CAPSTONE PROJECT REPORT

(Project Term July-November 2019)

LONG GREEN DETECTION AND CONVERSION

Submitted by

Adhikarla Sri Satya Kireet	**Registration Number :11602210**
Vemuri Ajay	**Registration Number :11613031**
Ravi teja Dimmiti	**Registration Number :11604144**

Project Group Number CSERGC0209

Course Code CSE439

Under the Guidance of

Richa Jain

School of Computer Science and Engineering

PAC FORM

TOPIC APPROVAL PERFORMA

School of Computer Science and Engineering (SCSE)

Program : P132::B.Tech. (Computer Science & Engineering)

COURSE CODE : CSE439 **REGULAR/BACKLOG :** Regular **GROUP NUMBER :** CSERGC0209

Supervisor Name : Richa Jain **UID :** 17688 **Designation :** Assistant Professor

Qualification : _____ **Research Experience :** _____

SR.NO.	NAME OF STUDENT	REGISTRATION NO	BATCH	SECTION	CONTACT NUMBER
1	Vemuri Ajay	11613031	2016	K1614	9010510021
2	Ravi Teja Dimmiti	11604144	2016	K1611	9491368588
3	Adhikarla Sri Satya Kireet	11602210	2016	K1614	9492183868

SPECIALIZATION AREA : Networking and Security **Supervisor Signature:** _____

PROPOSED TOPIC : Long green detection and conversion

Qualitative Assessment of Proposed Topic by PAC		
Sr.No.	**Parameter**	**Rating (out of 10)**
1	Project Novelty: Potential of the project to create new knowledge	7.00
2	Project Feasibility: Project can be timely carried out in-house with low-cost and available resources in the University by the students.	7.00
3	Project Academic Inputs: Project topic is relevant and makes extensive use of academic inputs in UG program and serves as a culminating effort for core study area of the degree program.	7.85
4	Project Supervision: Project supervisor's is technically competent to guide students, resolve any issues, and impart necessary skills.	7.85
5	Social Applicability: Project work intends to solve a practical problem.	7.15
6	Future Scope: Project has potential to become basis of future research work, publication or patent.	7.00

PAC Committee Members		
PAC Member (HOD/Chairperson) Name: Dr. Deepak Prashar	UID: 13897	Recommended (Y/N): Yes
PAC Member (Allied) Name: Vishu	UID: 18807	Recommended (Y/N): Yes
PAC Member 3 Name: Ravishanker	UID: 12412	Recommended (Y/N): Yes

Final Topic Approved by PAC: **Long green detection and conversion**

Overall Remarks: Approved

PAC CHAIRPERSON Name: 11024::Amandeep Nagpal **Approval Date:** 07 May 2019

11/14/2019 3:10:38 PM

DECLARATION

We hereby declare that the project work entitled “Long Green Detection and Conversion” is an authentic record of our own work carried out as requirements of Capstone Project for the award of B.Tech degree in Computer Science and Engineering from Lovely Professional University, Phagwara, under the guidance of Richa Jain during July to November 2019. All the information furnished in this capstone project report is based on our own intensive work and is genuine.

Project Group Number: CSERGC0209

Name of Student 1: Adhikarla Sri Satya Kireet

Registration Number: 11602210

Name of Student 2: Vemuri Ajay

Registration Number: 11613031

Name of Student 3: Ravi teja Dimmiti

Registration Number: 11604144

(Signature of Student 1)
Date:

(Signature of Student 2)
Date:

(Signature of Student 3)
Date:

CERTIFICATE

This is to certify that the declaration statement made by this group of students is correct to the best of my knowledge and belief. They have completed this Capstone Project under my guidance and supervision. The present work is the result of their original investigation, effort and study. No part of the work has ever been submitted for any other degree at any University. The Capstone Project is fit for the submission and partial fulfillment of the conditions for the award of B.Tech degree in Computer Science and Engineering from Lovely Professional University, Phagwara.

Signature and Name of the Mentor

Designation: Assistant Professor

School of Computer Science and Engineering,
Lovely Professional University,
Phagwara, Punjab.

