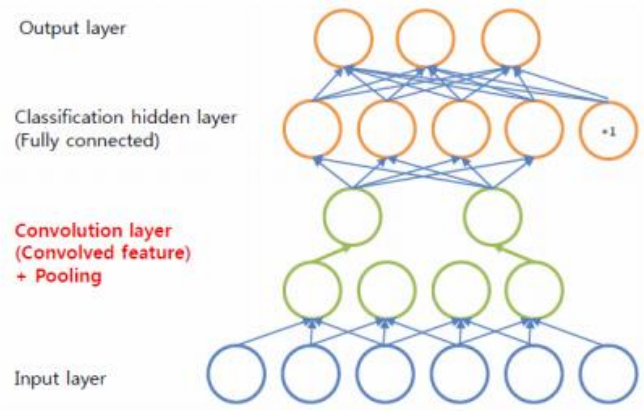

Figure 3.9 Basic structure of Convolutional Neural Network

At the convolutional layer, the input image is transferred into convolved and overlapped feature space. At the pooling layer, the result of the convolutional layer is aggregated into the pooled adjacent feature. Activation of the convolutional layer followed by the pooling layer is fed to the fully connected layer, which is composed of softmax regression or feedforward neural network. Between the pair of convolutional and pooling layers, there can be an additive pair of convolutional and pooling layers, repeatedly.

If a N x N input image is connected to the convolution layer and a pretrained feature (convolution kernel) size of m x m, the size of the convolved layer is (N - m + 1) x (N - m + 1). The activation of the convolution layer is calculated by Eq. (3.1).

$$For\ i,j\ =\ 0,1\ \ldots..N-m$$

$$z_{ij}^{(2)}\ =\ \sum_{u=0}^{m-1}\sum_{v=0}^{m-1} w_{uv}^{(1)}\ x_{(i+u)(j+v)}^{(1)} \quad \text{---------- Eq. (3.1)}$$

where w means the convolution weight vector, x means the input image. Pooling layer aggregates result of the convolution layer by averaging the image area of a k x k size.

In that case, there are (N / k) x (N / k) of pooling layer neurons. This is described in Figure. 3.10.

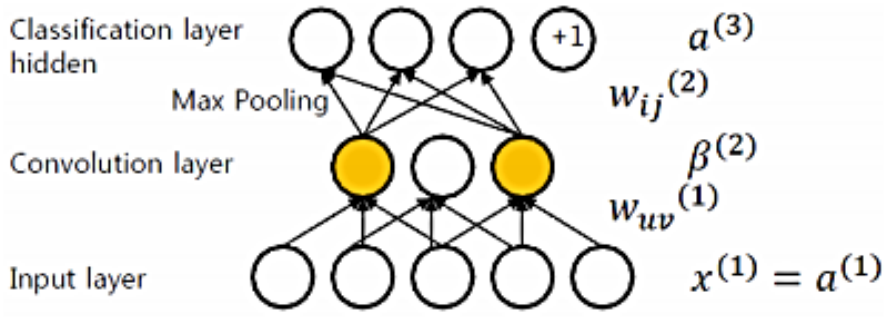

Figure 3.10 Structure of CNN Pooling Layer

The pooling layer is calculated using Eq. (3.2).

$$\beta_j^{(2)} = MaxPoop(a^{(2)})$$
$$z_i^{(3)} = w_{ij}^{(2)}\beta_j^{(2)} + b_i \quad \text{---------- Eq. (3.2)}$$
$$a_i^{(3)} = f(z_i^{(3)})$$

The pooling layer is followed by a fully connected layer or another convolutional layer. A fully connected layer is a simple softmax regression. Figure 3.11 describes the complete structure of a convolutional neural network with hidden layers.

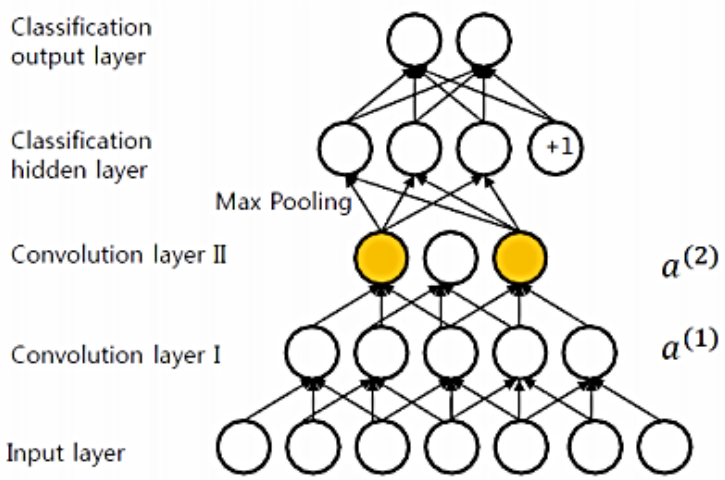

Figure 3.11 Complete Layer structure of the CNN

Network architecture deep CNNs have achieved state-of-the-art performance in various high-level computer vision tasks in recent years. However, these networks often have millions of parameters and require extensive computational resources. Four

pretrained CNN architecture models were investigated in this study to classify Ethiopian banknote denominations and find the fake note. Those network architectures are: Inception-v3, MobileNet-v2, XceptionNet and ResNet-50 models. These pretrained networks were trained on more than a million images from the ImageNet database. These pretrained networks can classify images into 1000 object categories. As a result, these pretrained networks have learned rich features representing a wide range of images. The properties of these CNN architectures are described in Table 3.1.

Table 3.1. Properties of proposed pretrained CNN Models

Model	Size	Parameters (Million)	Depth
InceptionV3	92 MB	23.8	159
MobileNet	16 MB	4.2	88
XceptionNet	88 MB	22.9	126
ResNet50	98 MB	25.6	_ _

3.6. CONTRIBUTION OF THE RESEARCH WORK

Deep learning techniques, particularly convolutional neural networks (CNNs), have proved their success and popularity recently in many fields, especially distinguishing and analysing image hyperparameters. Motivated by this direction, our work attempts for the first time to investigate the application of a state-of-the-art deep learning technique on genomic sequences to classify Ethiopia banknotes denominations. The research work focuses on three contributions based on image preprocessing and classification. The contributions are summarized as follows:

- We consider four CNN architectures, namely, InceptionV3, MobileNet, XceptionNet and ResNet50 to use feature extraction and parameter-tuning to identify and classify banknotes.
- Applied six optimization techniques SGD, Adam, RMSProp, Nadam, Adagrad and Adadelta to enhance the performance.
- Chosen the best fit model implemented on the embedded platform.

3.7. IMPLEMENTATION METHOD

This part is to propose the methodology to determine the best CNN model for the real-time Ethiopian banknote classification by iteratively varying the pre-trained

model configurations. The pre-trained model configuration contains three aspects which are the pre-trained model based on the specific dataset, the pre-defined feature extractor network and the number of proposals in the feature extractor network. Figure 3.12 demonstrates the block diagram of proposed model procedure. The figure 3.13 shows retrain stage process of proposed model, which is described below.

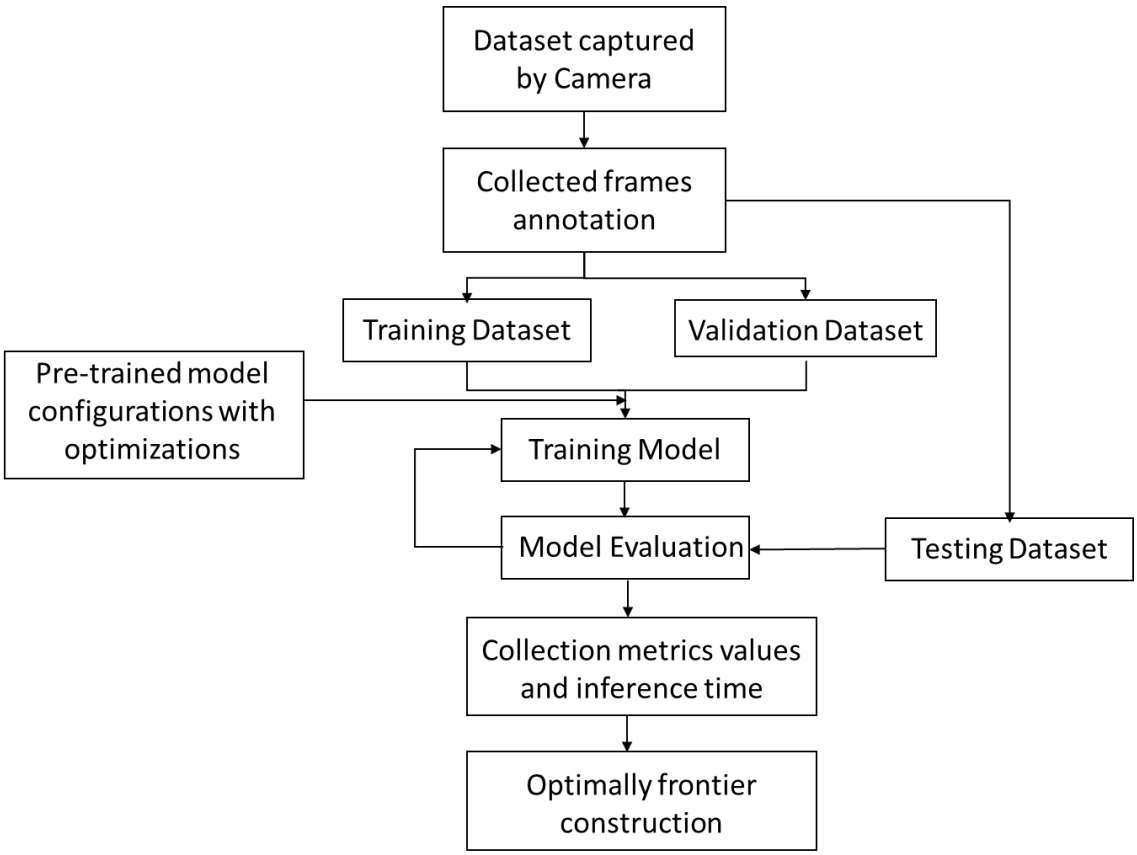

Figure 3.12 Methodology of proposed model

3.7.1. Flowchart of Proposed Methodology

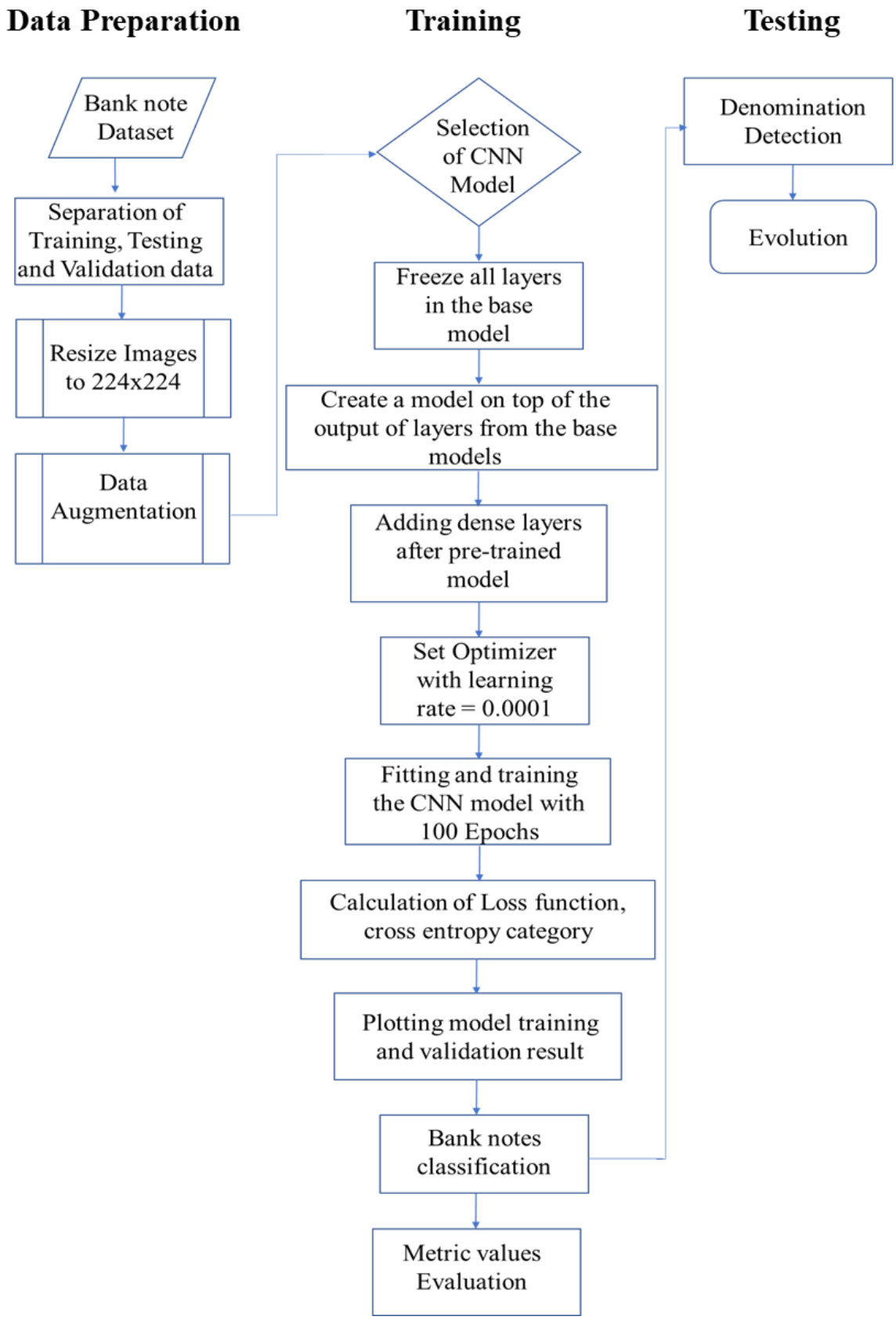

Figure 3.13 Retrain stage process of proposed model

3.7.2. Dataset Captured by camera

The banknote images were captured using a Logitech C920 digital camera with a resolution of 1080 pixel. All banknotes collected from Oromia National Bank, Zero arath branch, Adama. The main advantage of this camera is automatic low light correction. Furthermore, the collected images presented the number of new, old, full, and partial notes utilized in this research. Finally, the dataset was made up of images captured in two directions for each banknote: the front side and the backside. Normally, some preprocessing techniques, e.g., augmentation, image noise elimination, are required to enhance the dataset and accordingly contribute to the performance enhancement of the CNN classification approach.

3.7.3. Preparing Dataset

This collected dataset contains four denominations/classes such as five, ten, fifty and hundred demonstrated sample images in figure 3.14.

Figure 3.14 Sample images of Ethiopia currency collected from bank

Images of bank notes are in RGB format and there are stored in Joint Photographic Experts Group (JPG) format. They possess some challenges such as blurry images, partially occluded signs, low resolution, and poor illumination Furthermore, different dimensions. These images are filtered and renewed size into 500 × 500 pixels.