Date:

ACKNOWLEDGEMENT

We humbly take this opportunity to present out vote of thanks to all those guideposts who really acted as lightening pillars to enlighten our way throughout this project that has led to successful and satisfactory completion of this study.

We are grateful to our HOS Mr. Amandeep Nagpal and our mentor Richa Jain for providing us with this opportunity to undertake this project in this university and providing us with all the bright and innovative ideas for making our project as worthwhile of running in an organization. We are highly thankful for her active support, valuable time and advice, whole-hearted guidance, sincere cooperation and pains-taking involvement during the study and in completing the capstone project within the time stipulated.

We are thankful to all those, particularly our friends, who have been instrumental in creating proper, healthy and conductive environment and including new and fresh innovative ideas for us during the project, without their help, it would have been extremely difficult for us to prepare the project in a time bound framework.

Adhikarla Sri Satya Kireet

Vemuri Ajay

Ravi teja Dimmiti

Table of Content

List of Figures

List of Tables

1. INTRODUCTION

There are 195 countries in the world today, with each of them looking totally different. For example, the size of the paper is different, the same as the color and pattern also vary from each other. The staffs who work for the money exchanging must distinguish different types of currencies and that is not an easy job. They must remember the symbol of each currency. This may cause some problems (e.g. false recognition), so they need an efficient and exact system to help their work. As we mentioned before, the aim of our system is to help people to recognize different currencies, and work with convenience and efficiency. For people working in banks there is a "Currency Sorting Machine" which helps them to recognize different kinds of currencies. The main working of "Currency Sorting Machine" are image acquisition and recognitions. It is accurate and highly efficient. But for most staffs, they must remember a lot of different characteristics and anti-fakes label for different commonly used currencies in their mind. However, each of them has a handbook which contains the characteristics and anti-fakes labels of some less commonly used currencies. Even for that, no one can ever be 100 per cent confident about the manual recognition. The system that we developed is based on image processing and techniques which include filtering, edge detection, segmentation, etc. In our system, we took Indian RUPEE, American DOLLAR and Malaysian RINGGIT as examples. The system will be programmed based on PYTHON OPENCV and include a user-friendly interface. The main steps in the system are:

1. Read image, reading the image we get from database in the format of JPEG/JPG.

2. Removing noise, pre-processing, smoothening image.

3. Edge detection, segmentation, pattern matching.

4. Results printing. Images can be read from different derivations.

However, we delimit our system which can read the currency from database. The device that the system uses is very common in our daily life, so we do not need to buy an extra device to realize the system. Some similar recognition systems such as face recognition system, fingerprint recognition system also work on the same process. The theories they use are similar, but the techniques and approaches vary from each other.[1]

2. PROFILE OF THE PROBLEM

Around 190+ currencies are available around the world and the need for an automated system related to currencies has been increasing exponentially recently. The need for developing systems that process notes without human intervention for various uses has been pivotal for the development of systems that help in detecting and recognizing currency notes. However, the varying features in each note and the security aspects involved in different currencies make this task extremely difficult. Various systems have been proposed in the past that consider different features such as aspect ratio and HSV values. Methods such as pattern matching have been proposed to develop a system that uses a single algorithm for all the currencies. However not a single method has proven to be efficient enough for actual development thereby making this problem statement an interesting area of research.

One of the first methods proposed to identify the currency notes using image processing techniques was in the early 90's. It has been assumed that the notes are in good condition and images as desired are obtained. It is noteworthy to mention that the system proposed requires the input images to be taken in a predefined angle and distance. The system proposed then applies a series of pre-processing steps on the input images and then extracts certain features such as hue, saturation and value parameters in order to compute a Euclidean distance using these values and compare them with the values that are used as standards. Though this method tries to propose an overall algorithm for all the currencies, it is not an efficient method to identify the notes as certain notes across countries have similar features.

The overall objective of our project is to recognize the currency notes along with their values. We have implemented this currency recognition using TensorFlow's Object Detection API to train an object detection classifier for multiple objects on Windows 10,8, or 7.