3.7.4. Splitting Data

Splitting dataset is essential for an unbiased evaluation of prediction performance. In most cases, it's enough to split your dataset randomly into three subsets. The training dataset is applied to train, or fit, model. Use the training set to find the optimal weights, or coefficients, for neural networks. The validation dataset is used for unbiased model evaluation during hyperparameter tuning. For each considered setting of hyperparameters, fit the model with the training set and assess its performance with the validation set. The test dataset is needed for an unbiased evaluation of the final model. Shouldn't use it for fitting or validation.

The main idea behind splitting data into different sets is to evaluate whether or not the trained model is generalized on unseen samples. If the number of samples in the dataset is very high and they are diverse, splitting data with ratio of train dataset 70%, validation dataset10%, test dataset 20% is a good choice. Table 3.2 represents the splitting ratio of Ethiopia banknotes.

Table 3.2 Splitting the bank note Datasets

Type of currency	**Training**	**Validation**	**testing**
Fifty	1430	231	427
Five	1534	238	449
Hundred	1393	219	419
Ten	1345	221	427
Total	**5702**	**909**	**1722**
Total Images: 8,333			

3.7.5. Augmenting Dataset

Assume a training image $\mathbf{x}_i \in R^{H\times W\times 3}$. To human eye, the class of any sample $\mathbf{x}_j \in R^{H\times W\times 3}$ where $||x_i - x_j|| << \varepsilon$ is exactly the same as $\mathbf{x_i}$. However, in the case of ConvNets these slightly modified samples might be problematic. Techniques for augmenting a dataset try to generate several $\mathbf{x}_i$ for each sample in the training set X_{train}. This way the model will be able to be adjusted not only on a sample but also on its neighbors. In the case of datasets composed of only images, this is analogous to

slightly modifying $\mathbf{x}_i$ and generating $\mathbf{x}_j$ in its close neighborhood. Augmenting dataset is important since it usually improves the accuracy of the ConvNets and makes them more robust to small changes in the input. In short, Data Augmentation is a preprocessing technique to help us to generate more image variations to both avoid overfitting and increase accuracy of the model. Here is the index of techniques we used in research:

a) Random Rotations

Image rotation helps our model to become more robust to the changes in the orientation of bank notes. The information of the image remains the same, for example, A ten-birr note is a ten-birr note even if we see it from a different angle. Image rotation is one of the widely used augmentation techniques and allows the model to become invariant to the orientation of the object. Image Data Generator class allows to randomly rotate images through any degree between 0 and 360. When the image is rotated, some pixels will move outside the image and leave an empty area that needs to be filled in. To fill this empty area used nearest pixel values. Here we rotated images into 45^{o}.

b) Random Shifts

All the bank currency notes are may not always be in the center of the image. To overcome this problem, we shifted the pixels of the image either horizontally or vertically. This is done by adding a constant value 0.2 to all the pixels. Here float value is indicating the percentage of width or height of the image to shift.

Figure 3.15 Output images of Image data augmentation techniques rotation, shifts and flips

c) Random Flips

Random flip technique should be according to the object in the image. The pretrained network will be trained on patches of images which are randomly flipped from the original dataset. So here we flipped the bank notes into horizontally.

d) Random Sampling

In random sampling, samples are selected using uniform distribution. Specifically, all samples have the same probability to be assigned to one of the sets without replacement. This method is not deterministic meaning that if we run the algorithm 10 times, we will end up with 10 different training, validation, and test sets. The easiest way to make this algorithm deterministic is to always seed the random number generator with a constant value.

e) Random Crop

Another effective way for augmenting a dataset is to generate random crops for each sample in the dataset. This may generate samples that are far from each other in the input space, but they belong to the same class of object. This is a desirable property since it helps to cover some gaps in the input space. However, in some cases you may develop a special python layer for our dataset or may want to store random copies of each sample on disk.

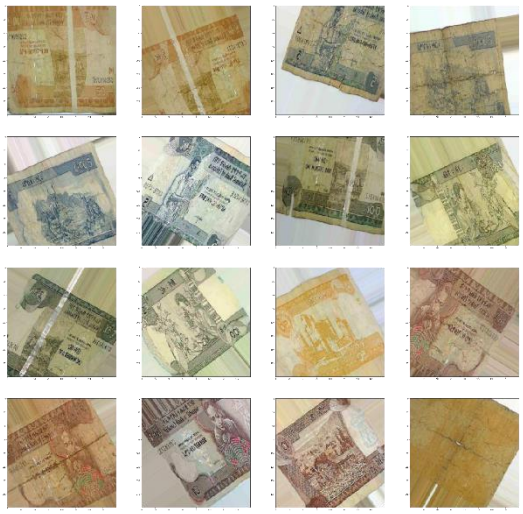

Figure 3.16 Random crop images generation

f) Resizing

In order to simulate the images taken from a distant object, we can resize a sample with a scaler factor less than one. This way, the size of image will be reduced. Likewise, a sample might be upscaled using interpolation techniques. Moreover, the scale factor along each axis might be different but close to each other. Augmenting

datasets with this technique is also a good practice. Especially, if the number of low-resolution images is low, we can simply augment them by resizing high-resolution images with a small scaler factor.

g) Dropout

The last technique that we explain in this section is to generate noisy samples by randomly zeroing some of pixels in the image. This can be done using two different ways. The first way is to connect a Dropout layer to an input layer in the network definition. Alternatively, this can be done by generating a random binary mask using the *binomial* distribution and multiplying the mask with input image. Here we implemented the dropout layer in the input layer.

A higher number of images were needed to train a deep learning algorithm. The data augmentation technique was used to increase the number of samples in the dataset. After the data augmentation, we obtained 30,580 samples.

Table 3.3 List the images quantity after data augmentation generation

Type of currency	**Training**	**Validation**	**testing**
Fifty	4079	320	3227
Five	4290	334	3355
Hundred	4039	322	3176
Ten	4004	304	3130
Total	**16,412**	**1,280**	**12,888**
Total Images: 30,580			

3.8. FEATURE-EXTRACTION

Deep learning systems are layered architectures that learn different features at different layers. These layers are then finally connected to a last layer (usually a fully connected layer, in the case of classification) to get the final output. This layered architecture allows us to utilize a pretrained networks such as Inception V3, MobileNet, XceptionNet and Resnet without its final layer as a fixed feature extractor for other tasks. The following diagram represents deep transfer based on feature extraction:

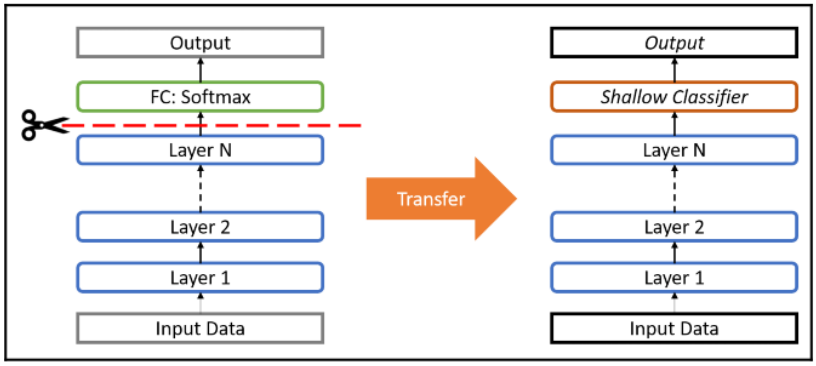

Figure 3.17 feature extraction for the deep transfer learning

For instance, if we utilize InceptionV3 without its final classification layer, it will help us transform images from a new domain task into a 4,096-dimensional vector based on its hidden states, thus enabling us to extract features from a new domain task, utilizing the knowledge from a source-domain task. This is one of the most widely utilized methods of performing transfer learning using deep neural networks.

3.9. FINE-TUNING

This is a more involved technique where we do not just replace the final layer (for classification), but we also selectively retrain some of the previous layers. Deep neural networks are highly configurable architectures with various hyperparameters. The initial layers have been seen to capture generic features, while the later ones focus more on the specific task at hand. Using this insight, we may freeze (fix weights) certain layers while retraining or fine-tune the rest of them to suit our needs. In this case, we utilize the knowledge in terms of the overall architecture of the network and use its states as the starting point for our retraining step. This, in turn, helps us achieve better performance with less training time.

3.10. STATE-OF-THE-ART DEEP IMAGE CLASSIFICATION PRETRAINED MODELS

One of the fundamental requirements for transfer learning is the presence of models that perform well on source tasks. Many of the state-of-the art deep learning architectures have been openly shared by their respective teams. The teams behind those networks have not just shared the results, but also their pretrained models. Pretrained models are usually shared in the form of the millions of parameters/weights the model achieved while being trained to a stable state. Pretrained models are available for everyone to use through different means. The famous deep learning Python library, Keras, provides an interface to download various available pretrained networks, such as Inception, MobileNet, Xception, ResNet and VGG16. Along the

same lines, pretrained models are also available through TensorFlow and other deep learning libraries.

3.10.1. InceptionV3

Inception V3 by Google is the 3rd version in a series of Deep Learning Convolutional Architectures. Inception V3 was trained using a dataset of 1,000 classes from the original ImageNet dataset which was trained with over 1 million training images. Inception v3 is a widely used image recognition model that has been shown to attain greater than 78.1% accuracy on the ImageNet dataset. Inception Network. Szegedy et al. [16] extended the concept of network in network and proposed a modified CNN architecture to achieve improved performance by increasing the depth of the network and keeping the computational cost low.

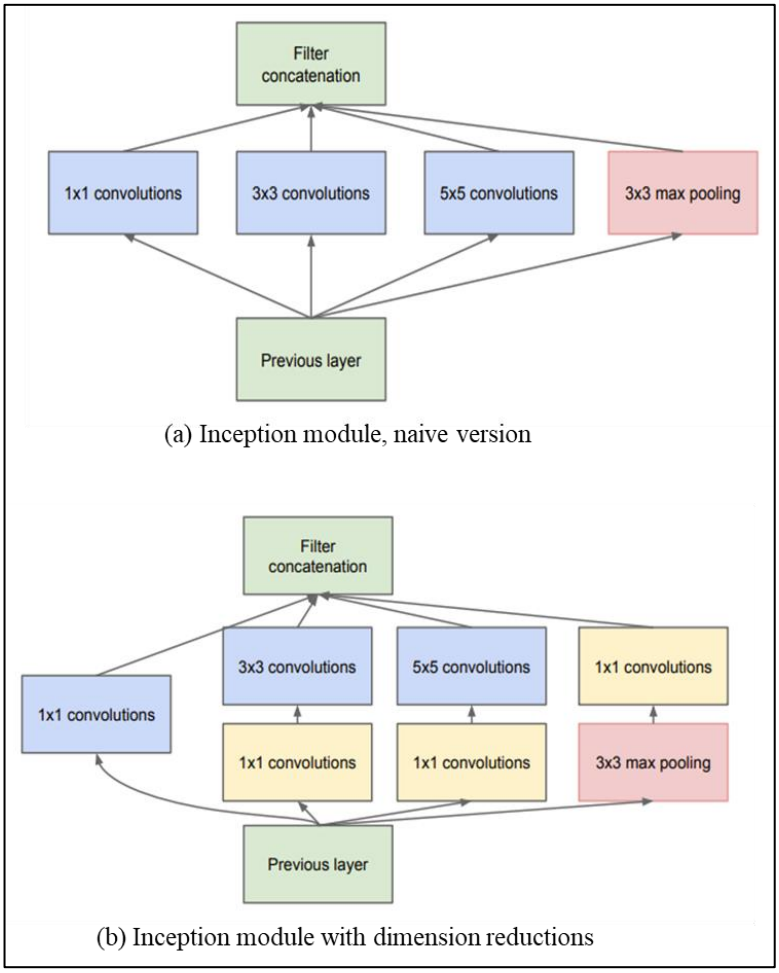

Figure 3.18 Structure of Inception module

The model itself is made up of symmetric and asymmetric building blocks, including convolutions, average pooling, max pooling, concats, dropouts, and fully connected

layers. Batch norm is used extensively throughout the model and applied to activation inputs. Loss is computed via Softmax. In contrast to ResNet, Inception networks proved to be computationally efficient in terms of computing resource utilization as well as the number of parameters. However, the downside of the original inception network was its limited application adaptability in new use cases. Szegedy et al. [14] refined the original inception network model by introducing factorized convolutions with large filter size, factorization into smaller convolutions, and asymmetric convolutions. A high-level diagram of the model is shown below:

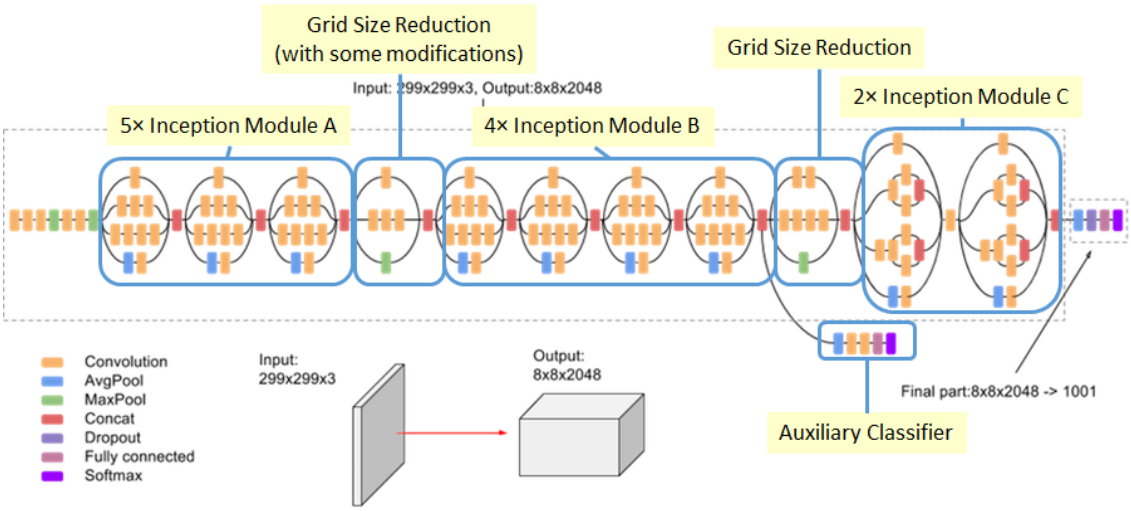

Figure 3.19 High-level diagram of the InceptionV3 model

3.10.2. MobileNet

MobileNet is a CNN architecture model for Image Classification and Mobile Vision. MobileNet is an efficient and portable CNN architecture that is used in real world applications. MobileNet primarily use **depthwise separable convolutions** in place of the standard convolutions used in earlier architectures to build lighter models. MobileNet introduce two new global hyperparameters (width multiplier and resolution multiplier) that allow model developers to trade off **latency** or **accuracy** for speed and low size depending on their requirements. MobileNet is widely used in many real-world applications which includes object detection, fine-grained classifications, face attributes, and localization.

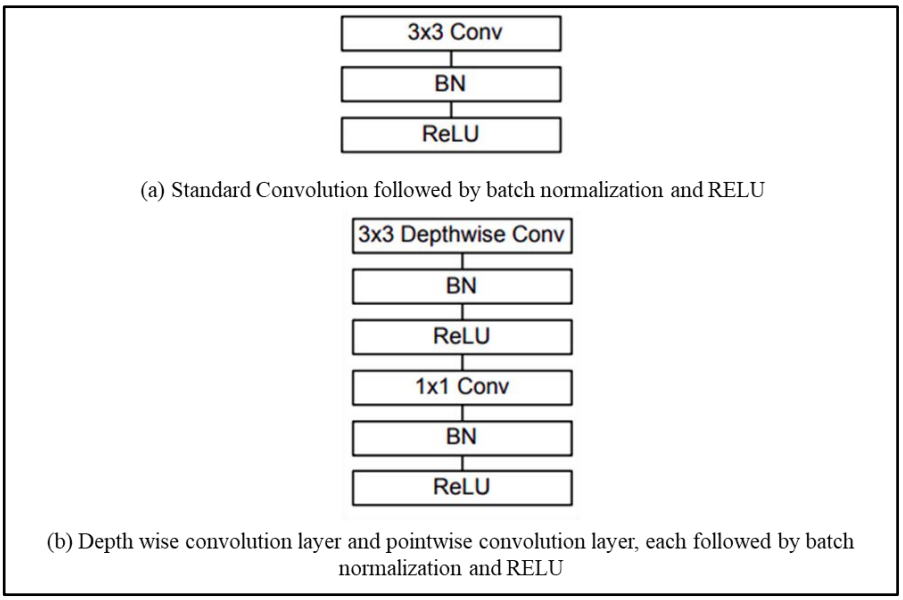

Figure 3.20 MobileNet convolutional layer models

3.10.3. XceptionNet

XceptionNet was proposed by none other than François Chollet himself, the creator and chief maintainer of the Keras library. XceptionNet is an extension of the Inception architecture which replaces the standard Inception modules with depthwise separable convolutions. The difference between InceptionNet and XceptionNet is that, in InceptionNet normal convolutional operations are performed whereas in XceptionNet, Depthwise Separable Convolutional operations are performed. Depthwise Separable Convolutions are different from normal convolutions in a way that, in normal Conv2D layer, for an input of (32, 32, 3) image we can use any number of filters in the Convolutional layer. Each of those filters will be operated over all three channels and the output is the sum of all corresponding values. But in Depthwise separable convolutions, each channel has only one kernel to do convolution. Hence, by performing Depthwise Separable Convolutions, we can reduce the computational complexity as every kernel is of two dimensional only and is convoluting only over one channel. In Keras, we can implement this by using DepthwiseConv2D layer. XceptionNet is comparatively a more efficient model and the weight files produced are also smaller compared to other models.

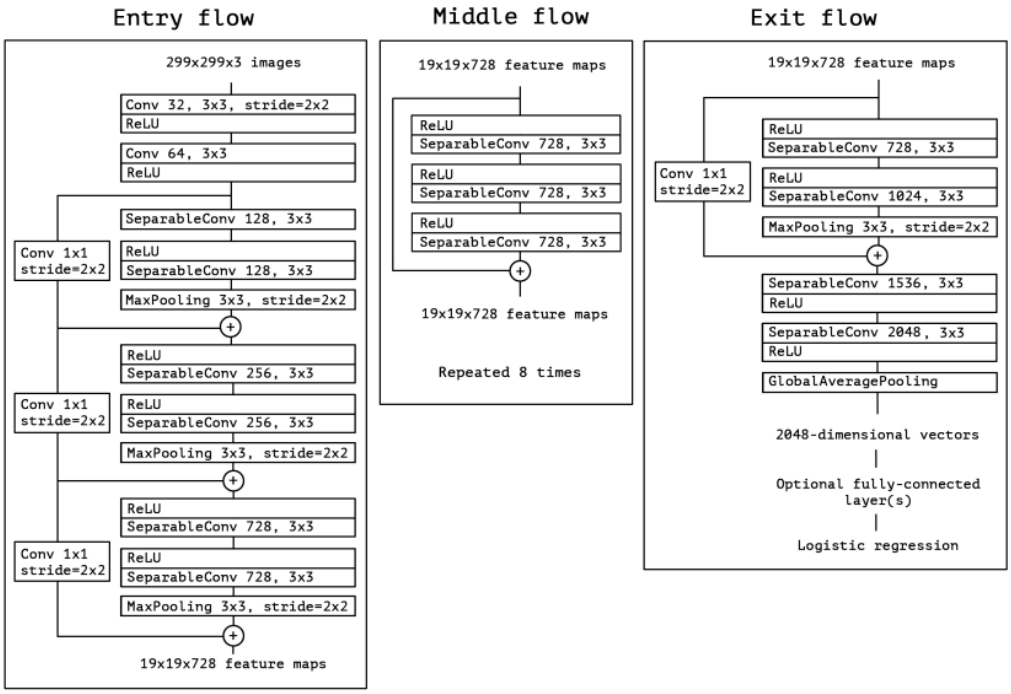

Figure 3.21 The XceptionNet architecture [7].

3.10.4. ResNet-50

Residual Network-50 (ResNet-50) is a deep convolutional neural network to achieve significant results in the classification of ImageNet database [32]. ResNet addressed the problem of training and overfitting in deep neural networks by introducing the concept of residual learning [15]. ResNet-50 is composed of numerous sizes of convolutional filters to reduce the training time and manage the degradation issue that happens because of deep structures. In this model highlighted that as the neural network architecture becomes deeper, degradation occurs. Degradation is the phenomenon of increase in the training error as more layers are added to the architecture of a neural network. To solve the problem of degradation, the authors introduced residual block. Unlike VGG, which adds a stack of convolutional layers followed by a MaxPool layer, ResNet attempts to identify a residual mapping between the input to the convolutional layers and the output at the MaxPool layer, thus eliminating the computational cost of input being processed by the convolutional layer stack.

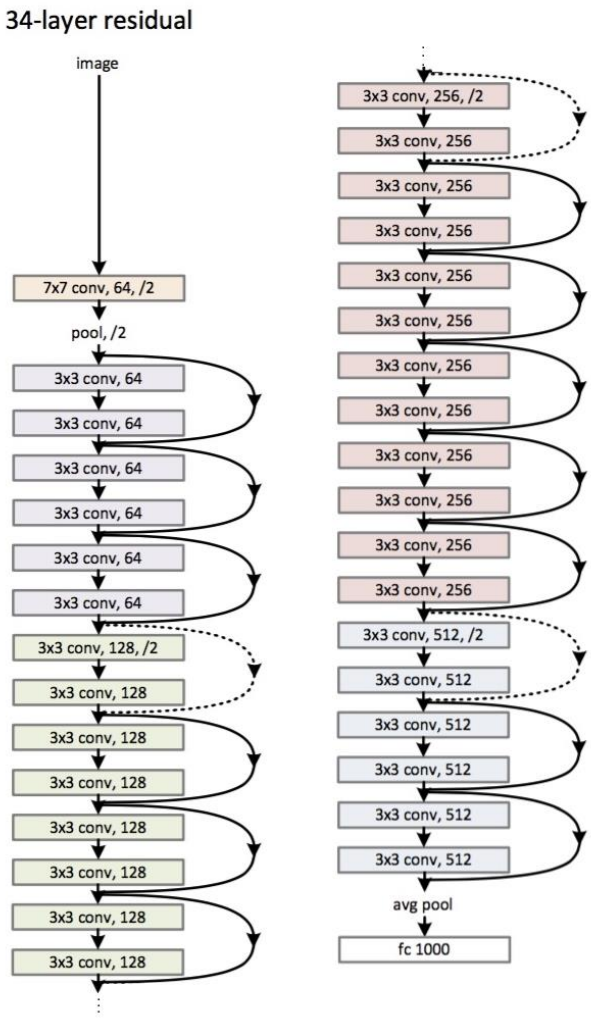

Figure 3.22 Resnet50 architecture module

3.11. EMBEDDED CNN PLATFORM

In this research, we present our embedded inference framework that uses the CPU cores on embedded platforms to execute the lightweight CNNs required by applications. Our inference framework can process lightweight MobileNet [14] on commodity embedded platform at the other hand, designing a lightweight convolutional neural network and deploying it on an embedded platform has been considered as an alternative approach. In spite of the fact that a lot of convolutional neural network are designed for desktop platform, there are many lightweight convolutional neural network [14], [16], [23], [24], specially designed for embedded platform. Forrest N. Iandola et al. used a bottleneck approach to design a very small but efficient network SqueezeNet, which achieves AlexNet-level accuracy on ImageNet with much fewer parameters even ban be compressed to less than 0.5MB, in [15]. MobileNet in [14] is efficient for embedded vision applications by using

depthwise-seprable convolution and it has two simple hyper-parameters that can efficiently trade-off between inference efficiency and accuracy. In [24], MobileNetV2 has been designed and it has some performance improvements on both inference efficiency and accuracy adopting inverted residual structure with linear bottleneck.

The depthwise and wider convolutional neural networks such as InceptionV3, XceptionNet and ResNet50 are having a greater number of layers and requires huge memory size. The InceptionV3 has 22-layer deep architecture having 23.8 million parameters and its size is 92 MB. The depth of InceptionV3 is 159 layers. The XceptionNet architecture has 36 convolutional layers forming the feature extraction base of the network. The 36 convolutional layers are structured into 14 modules, all of which have linear residual connections around them, except for the first and last modules. It contains 126 layers depth in which is having 22.9 million parameters with size of 88 MB. In order to solve the problem of the vanishing/exploding gradient, this architecture introduced the concept called Residual Network i.e., ResNet architecture. The ResNet50 network uses a 34-layer plain network architecture which contains 25.6 million parameters occupying 98MB memory. Compared to all above architectures, the MobileNet has a smaller number of parameters are 4.2 million with size is 16 MB only. MobileNet architecture depth is 88 layers only. Moreover, larger size typically means a larger number of parameters, which makes the enlarged network more prone to overfitting. Hence, in on-device approach, deploying MobileNet lightweight CNN on embedded platform is a best feature.

3.12. OPTIMIZATION TECHNIQUES

Optimization is one of the core components of machine learning. The essence of most Machine Learning algorithms is to build an optimization model and learn the parameters in the objective function from the given data. In the era of immense data, the effectiveness and efficiency of the numerical optimization algorithms dramatically influence the popularization and application of the machine learning models. In order to promote the development of machine learning, a series of effective optimization methods were put forward, which have improved the performance and efficiency of machine learning methods. As the representative of first-order optimization methods, the stochastic gradient descent method [47], [48], as well as its variants, has been widely used in recent years and is evolving at a high speed.

3.12.1. Stochastic Gradient Descent

The batch gradient descent has high computational complexity in each iteration for large-scale data and does not allow online update, stochastic gradient descent (SGD) was proposed [52]. The idea of stochastic gradient descent is using one sample randomly to update the gradient per iteration, instead of directly calculating the exact value of the gradient. The stochastic gradient is an unbiased estimate of the real gradient [52]. The cost of the stochastic gradient descent algorithm is independent of sample numbers and can achieve sublinear convergence speed [49]. SGD reduces the update time for dealing with large numbers of samples and removes a certain amount of computational redundancy, which significantly accelerates the calculation. In the strong convex problem, SGD can achieve the optimal convergence speed [50], [51], [52], [53]. Meanwhile, it overcomes the disadvantage of batch gradient descent that cannot be used for online learning. The loss function can be written as the following Equation

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}(y^i - f_\theta(x^i))^2 \;=\; \frac{1}{N}\sum_{i=1}^{N} cost\,(\theta, (x^i, y^i)) \text{ ---------→ Eqn(3.3)}$$

If a random sample i is selected in SGD, the loss function will be

$$L^*(\theta) = cost\left(\theta, (x^i, y^i)\right) = \frac{1}{2}\,(y^i - f_\theta(x^i))^2 \text{ ----------→ Eqn(3.4)}$$

The gradient update in SGD uses the random sample i rather than all samples in each iteration,

$$\theta' \;=\; \theta + \eta(y^i - f_\theta(x^i))x^i \text{ ----------→ Eqn(3.5)}$$

Since SGD uses only one sample per iteration, the computation complexity for each iteration is O(D) where D is the number of features. The update rate for each iteration of SGD is much faster than that of batch gradient descent when the number of samples N is large. SGD increases the overall optimization efficiency at the expense of more iterations, but the increased iteration number is insignificant compared with the high computation complexity caused by large numbers of samples. It is possible to use only thousands of samples overall to get the optimal solution even when the sample size is hundreds of thousands. Therefore, compared with batch methods, SGD can effectively reduce the computational complexity and accelerate convergence.

However, one problem in SGD is that the gradient direction oscillates because of additional noise introduced by random selection, and the search process is blind in the solution space. Unlike batch gradient descent which always moves towards the

optimal value along the negative direction of the gradient, the variance of gradients in SGD is large and the movement direction in SGD is biased [47].

3.12.2. Adaptive and Adagrad Learning Rate Method

The manually regulated learning rate greatly influences the effect of the SGD method. It is a tricky problem for setting an appropriate value of the learning rate [54], [55], [56]. Some adaptive methods were proposed to adjust the learning rate automatically. These methods are free of parameter adjustment, fast to converge, and often achieving not bad results. They are widely used in deep neural networks to deal with optimization problems. The most straightforward improvement to SGD is AdaGrad [55]. AdaGrad adjusts the learning rate dynamically based on the historical gradient in some previous iterations. The update formulae are as follows:

$$\begin{cases} g_t = \frac{\partial L(\theta_t)}{\partial \theta}, \\ V_t = \sqrt{\sum_{i=1}^{t}(g_i)^2 + \varepsilon}, \\ \theta_{t+1} = \theta_t - \eta \frac{g_t}{V_t} \end{cases} \quad \text{----------} \rightarrow \text{Eqn(3.6)}$$

Where g_t is the gradient of parameter θ at iteration t, V is the accumulate historical gradient of parameter θ at iteration t, and θ is the value of parameter θ at iteration t. The difference between AdaGrad and gradient descent is t that during the parameter update process, the learning rate is no longer fixed, but is computed using all the historical gradients accumulated up to this iteration. One main benefit of AdaGrad is that it eliminates the need to tune the learning rate manually. Most implementations use a default value of 0.01 for η in (Eqn(4.6)). Although AdaGrad adaptively adjusts the learning rate, it still has two issues.

i) The algorithm still needs to set the global learning rate η manually.

ii) As the training time increases, the accumulated gradient will become larger and larger, making the learning rate tend to zero, resulting in ineffective parameter update.