3. EXISTING SYSTEM

3.1 INTRODUCTION

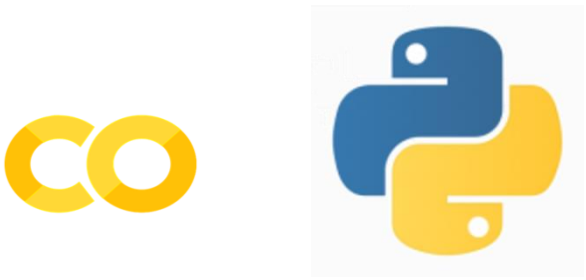
There are many currency recognizing machines that are available in current market through which currency can be recognized whether by using image processing technique or neural networks. All the existing currency recognition systems are mainly based on processing of image using image processing techniques and neural networks. Some system uses Gaussian function in hidden layer and output layer of NN in the place of sigmoid function. Systems have shown that the Gaussian function is more effective than sigmoid function for the recognition of known features and rejection of unknown patterns.

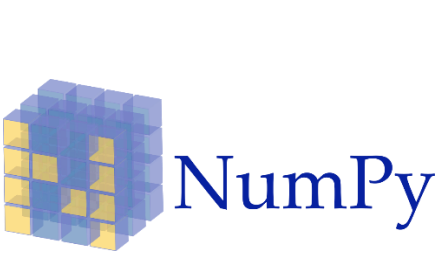
3.2 EXISTING SOFTWARE

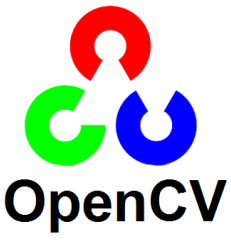
Any PC or laptop with good working condition with a processors like i3,i5 and above and with python compiling IDE software 3 and the latest version with few necessary package/modules installed in order to perform image recognition using CNN. And the packages are

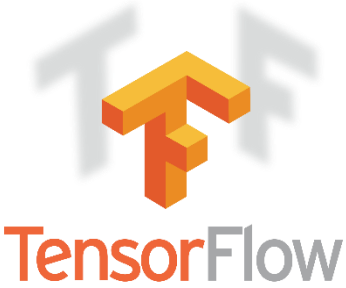
- NumPy
- Faster R-CNN
- OS
- Math
- Open CV
- TensorFlow
- Kera’s model

There is no exact system existing, but one application has been introduced as a mobile system for currency recognition that can recognize the Jordanian currency in different perspective views and scales. For example, images may not be ideally oriented and may have scale changes due to the variation of distance from the camera and illumination changes. Banknote recognition in a variable environment is a complex problem because we have many uncontrolled conditions that affect the image quality. They developed a smart application to identify currencies that are partially visible, folded, wrinkled or even worn by usage. [4]









3.3 DFD FOR PRESENT SYSTEM

Data flow diagram graphically representing the functions, or process which can create and manipulate to store the data and distribute the data between a system and its environment and between components of a system. This makes a meaning full communication between the user and system designer.