AdaGrad was further improved to AdaDelta [57] and RMSProp [58] for solving the problem that the learning rate will eventually go to zero. The idea is to consider not accumulating all historical gradients, but focusing only on the gradients in a window over a period, and using the exponential moving average to calculate the second-order cumulative momentum,

$$V_t = \sqrt{\beta V_{t-1} + (1-\beta)(g_t)^2} \quad \text{----------} \rightarrow \text{Eqn(3.7)}$$

where ß is the exponential decay parameter. Both RMSProp and AdaDelta have been developed independently around the same time, stemming from the need to resolve the radically diminishing learning rates of AdaGrad. Adaptive moment estimation (Adam) [55] is another advanced SGD method, which introduces an adaptive learning rate for each parameter. It combines the adaptive learning rate and momentum methods. In addition to storing an exponentially decaying average of past squared gradients V, like AdaDelta and RMSProp, Adam also keeps an exponentially decaying average of past gradients mt, similar to the momentum method:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \text{ ----------} \rightarrow \text{Eqn(3.8)}$$

$$V_t = \sqrt{\beta_2 V_{t-1} + (1-\beta_2)(g_t)^2} \text{ ----------} \rightarrow \text{Eqn(3.9)}$$

Where β_1 and β_2 are exponential decay rates. The final update formula for the parameter θ is

$$\theta_{t+1} = m_t - \eta \frac{\sqrt{1-\beta_2}}{1-\beta_1} \frac{m_t}{V_t+\epsilon} \text{ ----------} \rightarrow \text{Eqn(3.10)}$$

The default values of β_1, β_2 and ϵ are suggested to set to 0.9, 0.999 and 10^{-8}, respectively. Adam works well in practice compares favourably to other adaptive learning rate algorithms.

3.12.3. RMSProp

RMSprop and Adadelta have both been developed independently around the same time stemming from the need to resolve Adagrad's radically diminishing learning rates [47]. RMSprop in fact is identical to the first update vector of Adadelta that we derived above:

$$Eg_t^2 = 0.9Eg_{t-1}^2 + 0.1g_t^2 \text{ ----------} \rightarrow \text{Eqn(3.11)}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{Eg_t^2+\epsilon}} g_t \text{ ----------} \rightarrow \text{Eqn(3.12)}$$

RMSprop as well divides the learning rate by an exponentially decaying average of squared gradients. Hinton suggests γ to be set to 0.9, while a good default value for the learning rate η is 0.001.

3.12.4. Adam

Adaptive Moment Estimation (Adam) [55] is another method that computes adaptive learning rates for each parameter. In addition to storing an exponentially decaying average of past squared gradients vt like Adadelta and RMSprop, Adam also keeps an

exponentially decaying average of past gradients mt, similar to momentum. Whereas momentum can be seen as a ball running down a slope, Adam behaves like a heavy ball with friction, which thus prefers flat minima in the error surface [47]. We compute the decaying averages of past and past squared gradients mt and vt respectively as follows:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \text{ ----------} \rightarrow \text{Eqn(3.13)}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \text{ ----------} \rightarrow \text{Eqn(3.14)}$$

mt and vt are estimates of the first moment (the mean) and the second moment (the uncentered variance) of the gradients respectively, hence the name of the method. As mt and vt are initialized as vectors of 0's, the authors of Adam observe that they are biased towards zero, especially during the initial time steps, and especially when the decay rates are small (i.e. β1 and β2 are close to 1).

They counteract these biases by computing bias-corrected first and second moment estimates:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \text{ ----------} \rightarrow \text{Eqn(3.15)}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \text{ ----------} \rightarrow \text{Eqn(3.16)}$$

They then use these to update the parameters just as we have seen in Adadelta and RMSprop, which yields the Adam update rule:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t \text{ ----------} \rightarrow \text{Eqn(3.17)}$$

They show empirically that Adam works well in practice and compares favorably to other adaptive learning-method algorithms.

3.12.5. Nadam

As we have seen before, Adam can be viewed as a combination of RMSprop and momentum: RMSprop contributes the exponentially decaying average of past squared gradients vt, while momentum accounts for the exponentially decaying average of past gradients mt. We have also seen that Nesterov accelerated gradient (NAG) is superior to vanilla momentum. Nadam (Nesterov-accelerated Adaptive Moment Estimation) thus combines Adam and NAG [47]. In order to incorporate NAG into Adam, we need to modify its momentum term mt. First, let us recall the momentum update rule using our current notation:

$$g_t = \nabla_{\theta t} J(\theta_t) \text{ ----------} \rightarrow \text{Eqn (3.18)}$$

$$m_t = \gamma m_{t-1} + \eta g_t \text{ ----------} \rightarrow \text{Eqn (3.19)}$$

$$\theta_{t+1} = \theta_t - m_t \text{ ----------} \rightarrow \text{Eqn (3.20)}$$

where J is our objective function, γ is the momentum decay term, and η is our step size. Expanding the third equation above yields:

$$\theta_{t+1} = \theta_t - (\gamma m_{t-1} + \eta g_t) \text{ ----------} \rightarrow \text{Eqn (3.21)}$$

This demonstrates again that momentum involves taking a step in the direction of the previous momentum vector and a step in the direction of the current gradient. NAG then allows us to perform a more accurate step in the gradient direction by updating the parameters with the momentum step *before* computing the gradient. We thus only need to modify the gradient gt to arrive at NAG:

$$g_t = \nabla_{\theta t} J\,(\theta_t - \gamma m_{t-1}) \text{ ----------} \rightarrow \text{Eqn (3.22)}$$

$$m_t = \gamma m_{t-1} + \eta g_t \text{ ----------} \rightarrow \text{Eqn (3.23)}$$

$$\theta_{t+1} = \theta_t - m_t \text{ ----------} \rightarrow \text{Eqn (3.24)}$$

Dozat proposes to modify NAG the following way: Rather than applying the momentum step twice -- one time for updating the gradient gt and a second time for updating the parameters θt+1-- we now apply the look-ahead momentum vector directly to update the current parameters:

$$g_t = \nabla_{\theta t} J\,(\theta_t) \text{ ----------} \rightarrow \text{Eqn (3.25)}$$

$$m_t = \gamma m_{t-1} + \eta g_t \text{ ----------} \rightarrow \text{Eqn (3.26)}$$

$$\theta_{t+1} = \theta_t - (\gamma m_{t-1} + \eta g_t) \text{ ----------} \rightarrow \text{Eqn (3.27)}$$

Notice that rather than utilizing the previous momentum vector mt−1 as in the equation of the expanded momentum update rule above, we now use the current momentum vector mt to look ahead. In order to add Nesterov momentum to Adam, we can thus similarly replace the previous momentum vector with the current momentum vector.

3.13. METRIC VALUES

Choosing the right evaluation metric for classification models is important to the success of a machine learning project. Monitoring only the ‘accuracy score’ gives an incomplete picture of project model’s performance and can impact the effectiveness. So, consider the following evaluation metrics before conclude classifier model.

3.13.1. Sensitivity

Sensitivity is a measure of the proportion of actual positive cases that got predicted as positive (or true positive). Sensitivity is also termed as Recall. This implies that there will be another proportion of actual positive cases, which would get predicted incorrectly as negative (and, thus, could also be termed as the false negative). This can also be represented in the form of a false negative rate. The sum of sensitivity and false negative rate would be 1.

Mathematically, sensitivity can be calculated as the following:

$$Sensitivity = \frac{True\ Positive}{(True\ Positive + False\ Negative)} \text{ ----------} \rightarrow \text{Eqn (3.28)}$$

The higher value of sensitivity would mean higher value of true positive and lower value of false negative. The lower value of sensitivity would mean lower value of true positive and higher value of false negative.

3.13.2. Specificity:

Specificity is defined as the proportion of actual negatives, which got predicted as the negative (or true negative). This implies that there will be another proportion of actual negative, which got predicted as positive and could be termed as false positives. This proportion could also be called a false positive rate. The sum of specificity and false positive rate would always be 1. Mathematically, specificity can be calculated as the following:

$$Specificity = \frac{True\ Negative}{(True\ Positive + False\ Negative)} \text{ ----------} \rightarrow \text{Eqn (3.29)}$$

3.14. EXPERIMENTAL SETUP

3.14.1. Training and testing infrastructure

The hardware utilized comprised a Core i7, 2.4 GHz processor with 16 GB RAM and NVIDIA RTX2060 GPU. The operating system used was Windows 10, 64-bit. Image processing techniques and other related Python, Keras and Transfer learning toolboxes were utilized in the development of the system. The research utilized Cross-Validation for data analysis. It is a statistical technique used in evaluating predictive models by feature extraction of data into training and validation sets. After evaluation of model saved the weights into .h5py format. The h5py package is a Pythonic interface to the HDF5 binary data format. It lets you store huge amounts of numerical

data, and easily manipulate that data from NumPy. These saved files are loaded and tested using Raspberry Pi 3 B+ model processor.

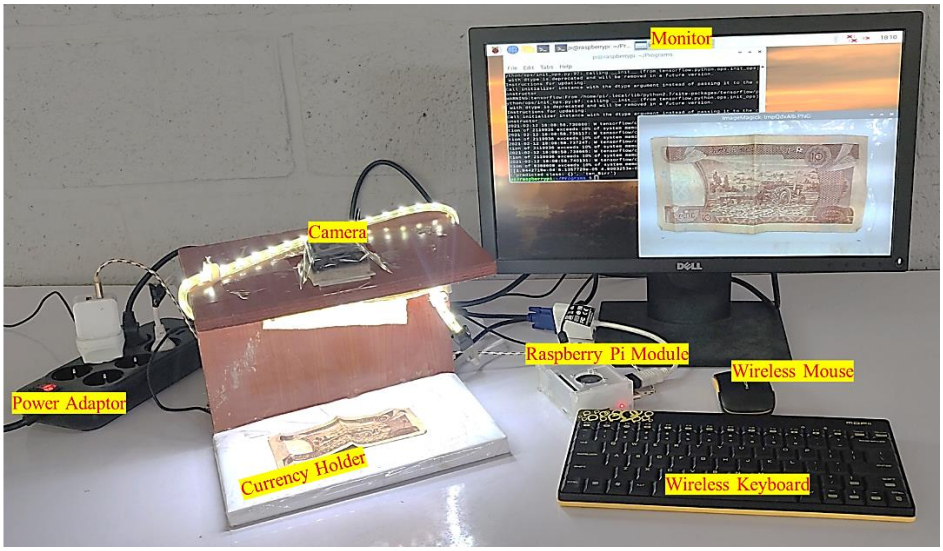

Figure 3.23 Experimental setup of the project

3.14.2. Raspberry Pi 3 B+ model

The specialized high performance artificial intelligence (AI) chips will eventually provide developers the ability to implement sophisticated machine learning algorithms. The Raspberry Pi 3 offers some immediate advantages as a development platform for machine learning applications. Its Arm Cortex-A53 quad core processor provides significant performance capabilities, and the core's NEON single instruction, multiple data (SIMD) extensions are capable of performing a certain level of multimedia and machine learning type processing. The Raspberry Pi is a wonderful microcomputer that brims with potential. With a Raspberry Pi you can build robots, learn to code, and create all kinds of weird and wonderful projects. It is a cheap, credit card sized computer running a Linux operating system (Raspbian/Ubuntu) designed for the researchers to learn programming. The Raspberry Pi 3 Model B+ is the latest product in the Raspberry Pi 3 range, boasting an updated 64-bit quad core processor running at 1.4GHz with built-in metal heatsink, dual-band 2.4 GHz and 5 GHz wireless LAN**,** faster (300 mbps) Ethernet, and PoE capability via a separate PoE HAT**.**

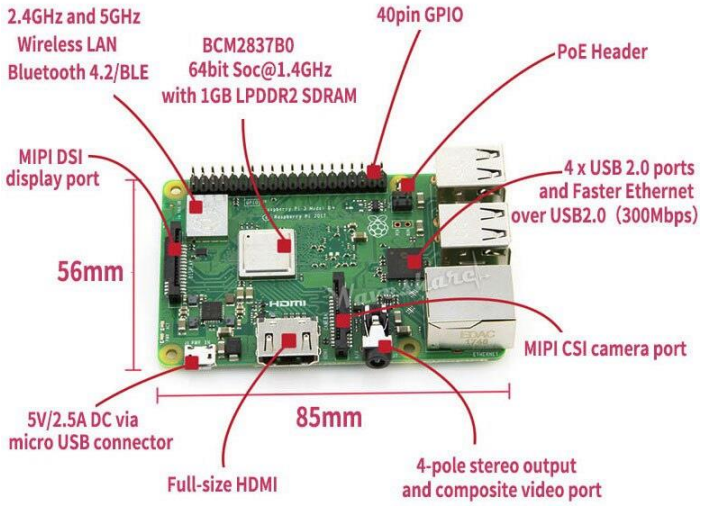

Figure 3.24 Raspberry Pi peripherals module

Table 3.4 Specifications of Raspberry Pi 3 B+ model

Specification	Range
Processor	Broadcom BCM2837B0, Cortex-A53, 64-bit SoC @ 1.4GHz
Memory	1GB LPDDR2 SDRAM
Connectivity	2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE, Gigabit Ethernet over USB 2.0, 4 × USB 2.0 ports
Access	Extended 40-pin GPIO header
Input power	5V/2.5A DC via micro-USB connector, 5V DC via GPIO header
GPU	Broadcom Video core-IV

a) Software:

To implement the project, we used Raspbian operating system on Raspberry Pi. The Raspberry Pi is an amazing single board computer (SBC) capable of running Linux and a whole host of applications. Raspbian is the official Linux distribution that's developed and maintained by the Raspberry Pi Foundation. Raspberry Pi OS is highly optimized for the Raspberry Pi line of compact single-board computers with ARM CPUs. Raspberry Pi OS uses a modified LXDE as its desktop environment with the Openbox stacking window manager, along with a unique theme. The distribution is shipped with a copy of the algebra program Wolfram Mathematica and a version of Minecraft called Minecraft: Pi Edition, as well as a lightweight version of the Chromium web browser. Raspberry Pi OS looks like many common desktops, such as macOS and Windows. The menu bar is positioned at the top and contains an

application menu and shortcuts to Terminal, Chromium, and File Manager. On the right is a Bluetooth menu, a Wi-Fi menu, volume control, and a digital clock.

b) Programming languages:

Python is the primary programming language on Raspberry Pi used in this research to develop the CNN architecture models. We saved proposed CNN architecture weight values using python program in h5py format. There is no limitation on the use of any programming language on the RPi board. Any programming language that is supported by the operating system running on it can be used for software development. Other popular programming languages that can be used on Raspberry Pi include C, C++, Java, HTML5, JavaScript, JQuery, Pearl, Erlang, etc.

3.14.3. Logitech Camera

To capture the Ethiopia currency 5-birr, 10-birr, 50-birr and 100-birr images we used the Logitech C920 camera module. This web camera captures the input/test image and send to the processor to classify the image. The specifications of camera mentioned in table.

Figure 3.25 Logitech camera device

Table 3.5 Specifications of Logitech camera

Specification	Range
Optical Resolution	True: 3 MP, software Enhanced: 15 MP
Image Capture	3 MP, 6 MP, 15 MP
Video capture	480p, 720p, 1080p
Frame Rate max	1080@30fps
Anti Flicker	50 Hz/ 60 Hz
Diagonal Field of View	78°
Horizontal Field of View	70.42°
Vertical Field of View	43.3°
Focal Length	3.67 mm

CHAPTER-4

RESULTS AND DISCUSSIONS

For the performance evaluation, the designed framework is trained on four (4) types of convolutional neural network architectures including Inception-v3, MobileNet, XceptionNet and ResNet-50, individually. The proposed architectures fused six different optimization techniques namely: Adam, SGD, RMSProp, Nadam, Adadelta and Adagrad. For all these models and optimization techniques are conducted simultaneously for 100 epochs. The experiments are performed on high performance computer using NVIDIA RTX2060 GPU.

When training a CNN model, one of the main things that to avoid overfitting. This is when model fits the training data well, but it can't generalize and make accurate predictions for data it hasn't seen before. To find out if their model is overfitting, a technique called cross-validation, where we split their data into three parts - the training dataset, test dataset and the validation dataset. The splitting data are performed in three different ways with the ratio of 70% for training, 20% for testing and 10% for validation. The training set is used to train the model, while the validation set is only used to evaluate the model's performance. Metrics on the training set tells how a model is progressing in terms of its training, but it's metrics on the validation set measures the quality of model able to make new predictions based on data it hasn't seen before. The loss and accuracy are measures of loss and accuracy on the training set, while test loss and test accuracy are measures of loss and accuracy on the test data set. Tables 4.1, 4.2, 4.3, 4.4 and 4.5 present average of train accuracy, test accuracy, train loss, test loss and validation time of bank note recognition using Inception V3, MobileNet, XceptionNet and ResNet50 architectures with various batch sizes of 32, 64 and 128 respectively. The figures 4.1, 4.2, 4.3 and 4.4 represent the persistence graph of train accuracy, test accuracy, train loss, test loss and validation time of above-mentioned models and optimizations.

4.1.TRAIN ACCURACY

A train accuracy metric is used to measure the algorithm's performance in an interpretable way. The train accuracy of a model is usually determined after the model parameters and is calculated in the form of a percentage. It is the measure of how accurate proposed model's prediction is compared to the true data. The bank note classification of training accuracy results are presented in Table 4.1. It is understood

that the highest training accuracy of 96.98% is achieved by MobileNet using RMSProp optimization in all the batch sizes. Because the MobileNet structure is built on depth wise separable convolutions except for the first layer which is a full convolution. MobileNet models were trained in TensorFlow using RMSprop with asynchronous gradient descent like Inception V3. When training MobileNets we do not use side heads or label smoothing and additionally reduce the amount image of distortions by limiting the size of small crops that are used in large Inception training. The second hyper-parameter resolution multiplier of MobileNet is reduce the computational cost of a neural network. The results are also presented graphically and are shown in Fig. 4.1. As shown in Fig.4.1 (a-c), the highest accuracy is obtained in MobileNet using RMSProp optimization technique. Apart from this, using the combinations of MobileNet with Adam and MobileNet Nadam optimizations, it is found that an accuracy of 96.96%. is achieved which are the second-best accuracy values among other optimization methods. However, the Resnet50 model also shows the better accuracy values of 96.88% using Adagrad optimization with batch sizes 32 and 128. But it is also almost nearer to value with the first model which is MobileNet model. Except the model InceptionV3 with Adadelta optimization technique, all other optimized combinations produced more than 90% accuracy.

The Adadelta optimization technique showing low performance with an accuracy of 46.46% for InceptionV3 model with a batch size of 128. Hence this optimization technique is unfit to predict all the Ethiopian currencies. Hence MobileNet is more accurate than InceptionV3, Xception and ResNet50 while being smaller and more than 2.5 times less Computation. The graph Fig. 4.1 shows the persistent of training accuracy for all the proposed models and optimization techniques. It is found that when compared to MobileNet, Resnet50 is showing almost constant values for all optimization techniques and batch sizes.

Table 4.1 Training accuracy of batch sizes 32, 64 and 128

Model / Optimization	Batch Size 32				Batch Size 64				Batch Size 128			
	InceptionV3	MobileNet	XceptionNet	ResNet 50	InceptionV3	MobileNet	XceptionNet	ResNet 50	InceptionV3	MobileNet	XceptionNet	ResNet 50
Adam	96.38	96.96	94.80	94.78	96.30	96.96	95.44	94.85	96.46	96.96	94.61	94.83
SGD	95.75	94.56	91.59	96.56	96.30	92.65	94.12	96.42	96.46	92.93	92.28	96.47
RMSProp	96.59	96.98	92.28	94.65	96.58	96.98	92.20	94.57	96.57	96.98	96.20	94.56
Nadam	96.45	96.96	96.29	94.93	96.35	96.96	96.24	96.00	96.37	96.96	96.25	96.02
Adadelta	59.34	87.31	89.61	94.75	51.15	86.09	91.35	94.99	46.46	87.10	94.30	94.89
Adagrad	91.48	94.82	94.47	96.88	85.34	94.89	94.44	96.87	91.94	96.00	94.61	96.88

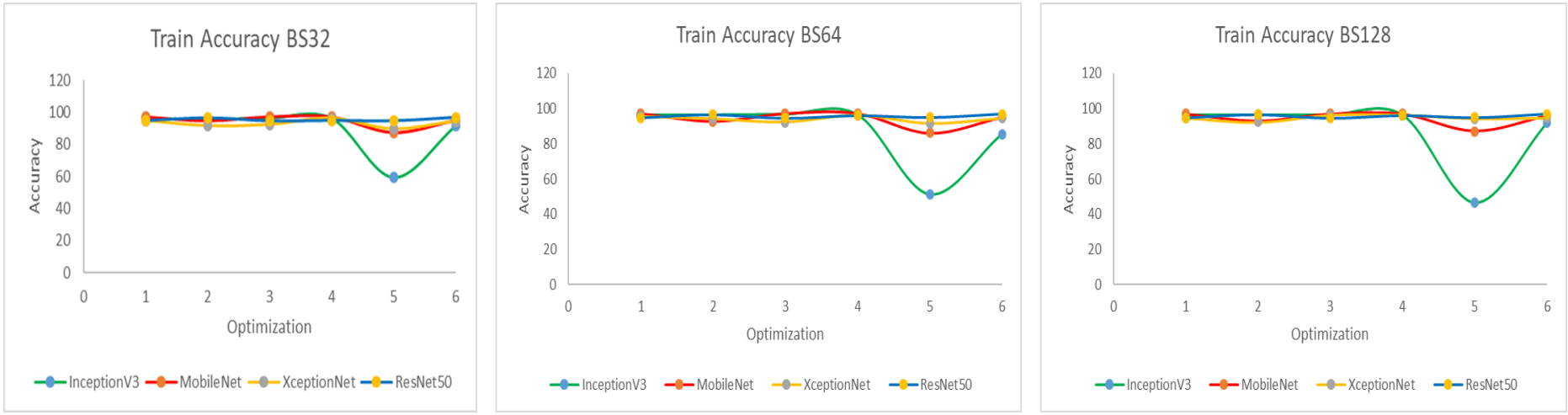

Figure 4.1 Graph of the average training accuracy for each dataset with each classifier and optimizer

4.2.TEST ACCURACY

The test accuracy must measure performance on *unseen* data, which are not used for training data. To measure the test accuracy used the test dataset. It is a dataset used to provide an unbiased evaluation of a *final* model fit on the training dataset. If the data in the test dataset has never been used in training. The bank note classification test accuracy results are presented in Table 4.2. It is understood that the highest training accuracy of 96.97% is achieved by ResNet50 using Adagrad optimization in batch size 64 and 128. The results are also presented graphically and are shown in Fig. 4.2. As shown in Fig.4.2 (a-c), the highest accuracy is obtained in ResNet50 using Adagrad optimization technique. However, the MobileNet model also shows the better accuracy values of 96.96% using RMSProp optimization with batch sizes 32. But it is also almost nearer to value with the first model which is ResNet50 model. The test accuracy values are very low for Adadelta optimization for all proposed models.

The Adadelta optimization technique showing low performance with all proposed models. Hence this optimization technique is unfit to predict all the Ethiopian currencies. The graph Fig. 4.2 shows the persistent of test accuracy for all the proposed models and optimization techniques. It is found that when compared to MobileNet, XceptionNet is showing almost constant values for all optimization techniques and batch sizes. But XceptionNet is showing less test accuracy compared to MobileNet and ResNet50.

Table 4.2 Test accuracy of batch sizes 32, 64 and 128

Model / Optimization	Batch Size 32				Batch Size 64				Batch Size 128			
	Inception V3	**MobileNet**	**XceptionNet**	**ResNet50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet50**
Adam	96.26	96.94	90.94	79.06	94.30	95.95	96.32	77.26	96.29	96.86	96.34	79.47
SGD	94.21	93.20	92.18	96.87	94.30	91.34	92.70	96.85	96.29	93.46	92.76	96.72
RMSProp	96.45	96.96	92.76	76.06	96.42	96.93	90.33	77.16	96.52	96.90	92.96	78.57
Nadam	92.30	96.92	91.12	75.63	94.01	96.90	94.59	80.48	94.04	96.98	94.25	80.62
Adadelta	61.61	88.15	90.69	96.00	50.28	89.05	93.52	96.33	46.95	87.91	94.21	96.24
Adagrad	90.19	96.21	96.38	96.96	90.19	96.22	96.32	96.97	90.53	96.29	96.34	96.97

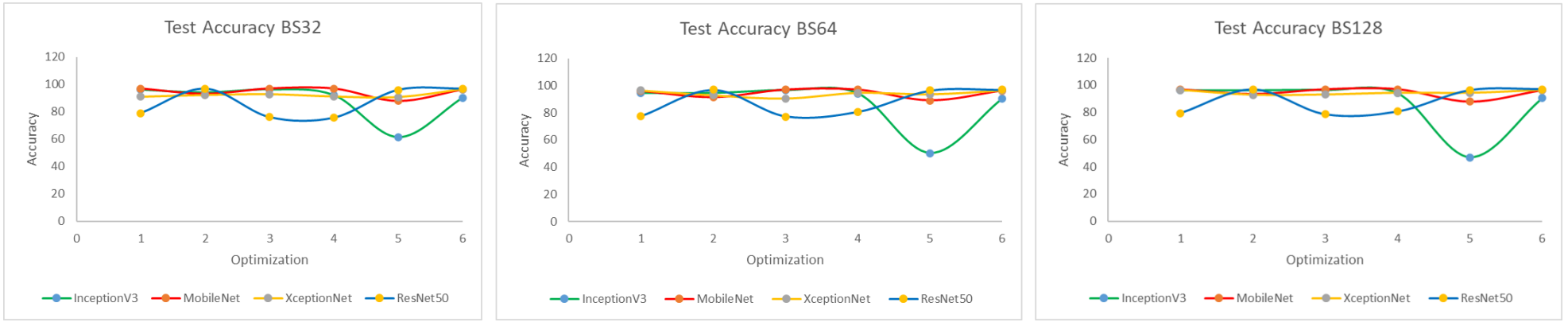

Figure 4.2 Graph of the average test accuracy for each dataset with each classifier and optimizer

4.3.TRAIN LOSS

Loss is the penalty for a bad prediction. That is**,** loss is a number indicating how bad the model's prediction was on a single example. If the model's prediction is perfect, the loss is zero; otherwise, the loss is greater. The goal of training a model is to find a set of weights and biases that have *low* loss, on average, across all examples. The loss is calculated on training and validation and its interpretation is based on how well the model is doing in these two sets. It is the sum of errors made for each example in training or validation sets. Loss value implies how poorly or well a model behaves after each iteration of optimization. In this research used the loss function as Categorical crossentropy to classify multiple classes.

The MobileNet architecture showing training loss is zero for Adam, RMSProp and Nadam optimization techniques in batch sizes 32, 64 and 128. Compared to other models with various optimization techniques and batch sizes MobileNet showing low loss values. The ResNet50 showed good training and test accuracy but it hasn’t achieved zero loss for any optimization techniques.

Table 4.3 Train loss of batch sizes 32, 64 and 128

Model / Optimization	Batch Size 32				Batch Size 64				Batch Size 128			
	Inception V3	**MobileNet**	**XceptionNet**	**ResNet50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet 50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet 50**
Adam	0.02	0.00	0.07	0.05	0.02	0.00	0.07	0.05	0.02	0.00	0.07	0.05
SGD	12.75	18.01	21.80	2.66	0.02	0.17	0.18	0.03	0.02	0.17	0.18	0.03
RMSProp	0.01	0.00	0.18	0.10	0.02	0.00	0.09	0.10	0.02	0.00	0.06	0.09
Nadam	0.36	0.00	0.03	0.04	0.35	0.00	0.03	0.04	0.24	0.00	0.03	0.04
Adadelta	1.15	0.49	0.34	0.06	1.24	0.48	0.32	0.05	1.26	0.50	0.31	0.05
Adagrad	0.33	0.07	0.07	0.01	0.33	0.07	0.07	0.01	0.32	0.07	0.07	0.01

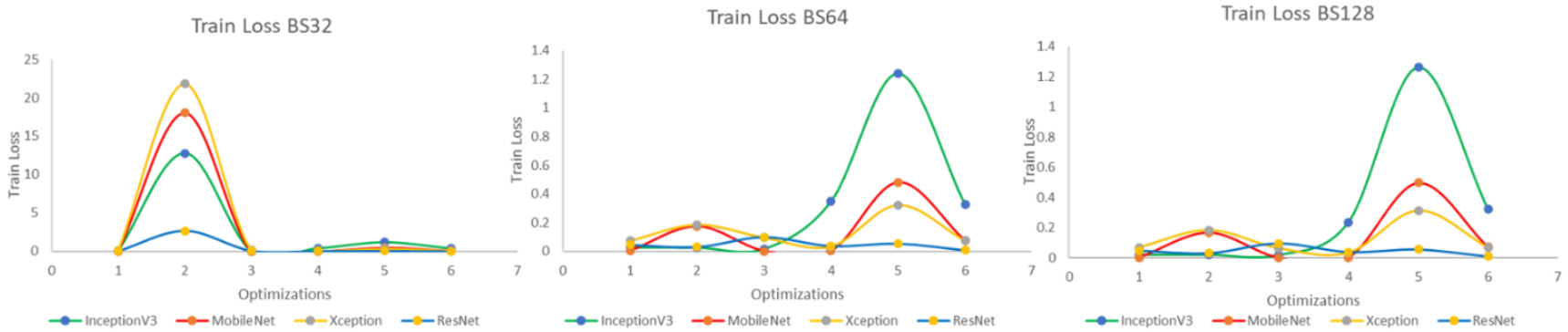

Figure 4.3 Graph of the average train loss for each dataset with each classifier and optimizer

4.4.TEST LOSS

Every subsequent winning architecture uses more layers in a deep neural network to reduce the error rate. This works for a smaller number of layers, but when we increase the number of layers, there is a common problem in deep learning associated with that called Vanishing/Exploding gradient. This causes the gradient to become 0 or too large. Thus, when we increase number of layers, the training and test error rate also increases. In the above plot figure 4.4, we can observe that a 50-layer Resnet, 48-layer InceptionV3 and 36-layer XceptionNet CNN gives more error rate on both training and testing dataset than a MobileNet 28-layer CNN architecture. The MobileNet architecture contains a smaller number of layers compared to other network models. Hence it is showing training loss become zero. The MobileNet and ResNet50 are showed almost equal training and test accuracy but the test loss is very heigh for the ResNet50 because of its gradient is too large so, this model is overfit the data.