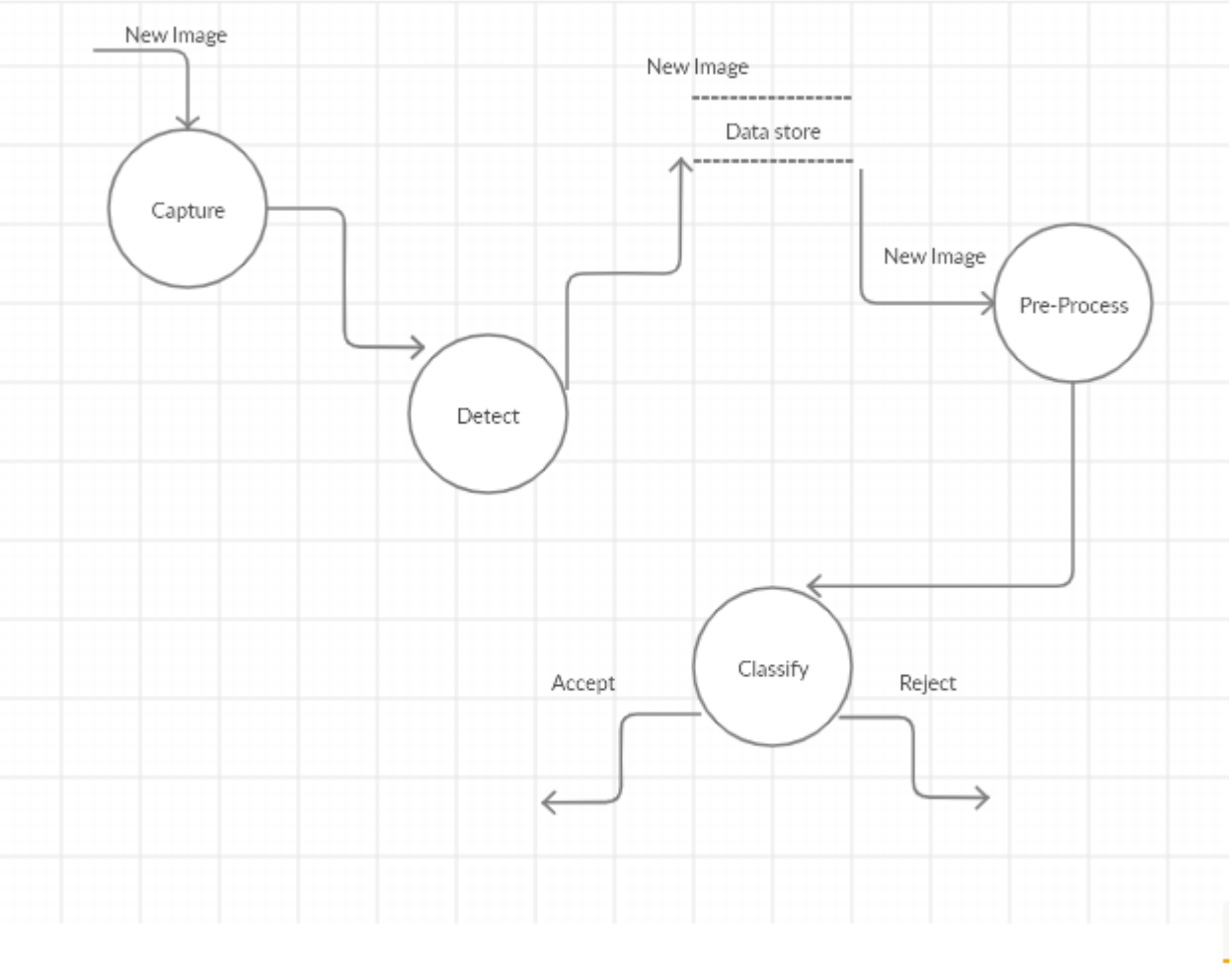


Figure 1 Level 0 DFD

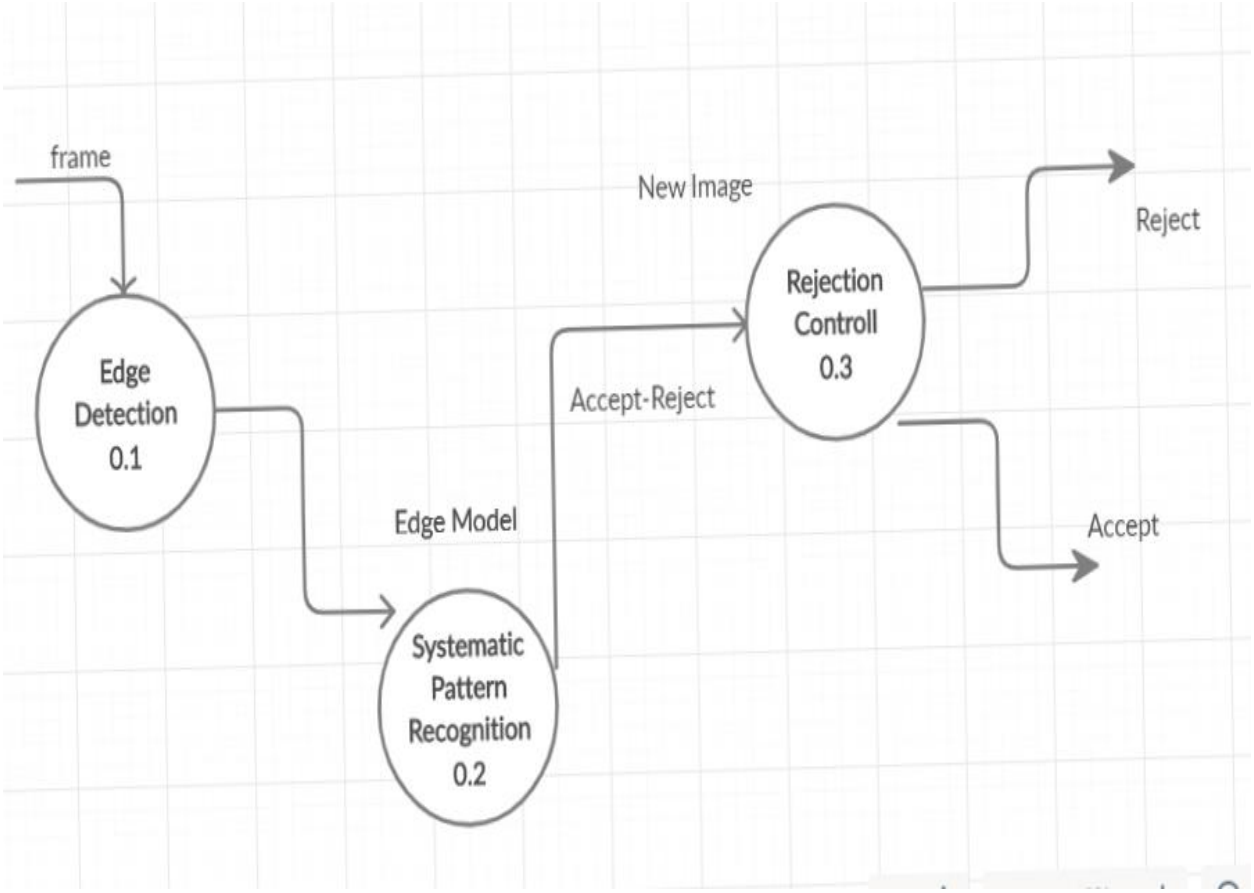


Figure 2 Level 1 DFD

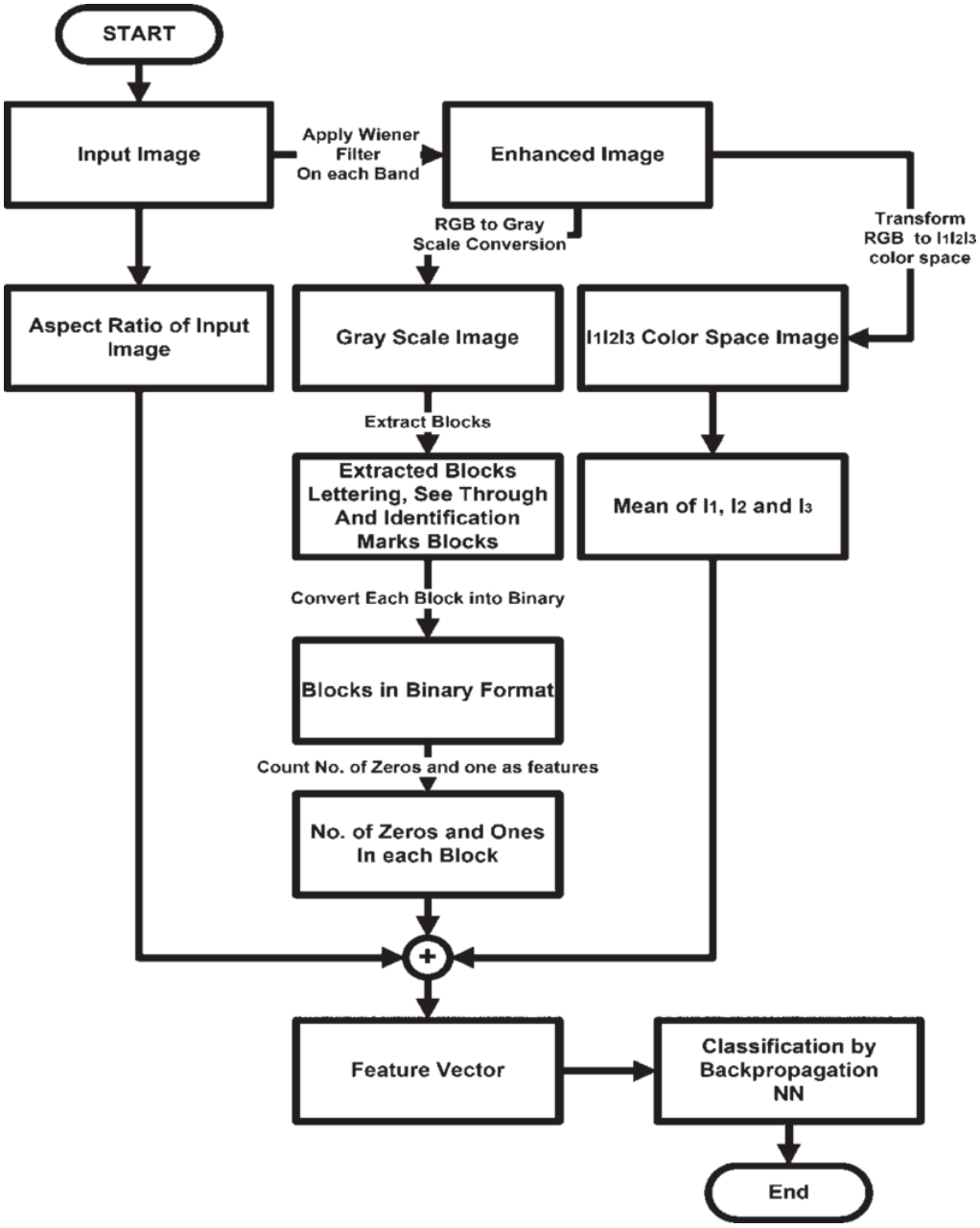


Figure 3 Block diagram representing the process

3.4 What's new in the system to be developed

The proposed approach provides good result in the case of paper banknotes except some cases as:

- Too wrinkled: it is difficult to match the exact descriptors (some key-points do not appear or changed).
- Folded several times: it is difficult to match the exact descriptors (some key-points do not appear or changed).
- Take image from close distance: crop large portion of the image which means it is difficult to match the exact descriptors (many key-points do not appear).
- Take image from too far distance: crop large portion of the background which means it is difficult to match the exact descriptors (large numbers of redundant key-points appear).

Currency recognition in an uncontrolled environment (i.e., using a smartphone) is not an easy task because of the multiple variable conditions that could affect the quality of the image. The experimental results have shown the effectiveness of SIFT algorithm in general for Jordanian banknote recognition, although our algorithm is tested in a more challenging dataset with the images taken in different conditions.

4. PROBLEM ANALYSIS:

4.1 PRODUCT DEFINITION

The purpose of our system is to explain how to train your own convolutional neural network object detection classifier for multiple objects, starting from scratch. At the end of this project, you will have a program that can identify and draw boxes around specific objects in pictures, videos, or in a webcam feed.[3]

There are several good tutorials available for how to use TensorFlow's Object Detection API to train a classifier for a single object. These usually assume you are using a Linux operating system. If you're like me, you might be a little hesitant to install Linux on your high-powered gaming PC that has the sweet graphics card you're using to train a classifier. Setting up TensorFlow to train a model on Windows, there are several workarounds that need to be used in place of commands that would work fine on Linux. Also, this project provides instructions for training a classifier that can detect multiple objects, not just one.