Table 4.4 Test loss of batch sizes 32, 64 and 128

Model / Optimization	Batch Size 32				Batch Size 64				Batch Size 128			
	InceptionV3	**MobileNet**	**XceptionNet**	**ResNet 50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet 50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet 50**
Adam	3.28	0.34	104.48	183.74	27.65	0.21	3.99	182.01	2.44	0.50	3.79	250.94
SGD	7.92	15.81	14.42	1.06	37.65	15.30	13.57	1.30	2.44	14.83	13.88	1.44
RMSProp	1.83	0.11	13.88	291.95	2.30	0.18	110.69	373.63	2.72	0.24	160.91	455.04
Nadam	1.38	0.24	88.62	484.78	2.94	0.44	87.80	189.05	5.56	0.08	48.57	416.08
Adadelta	112.05	46.75	28.59	4.19	124.38	45.63	26.05	3.92	126.22	47.32	25.34	3.93
Adagrad	25.90	5.83	3.92	0.29	25.86	5.96	3.99	0.34	25.38	5.90	3.79	0.34

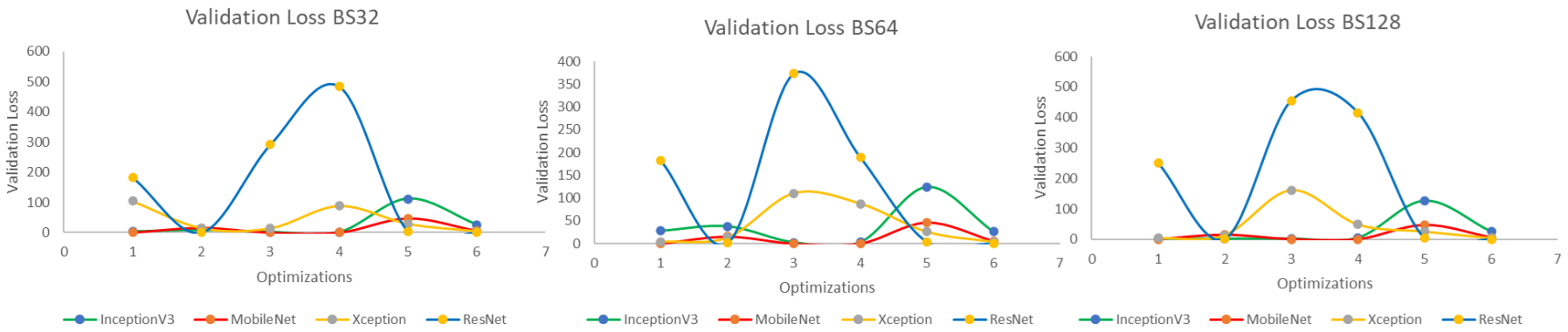

Figure 4.4 Graph of the average test loss for each dataset with each classifier and optimizer

4.5.VALIDATION TIME

Table 4.5 presents the average processing (testing) time for the Ethiopian bank currency 5-birr, 10-birr, 50-birr, and 100-birr, employed using various CNN models. To improve the recognition time, each system was pre-trained before testing on a GPU based Laptop. The time for each set computed with different classifier as each technique varied because of the architecture's approach. It shows the validation time for the dataset with each method, the InceptionV3 using Adam optimization technique in batch size 32 had the shortest validation time 34.15 seconds. But Inceptionv3 showed less training accuracy and test accuracy. The second fastest validation time classifier is MobileNet model. It is showing 35.09 sec. Figure 4.5 shows the graph of average validation time for each dataset with each classifier. The InceptionV3 and MobileNet had the shortest processing time when optimization technique using the Adam and SGD. The XceptionNet had the highest validation time 664.41 seconds on the proposed classification methods.

From the result, we concluded that the banknote dimensions caused a variation in the image processing time and the feature extraction time was dispersed as a function of the number of pixels, cells, and blocks utilized in each dataset. The feature reduction process was taken into consideration as it contributed to the processing time. Although, the validation time differences for each classifier were small regardless of the datasets. In conclusion, although all other classifiers have a remarkable detection rate and computation time, the suggested MobileNet classification method using RMSProp produced more promising results although each image varied because of the difference in image size, quality, and pixel density.

Table 4.5 Validation time of batch sizes 32, 64 and 128 for classifiers and optimizers

Model / Optimization	Batch Size 32				Batch Size 64				Batch Size 128			
	InceptionV3	**MobileNet**	**XceptionNet**	**ResNet50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet 50**
Adam	34.15	36.23	128.91	36.82	35.15	35.09	310.34	59.58	43.33	52.57	132.35	36.23
SGD	33.92	35.12	155.89	35.44	38.92	35.09	124.84	51.14	43.33	50.91	304.66	35.6
RMSProp	35.87	50.91	135.54	37.35	37.87	36.22	132.87	53.21	34.66	53.05	129.42	36.4
Nadam	37.6	55.36	187.16	40.85	39.6	36.58	124.09	56.84	37.49	55.93	125.39	40.26
Adadelta	36.65	51.23	127.73	69.61	42.65	45.51	126.27	51.77	237.57	51.44	664.41	71.11
Adagrad	37.79	52.42	128.13	89.92	37.79	36.84	123.32	59.58	36.7	52.57	123.41	66.07

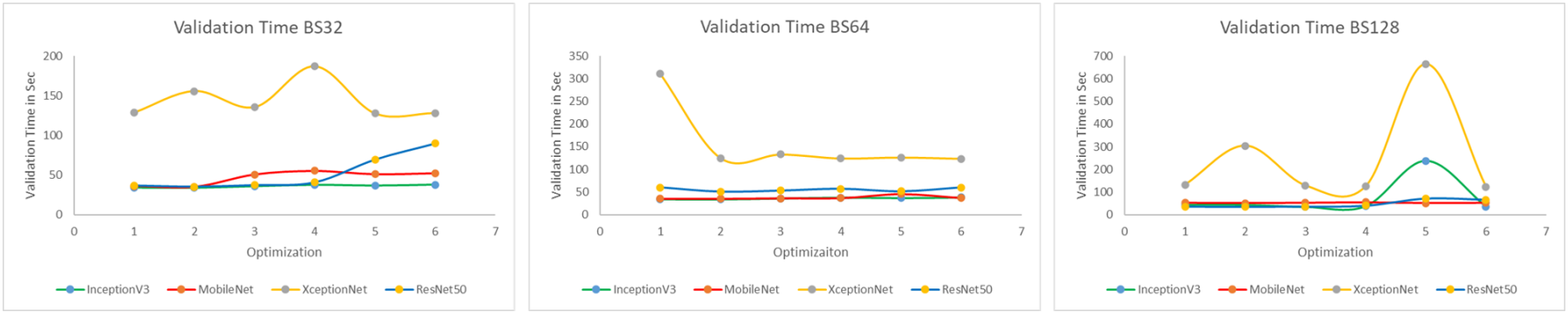

Figure 4.5 Graph of the average validation time for each dataset with each classifier and optimizer

4.6.SENSITIVITY

Sensitivity is a measure of the proportion of actual positive banknotes cases that got predicted as positive banknotes (or true positive). This implies that there will be another proportion of actual positive notes, which would get predicted incorrectly as negative (and, thus, could also be termed as the false negative). This can also be represented in the form of a false negative rate. The sum of sensitivity and false negative rate would be 1. Mathematically, sensitivity can be calculated as the following:

$$Sensitivity = \frac{(True\ Positive)}{(True\ Positive + Flse\ Negative)} \quad \text{----------} \rightarrow \text{Eqn. (4.1)}$$

The MobileNet model with RMSProp in batch size 32 showed high sensitivity 96.53 % compared to remaining models shown in the table 4.6. The figure 4.6 shows the graph of proposed models' sensitivity.

4.7.SPECIFICITY

Specificity is the proportion of actual negatives, which got predicted banknotes as the negative (or true negative). This implies that there will be another proportion of actual negative, which got predicted as positive and could be termed as false positives. This proportion could also be called a false positive rate. The sum of specificity and false positive rate would always be 1. Mathematically, specificity can be calculated as the following:

$$Specificity = \frac{(True\ Negative)}{(True\ Negative + Flse\ Positive)} \quad \text{----------} \rightarrow \text{Eqn. (4.2)}$$

The MobileNet model with RMSProp in batch size 128 showed high sensitivity 96.28 % compared to remaining models shown in the table 4.7. The figure 4.7 shows the graph of proposed models sensitivity.

Table 4.6 Sensitivity of batch sizes 32, 64 and 128 for classifiers and optimizers

Model / Optimization	Batch Size 32				Batch Size 64				Batch Size 128			
	InceptionV3	**MobileNet**	**XceptionNet**	**ResNet50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet50**
Adam	95.98	93.87	93.52	89.96	96.1	96.26	92.44	90.83	95.9	95.84	94.96	89.38
SGD	94.56	95.56	91.59	93.56	95.3	96.10	94.42	93.6	94.46	95.7	91.92	93.5
RMSProp	95.95	96.53	89.81	96.02	95.85	96.45	95.32	96.08	95.75	96.49	95.12	95.82
Nadam	94.54	94.35	95.29	89.26	96.01	95.95	96.04	90.36	95.73	95.02	95.25	93.36
Adadelta	69.34	91.57	91.17	79.31	69.41	96.10	92.66	82.73	66.57	94.99	93.43	84.53
Adagrad	87.84	96.01	95.47	85.82	89.34	96.15	89.67	86.62	87.65	96.08	94.18	84.82

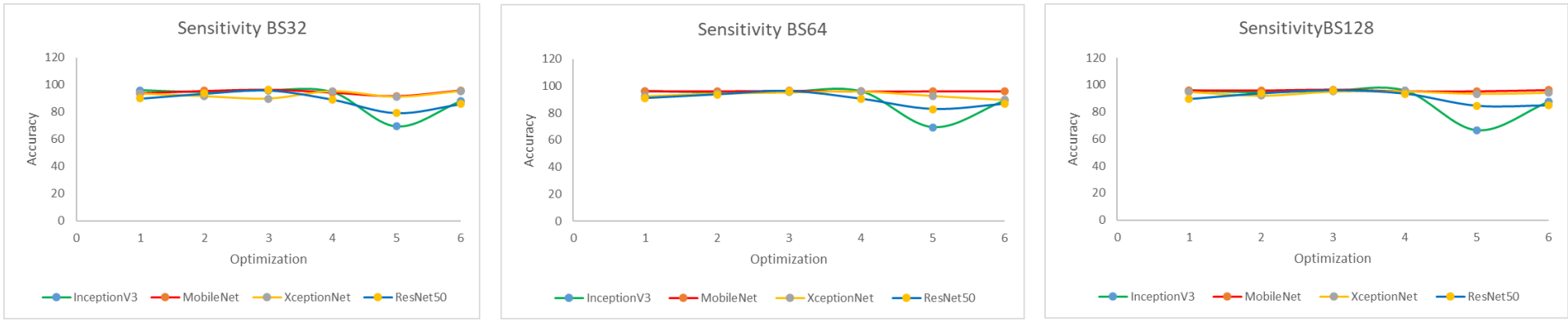

Figure 4.6 Sensitivity of proposed architecture models

Table 4.7 Specificity of batch sizes 32, 64 and 128 for classifiers and optimizers

Model / Optimization	Batch Size 32				Batch Size 64				Batch Size 128			
	InceptionV3	**MobileNet**	**XceptionNet**	**ResNet 50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet 50**	**InceptionV3**	**MobileNet**	**XceptionNet**	**ResNet 50**
Adam	92.34	90.87	91.43	86.23	95.65	92.36	88.54	92.38	95.32	90.74	89.45	89.29
SGD	89.64	90.34	94.23	92.37	94.47	94.95	90.67	92.93	93.67	90.34	86.67	91.1
RMSProp	93.82	95.81	89.67	92.93	95.84	95.98	92.23	90.28	94.97	96.28	88.24	94.25
Nadam	90.12	94.45	87.34	91.1	95.43	93.93	93.46	89.53	94.34	89.86	89.75	92.36
Adadelta	52.32	89.32	83.95	89.02	65.27	89.72	89.87	93.17	56.93	85.95	86.83	93.75
Adagrad	77.98	94.87	91.37	93.47	87.94	81.23	89.12	93.83	86.26	86.36	90.46	92.64

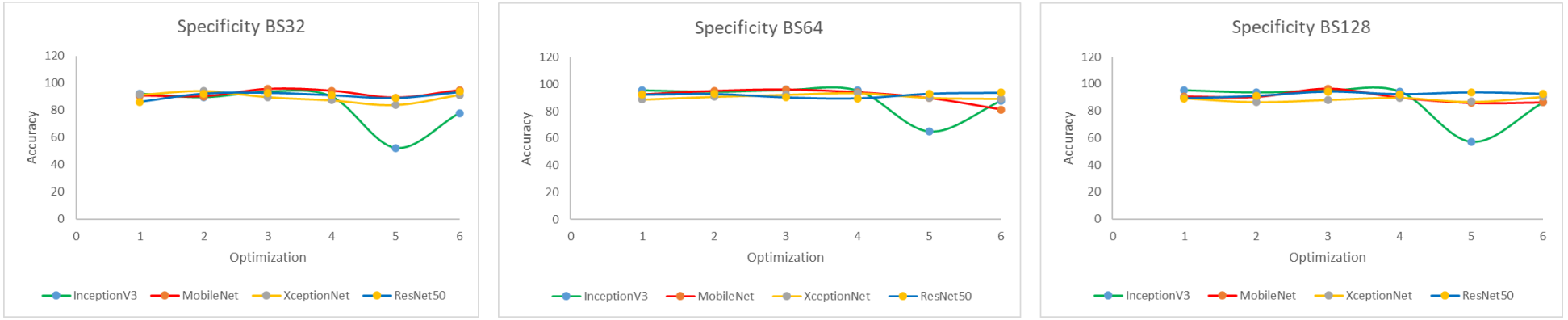

Figure 4.7 Graph of the average specificity

4.8. HARDWARE IMPLEMENTATION

As we observed the results from sensitivity and specificity metric values, the MobileNet architecture shown the best values. The second-best performance showed the ResNet50. So, we designed the embedded based hardware and tested the MobileNet and ResNet50 architecture models in batch size 32, 64 and 128. The embedded based hardware system i.e., raspberry pi captured the banknote images 10-birr and 100-birr using web camera. Using python code called the saved architecture model to test the input image. Raspberry pi successfully classified the banknote images and displayed the output on screen shown in figure 4.8. The table 4.8 shows the raspberry pi time consumption for processing and validate the banknotes. From the table clearly understand that MobileNet model with RMSProp optimization technology in batch size 32 shows less time 87 seconds due to its lightweight architecture. Compare to all the proposed architecture MobileNet is having few parameters are 4.2 M and small size 16 MB. The Resnet50 consuming more time for processing and validate compare to the MobileNet.

a) Raspberry Pi captured and tested the 10-birr note front side

b) Raspberry Pi captured and tested the 10-birr note back side

c) Raspberry Pi captured and tested the 50-birr note front side

d) Raspberry Pi captured and tested the 50-birr note back side

Figure 4.8 Embedded based hardware system for capture and test the sample bank note images

Table 4.8 Banknote image classification test time using embedded based hardware system

Model / Optimization	Batch Size 32		Batch Size 64		Batch Size 128
	Mobile Net	**ResNet 50**	**Mobile Net**	**ResNet 50**	**Mobile Net**
RMSProp	87 s	113s	105 s	118s	130s

4.9. WEB DEVELOPMENT

Researchers involved in banknote recognition and classification are required to possess deep knowledge in fake note detection, bank note classification technologies and hi-tech production of banknotes products. In such cases, use of a special authenticity verification method is extremely helpful. This web development-based currency recognition system is helpful to users and bank employees to verify the Ethiopian currencies (ETB) globally. The procedure to recognize the denomination is very simple. If a user or bank employee wants to identify the authentication of currency, they have to keep the bank note under the camera with good brightness as shown in Figure 4.9. Then, click on the “Take Snapshot” button to capture the bank note image as mentioned in Figure 4.10. Next, save the captured image from the camera using the “save snapshot” button as shown in Figure 4.11. After saving the captured image, the system can authenticate the Ethiopian currency and classify the denomination. The output window of the bank note image recognition appears as like in Figure 4.12 to show the outcome of the Ethiopian banknote recognition and denomination to be checked.