TensorFlow-GPU allows your PC to use the video card to provide extra processing power while training, so it will be used for this project. Using TensorFlow-GPU instead of regular TensorFlow reduces training time by a factor of about 8 (3 hours to train instead of 24 hours). The CPU-only version of TensorFlow can also be used for this project, but it will take longer. If you use CPU-only TensorFlow, you do not need to install CUDA and cuDNN.

4.2 FEASIBILITY ANALYSIS

4.2.1 Technical Feasibility

Technical feasibility of currency recognition needs to follow these aspects

- First, we need to ensure that the monetizing technique we are using is worthy or not. How we get fetch data for it. We trained images from google to ensure the capacity of the model.

- Training images to system may be a difficult task because all images vary in sizes and shapes. To train and test all image files we must be ensure that labelling is done perfectly.
- While user is requesting to detect any currency of a country the system should be able to access the information from previously trained models and able to recognize the given input accurately.
- Our system must be able to handle the given inputs optimally and accurately.

4.2.2 Economical feasibility

In economic feasibility we see that if we need to make our project live, we need to consider the cost needed and what hardware requirements are used. It was interesting to develop the system and was not that difficult as all the required material was available online and the mentor also helped whenever needed in her best way.

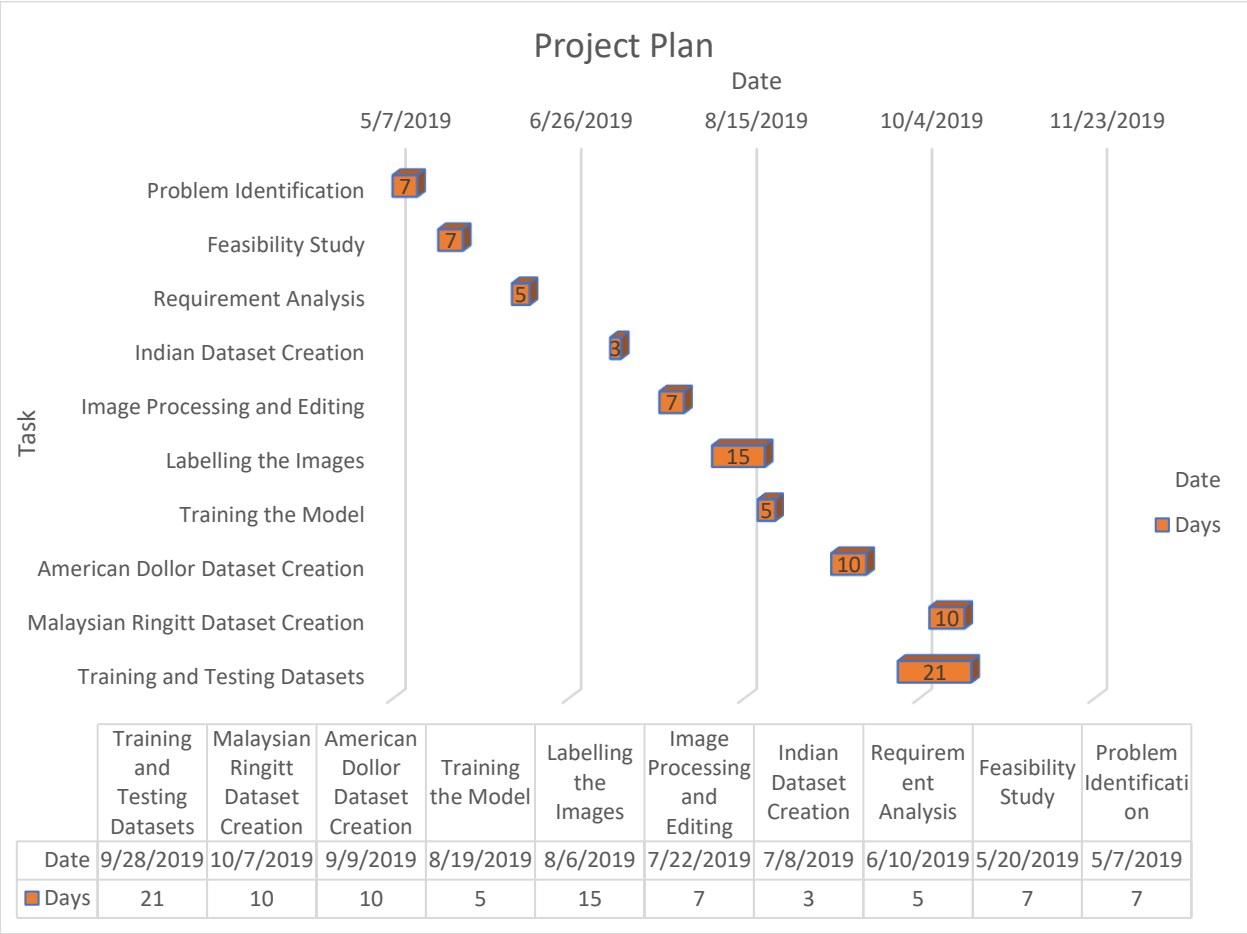
4.2.3 Operational Feasibility

After the completion of the project we need to have some operations that need to be done time to time. In this project we need to work on these following operational activities.

- Upgraded database: This job needs to be done frequently in order to get accurate output from these datasets.
- Database backup: This is very important to back up the website because there could be a chance to lose the data due to any hardware and software failure.
- Training the database on regular basis: This work is done to improve the model and make the machine to learn the images accurately. This results in best output.

4.3 PROJECT PLAN

The project was developed in a period of 12 weeks. We set our schedule according to days. We have spent daily few hours on our project. We got less than 90 days to complete it. Our schedule is as follow:



	Training and Testing Datasets	Malaysian Ringitt Dataset Creation	American Dollor Dataset Creation	Training the Model	Labelling the Images	Image Processing and Editing	Indian Dataset Creation	Requirement Analysis	Feasibility Study	Problem Identification