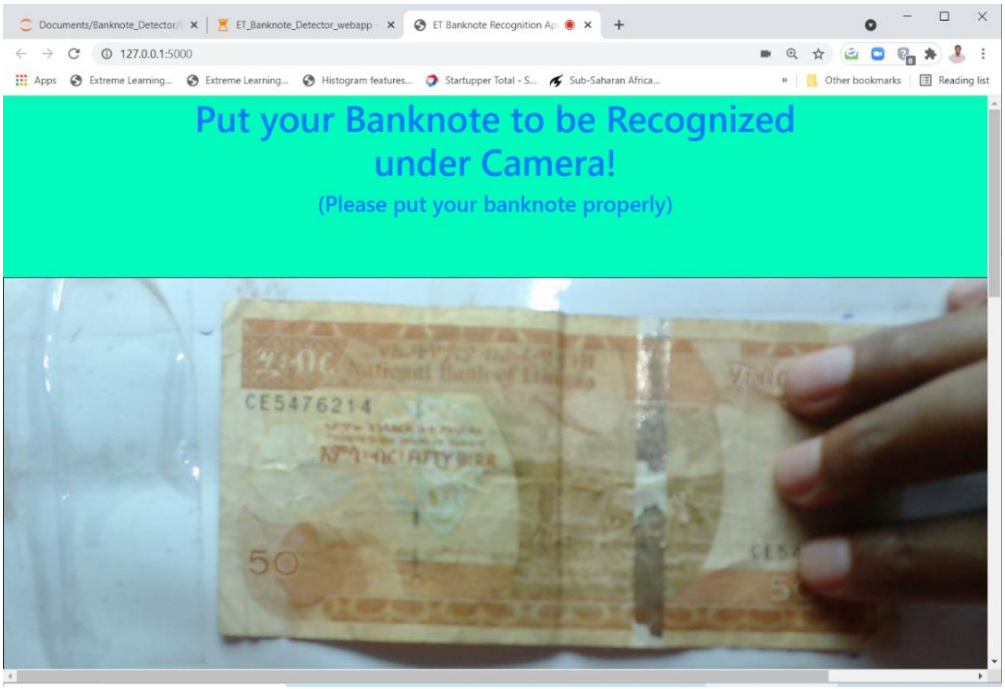

Figure 4.9 Placement of the banknote on web-based system to scan

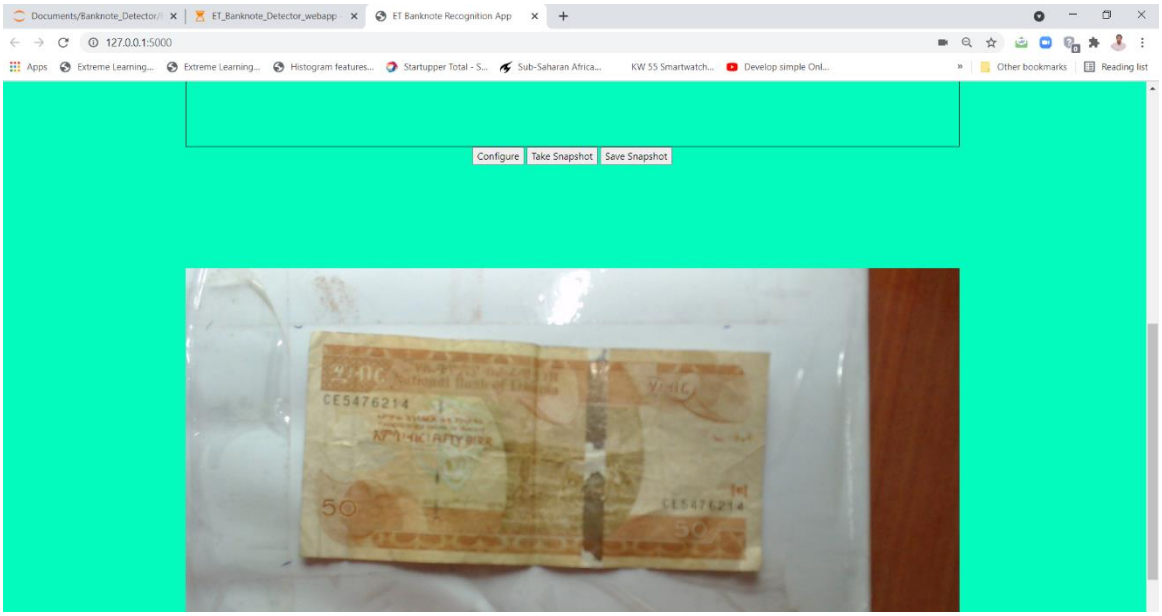

Figure 4.10 Capture the snapshot of bank note

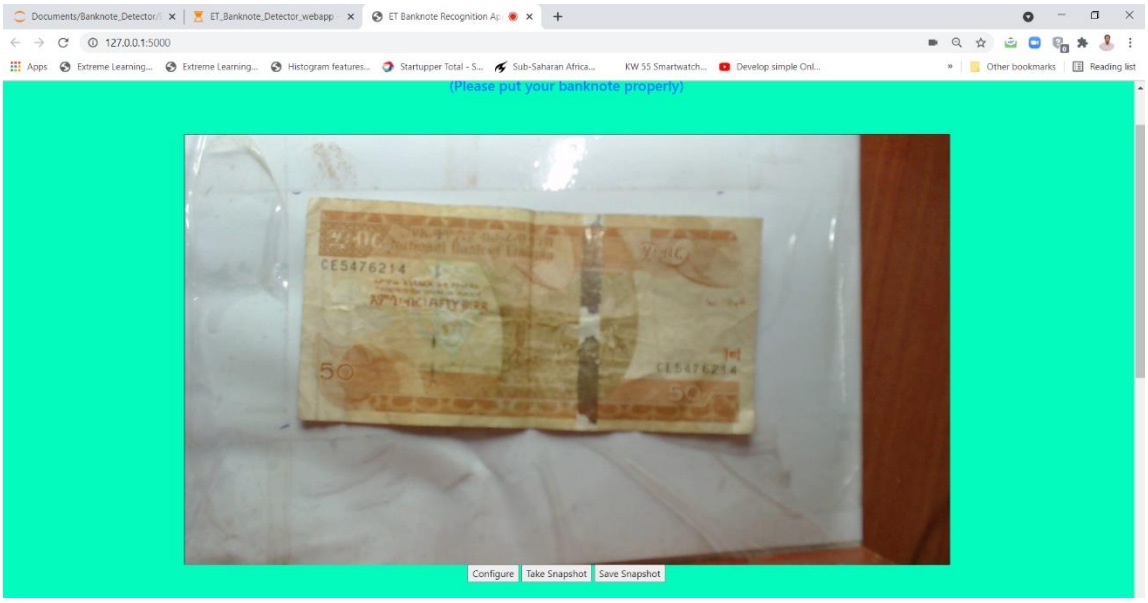

Figure 4.11 Save the snapshot of bank note images

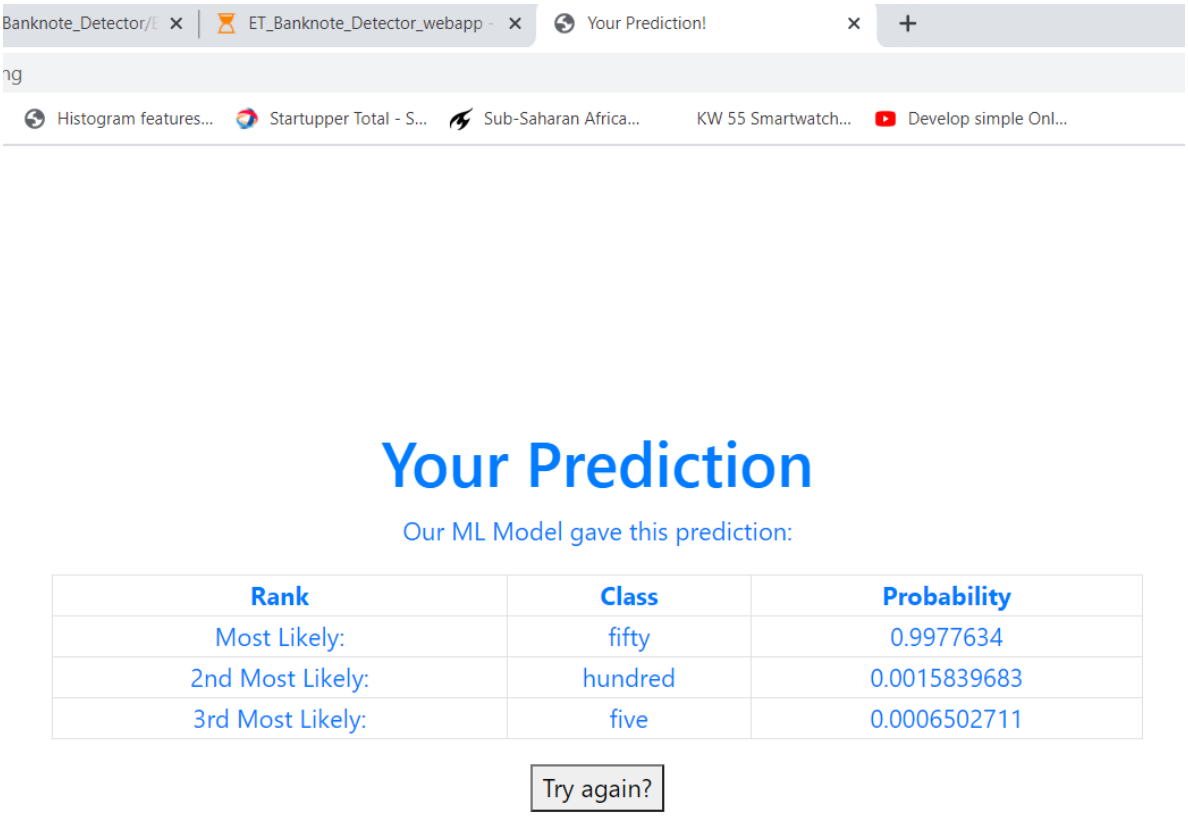

Figure 4.12 Output window of web-based bank note recognition system

CHAPTER-5
CONCLUSION AND RECOMMENDATIONS

5.1.CONCLUSION

In this research, we have developed a novel method for distinguishing and classifying original banknotes from counterfeits (another denomination) using deep learning technology. We have implemented a prototype for Ethiopian banknote recognition system that could be able to classify the various denominations of Ethiopian banknotes. We have collected about 8,500 old Ethiopian paper currency images and placed them in order to train, test and validate according to their denominations. For the performance evaluation, the designed framework of the proposed techniques are trained on four (4) types of convolutional neural network (CNN) architectures including Inception-v3, MobileNet, XceptionNet and ResNet-50 sequentially. The proposed architectures fused six different optimization techniques namely: Adam, SGD, RMSProp, Nadam, Adadelta and Adagrad used along with all the architectures for the training purposes. For all these models and optimization techniques, we have conducted iterations simultaneously for 100 epochs. CNN architectures with optimization techniques programmed using python language have been implemented on Raspberry Pi computer along with integrated Logitech camera. When compared to all the proposed CNN architectures, the MobileNet architecture with RMSProp optimization technique in batch size of 64 showed the best accuracy of about 96.80%. Furthermore, we designed the web-based program to evaluate the same identification of various denominations of Ethiopian currencies (ETB) globally. Using web-based design, we have captured various Ethiopian bank notes and classified the denominations and identifications successfully. This robust system is found to be helpful for ATMs for cash deposits and withdrawals, as well as for financial transactions, banknote counters, smart transportation system and automatic vending machines, into which money is inserted to purchase goods and automatic smart card charging machines.

The findings reported from this research work is expected to bring benefits to the development of a fast and efficient banknote recognition system using banknote images and contribute to further leading into development of more accurate point-of-care detection tools for Ethiopian banknote recognition and classification.

5.2.RECOMMENDATIONS

In the field of Banknote Recognition and classification, future work would include enlarging the size of the dataset to compare similar feature extraction techniques such as R-CNN,with

the proposed CNN based system and introducing bias-reducing features to the system to detect and reject counterfeit notes. In addition to incorporating new feature descriptors and reduction approach to reduce processing time and scale down the application to work on mobile devices, reducing the computational complexity to enable embedding in microchips. We will also be implementing a feature fusion system that combines other security features such as security thread, serial number, micro-letters, signature, and face on the banknote to improve the detection fake banknote. Lastly, we will introduce feature reduction and classification approaches to scale down the application to work solely by the processing capability of the mobile devices.

5.3. FUTURE WORK

Recently, the government of Ethiopia unveiled new currency notes replacing the existing Birr note in September 2020 (G.C.). According to the announcements, the new currency notes include Birr 10, 50 and 100 denominations, with an additional introduction of a new 200 Birr note.

This research was completed based on the old Ethiopian currencies (ETB) since this research was approved and started at the time where the new currency notes were not introduced / not in use.

Hence, we, the research project members hereby request the Office of Research Affairs of Adama Science and Technology University (ASTU) to extend this project for recognition of various denominations for the newly introduced Ethiopian currencies (ETB).

REFERENCES

[1] M.S. Uddin, P.P. Das, and M.S.A. Roney, “Image-based approach for the detection of counterfeit banknote of Bangladesh,” In Proc. of the 5th International Conference on Informatics, Electronics and Vision (ICIEV), pp. 1067-1072, 2016, Dhaka, Bangladesh.

[2] Jegnaw Fentahun Zeggeye, Yaregal Assabie, "Automatic Recognition and Counterfeit Detection of Ethiopian Paper Currency," I.J. Image, Graphics and Signal Processing, 2016.

[3] Asfaw Sheferaw and Million Meshesha, "Ethiopian Paper Currency Recognition System: An Optimal Feature Extraction," IEEE-SEM, vol. 7, no. 8, pp. 130 - 137, August 2019.

[4] Mekonnen Legess Meharu, Real-Time Ethiopian Currency Recognition for Visually Disabled Peoples Using Convolutional Neural Network, Research Square, December 2020.

[5] H. Hassanpour, A. Yaseri, and G. Ardeshiri, “Feature extraction for paper currency recognition,” In Proc. Of the 9th International Symposium on Signal Processing and Its Applications (ISSPA), pp. 1-4, 2007, Sharjah, United Arab Emirates.

[6] N. Kato, and S. Omachi, “A handwriting character recognition system using directional element feature,” In IEEE Trans. Pattern Anal. Mach. Intell, Vol. 21, No. 3, pp. 258–262, 1999.

[7] D. G. Pérez and E. B. Corrochano, ``Recognition system for Euro and Mexican banknotes based on deep learning with real scene images," Computación y Sistemas, vol. 22, no. 4, pp. 1065-1076, Dec. 2018.

[8] H. Aggarwal, and P. Kumar, “Indian Currency Note Denomination Recognition in Color Images,” In International Journal on Advanced Computer Engineering and Communication Technology, Vol. 1, No. 1, pp. 12-18, 2014.

[9] P.D. Pawar, and S.B. Kale, “Recognition of Indian Currency Note Based on HSV Parameters,” In International Journal of Science and Research (IJSR), Vo. 3, No.6, pp. 132137, 2014.

[10] Y. Liang, L. Liu, Y. Xu, Y. Xiang, and B. Zou, “Multi-task gloh feature selection for human age estimation,” (ICIP), In Proc. of the 18th IEEE International Conference on Image Processing, pp. 565-568, 2011, Brussels, Belgium.

[11] E. Choi, J. Lee, and J. Yoon, “Feature extraction for banknote classification using wavelet transform,” In Proc. of the 18th International Conference on Pattern Recognition (ICPR), pp. 934-937, 2006, Hong Kong, China.

[12] CNIB, Canadian Bank Note Reader. Retrieved from http://www.cnib.ca/en/services/products/bank-note-reader, 1996.

[13] D. A. K. S. Gunaratna, N. D. Kodikara, and H. L. Premaratne. "ANN based currency recognition system using compressed grayscale and application for Sri Lankan currency note-SLCRec," In International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol. 2, No. 9, pp. 235-240, 2008.

[14] L. D. Dunai, M. C. Pérez, G. P. Fajarnés, and I. L. Lengua, "Euro banknote recognition system for blind people,"' Sensors, vol. 17, no. 1, pp. 115, Jan. 2017.

[15] K. Verma, B. K. Singh, and A. Agarwal, "Indian currency recognition based on texture analysis," In Proc. of the Nirma University International Conference in Engineering (NUiCONE), pp. 1-5, 2011, Gujarat, India.

[16] K. N. Oo and A. K. Gopalkrishnan, "Zernike Moment Based Feature Extraction for Classification of Myanmar Paper Currencies," 2018 18th International Symposium on Communications and Information Technologies (ISCIT), Bangkok, 2018, pp. 1-6.

[17] O. K. Oyedotun & A. Khashman, "Banknote recognition: investigating processing and cognition framework using competitive neural network", Cognitive Neuro dynamics, Springer, vol. 11, issue 1, pp. 67-79, 2017.

[18] Dunai, Larisa & Chillarón Pérez, Mónica & Peris-Fajarnés, Guillermo & Lengua, Ismael. Euro Banknote Recognition System for Blind People. Sensors. 17. 184. 10.3390/s17010184, 2017.

[19] Orbit Research, iBill, Retrieved from http://www.orbitresearch.com/ibill_details.php, 2013.

[20] F. Takeda, L. Sakoobunthu and H. Satou, "Thai banknote recognition using neural network and continues learning by DSP unit," In Knowledge-Based and Intelligent Information and Engineering Systems, pp. 1169-1177, 2003.

[21] A. Ahmadi, S. Omatu, T. Fujinaka, and T. Kosaka, "Improvement of reliability in banknote classification using reject option and local PCA," In Information Sciences, Vol. 168, No. 1, pp. 277-293, 2004.

[22] T. Reiff, and P. Sincak, "Multi-Agent Sophisticated System for Intelligent Technologies," In Proc. of IEEE 6th International Conference on Computational Cybernetics, 2008, Stara Lesna, Slovakia.

[23] M. E. Ayalew Tessfaw, M. B. Ramani and M. T. Kebede Bahiru, "Ethiopian Banknote Recognition and Fake Detection Using Support Vector Machine," 2018 Second

International Conference on Inventive Communication and Computational Technologies (ICICCT), Coimbatore, 2018, pp. 1354-1359.

[24] A. Rashid, A. Prati, and R. Cucchiara, “On the design of embedded solutions to banknote recognition,” In Optical Engineering, Vol. 52, No. 9, 2013.

[25] B.V. Chetan, and P.A. Vijaya, “A Robust Side Invariant Technique of Indian Paper Currency Recognition,” In International Journal of Engineering Research and Technology, Vol. 1, No. 3, pp. 1-7, ESRSA Publications, 2012.

[26] J. Guo, Y. Zhao, and A. Cai, “A reliable method for paper currency recognition based on LBP,” In Proc. of the International Conference on Network Infrastructure and Digital Content, pp. 359-363, 2010, Beijing, China.

[27] S. Gai, G. Yang, and M. Wan, “Employing quaternion wavelet transform for banknote classification,” In Neurocomputing, Vol. 118, pp. 171-178, 2013.

[28] F. Ahangaryan, T. Mohammadpour, and A. Kianisarkaleh, “Persian Banknote Recognition Using Wavelet and Neural Network,” In Proc. of the International Conference on Computer Science and Electronics Engineering (ICCSEE), Vol. 3, pp. 679-684, 2012, Hangzhou, China.

[29] G. Baykal, U. Demir, I. Shyti and G. Ünal, "Turkish lira banknotes classification using deep convolutional neural networks," 2018 26th Signal Processing and Communications Applications Conference (SIU), Izmir, 2018, pp. 1-4.

[30] J. F. Liu, S. B. Liu, and X. L. Tang, “Long, An algorithm of real-time paper currency recognition,” In J. Comput. Res. Dev. Vol. 40, No. 7, pp. 1057–1061, 2003.

[31] M. Hasanuzzaman, X. Yang, and Y. Tian “Robust and effective component-based banknote recognition for the blind. In IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), Vol. 42, No. 6, 2012, pp. 1021-1030.

[32] J. He, H. Zhang, O. Jin, Z. Hu, and J. Zhang, J. “A Novel Recognition Method for Nigeria Paper Currency based on Support Vector Machine & Genetic Algorithms,” International Symposium on Signal Processing, Biomedical Engineering and Informatics (SPBEI 2013), Vol. 6, pp. 1087-1094, 2013, Hangzhou, China.

[33] R. Vijaya Kumar Reddy, Dr. B. Srinivasa Rao K. Prudvi Raju. “Handwritten Hindi Digits Recognition Using Convolutional Neural Network with RMSprop Optimization.” Proceedings of the Second International Conference on Intelligent Computing and Control Systems IEEE Xplore Compliant Part Number: CFP18K74-ART; ISBN:978-1-5386-2842-3, 2018.

[34] Chia-Ling Huang, Yan-Chih Shih, Chyh-Ming Lai "Optimization of a Convolutional Neural Network Using a Hybrid Algorithm". IJCNN 2019. International Joint Conference on Neural Networks. Budapest, Hungary. 14-19 July 2019.

[35] Arwa Mohammed Taqi, Fadwa Al-Azzo, Mariofanna Milanova, "The Impact of Multi-optimizers and Data Augmentation on TensorFlow Convolutional Neural Network Performance." IEEE Conference on Multimedia Information Processing and Retrieval, 2018.

[36] X. Liu, "A camera phone-based currency reader for the visually impaired," In Proc. of the 10th international ACM SIGACCESS conference on Computers and accessibility ACM, pp. 305-306, 2008, Halifax, Canada.

[37] B.P. Daniel and C.U. Idoko "The Macro-Economic Consequences and Regulatory Challenges of Currency Counterfeiting In Nigeria," In International Journal of Financial Economics Vol. 3, No. 2, pp. 113-120, 2014.

[38] N. Paisios, A. Rubinsteyn, V. Vyas, L. Subramanian, L. "Recognizing currency bills using a mobile phone: an assistive aid for the visually impaired," In Proc. of the 24th annual ACM symposium adjunct on User interface software and technology ACM, pp. 19-20, 2011, New York, USA.

[39] I.A. Doush, and A.B. Sahar, "Currency recognition using a smartphone: Comparison between color SIFT and grayscale SIFT algorithms," In Journal of King Saud University Computer and Information Sciences, 2016.

[40] K. Yoshida, M. Kamruzzaman, F.A. Jewel, and R.F. Sajal, "Design and implementation of a machine vision based but low cost stand alone system for real time counterfeit Bangladeshi banknote detection," In Proc. of the 10th international conference on Computer and information technology, pp. 1-5, 2007, Dhaka, Bangladesh.

[41] T. Ishigaki and T. Higuchi, Detection of Worn-out Banknote by Using Acoustic Signals, Transactions of the Society of Instrument and Control Engineers, Vol. 44, No. 5, pp. 444-449, 2008.

[42] A. Roy, B. Halder, and U. Garain, "Authentication of currency note through printing technique verification," In Proc. of the 7th Indian Conference on Computer Vision, Graphics and Image Processing, pp. 383-390, 2010, Chennai, Dubai.

[43] Z. Ahmed, S. Yasmin, M.N. Islam, and R.U. Ahmed "Image processing based Feature extraction of Bangladeshi banknote," In Proc. of the 8th International Conference of Software, Knowledge, Information Management and Applications (SKIMA), pp. 1-8, 2014, Dhaka, Bangladesh.

[44] C. Herley, P. Vora, and S. Yang, Detection and deterrence of counterfeiting of valuable documents, In Proc. of the International Conference on Image Processing (ICIP'04), pp. 2423-2426, 2004, Singapore, Singapore.

[45] G. Disken, Z. Tüfekçi, L. Saribulut and U. Çevik, "A review on feature extraction for speaker recognition under degraded conditions," In IETE Technical Review, Vol. 34, No. 3, pp. 321-332, 2017.

[46] H. I. Suk. and D. Shen, "Deep learning-based feature representation for AD/MCI classification," In Proc. of the International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 583-590, Springer, September 2013, Berlin, Heidelberg.

[47] H.R. Roth, L. Lu, J. Liu, J. Yao, A. Seff, K. Cherry, L. Kim, and R.M. Summers, "Improving computer-aided detection using convolutional neural networks and random view aggregation," In IEEE Trans. Med. Imag., Vol. 35, No. 5, pp. 1170–1181, May 2016.

[48] Y. Bar, I. Diamant, L. Wolf, S. Lieberman, E. Konen, and H. Greenspan, "Chest pathology detection using deep learning with non-medical training," In Proc. of the International Symposium on Biomedical Imaging (ISBI), pp. 294-297, April 2015, New York, USA.

[49] M. Anthimopoulos, S. Christodoulidis, L. Ebner, A. Christe, and S. Mougiakakou, "Lung pattern classification for interstitial lung diseases using a deep convolutional neural network," In IEEE transactions on medical imaging, Vol. 35, No. 5, pp. 1207-1216, 2016.

[50] Q. Dou, H. Chen, L. Yu, L. Zhao, J. Qin, D. Wang, V.C. Mok, L. Shi, and P.A. Heng, "Automatic detection of cerebral microbleeds from MR images via 3D convolutional neural networks," In IEEE Trans. Med. Imag., vol. 35, no. 5, pp. 1182–1195, May 2016.

[51] A. Setio, F. Ciompi, G. Litjens, P. Gerke, C. Jacobs, S. Van Riel, M. Wille, M. Naqibullah, C. Sanchez and B. Van Ginneken, "Pulmonary nodule detection in CT images using Multiview convolutional networks," In IEEE Trans. Med. Imag., Vol. 35, No. 5, pp. 1160–1169, May 2016.

[52] K. Sirinukunwattana, S.E.A. Raza, Y.W. Tsang, D. R. Snead, I.A. Cree, and N.M. Rajpoot, "Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images," In IEEE Trans. Med. Imag., Vol. 35, No. 5, pp. 1196–1206, May 2016.

[53] D. Wang, A. Khosla, R. Gargeya, H. Irshad and A.H. Beck, "Deep learning for identifying metastatic breast cancer," Quantitative Methods, Cornel University, 2016, Available: arXiv preprint arXiv:1606.05718.

[54] H. Robbins and S. Monro, “A stochastic approximation method,” The Annals of Mathematical Statistics, pp. 400–407, 1951.

[55] P. Jain, S. Kakade, R. Kidambi, P. Netrapalli, and A. Sidford, “Parallelizing stochastic gradient descent for least squares regression: mini-batching, averaging, and model misspecification,” Journal of Machine Learning Research, vol. 18, 2018.

[56] R. Johnson and T. Zhang, “Accelerating stochastic gradient descenusing predictive variance reduction,” in Advances in Neural Information Processing Systems, 2013, pp. 315–323.

[57] A. S. Nemirovsky and D. B. Yudin, Problem Complexity and Method Efficiency in Optimization. John Wiley & Sons, 1983.

[58] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, “Robust stochastic approximation approach to stochastic programming,” SIAM Journal on Optimization, vol. 19, pp. 1574–1609, 2009.

[59] A. Agarwal, M. J. Wainwright, P. L. Bartlett, and P. K. Ravikumar, “Information-theoretic lower bounds on the oracle complexity of convex optimization,” in Advances in Neural Information Processing Systems, 2009, pp. 1–9.

[60] N. L. Roux, M. Schmidt, and F. R. Bach, “A stochastic gradient method with an exponential convergence rate for finite training sets,” in Advances in Neural Information Processing Systems, 2012, pp. 2663–2671.

[61] J. Duchi, E. Hazan, and Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization,” Journal of Machine Learning Research, vol. 12, pp. 2121–2159, 2011.

[62] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in International Conference on Learning Representations, 2014, pp. 1–15.

[63] C. Darken and J. E. Moody, “Note on learning rate schedules for stochastic optimization,” in Advances in Neural Information Processing Systems, 1991, pp. 832–838.

[64] M. D. Zeiler, “AdaDelta: An adaptive learning rate method,” arXiv preprint arXiv:1212.5701, 2012.

[65] T. Tieleman and G. Hinton, “Divide the gradient by a running average of its recent magnitude,” COURSERA: Neural Networks for Machine Learning, pp. 26–31, 2012.

Approval of Investigators

We hereby declare that the research report entitled "**Ethiopian Banknotes Recognition Using Convolutional Neural Network**" is our original work; all sources are duly acknowledged, and the report is compiled by incorporating the necessary comments and suggestions given by the reviewers.

	Name	Signature	Date
Principal Investigator	**Dr. Dereje Tekilu**_____	_____	_____
Co- Investigator	Dr. Harish Kalla_____	_____	_____

Approval of Reviewers

I hereby confirm that (PI)Dr./Mr. _____has accomplished his/her work as per the approved proposal and incorporated all the comments given by the reviewers in his/her terminal report of the project entitled **Ethiopian Banknotes Recognition Using Convolutional Neural Network**_____ and hence the report qualifies for submission as standard research output.

	Name	Signature	Date
Reviewer 1.	_____	_____	_____
Reviewer 2.	_____	_____	_____

Approval: **School Ethical Review Board (School Scientific Committee)**

	Name	Signature	Date
1.	_____	_____	_____
2.	_____	_____	_____
3.	_____	_____	_____
4.	_____	_____	_____