Date	9/28/2019	10/7/2019	9/9/2019	8/19/2019	8/6/2019	7/22/2019	7/8/2019	6/10/2019	5/20/2019	5/7/2019
Days	21	10	10	5	15	7	3	5	7	7

Figure 4 Project Plan

5. SOFTWARE REQUIREMENT ANALYSIS

5.1 INTRODUCTION

The user requires the system to help them in the best way to recognize their required currencies and their values. These are the technologies that we need:

1. **Install Anaconda, CUDA, and cuDNN:** Download and install the CUDA and cuDNN versions for the latest TensorFlow version, rather than CUDA v8.0 and cuDNN v6.0

2. **Set up TensorFlow Directory and Anaconda Virtual Environment:** Create a folder directly in C: and name it "tensorflow1". TensorFlow Object detection framework will be contained in this directory, as well as your training images, training data, trained classifier, configuration files, and everything else needed for the object detection classifier.

   The TensorFlow model's repository's code is continuously updated by the developers. Some changes that were made breaks the functionality with old versions of TensorFlow. It is always best to use the latest version of TensorFlow and download the latest model's repository. If you are not using the latest version, clone or download the updated version.

5.2 GENERAL DESCRIPTION:

- **General Description of TensorFlow:** The name "TensorFlow" is derived from the operations which neural networks perform on multidimensional data arrays or tensors! It's literally a flow of tensors. It is used to design, build, and train deep learning models and is the second machine learning framework that Google created. TensorFlow library can be used to do numerical computations, which doesn't seem all too special, but these computations are done with data flow graphs. Mathematical operations are represented by these nodes, while the edges represent the data, which usually are multidimensional data arrays or tensors, that are communicated between these edges.

- **General Description of Protobuf:** Protocol buffers are those which include google's language-neutral, platform-neutral, extensible mechanism for serializing structured data – think XML, but smaller, faster, and simpler. Structuring of data can be done once, then you can use special generated source code to easily write and read your structured data to and from a various variety of data streams and by using a variety of languages.

- **General Description of Anaconda Environment: Anaconda distribution** comes with more than 1,500 packages as well as the conda package and virtual environment manager. It also includes a Graphical User Interface, **Anaconda Navigator**, as a graphical alternative to the command line interface (CLI). The big difference between conda and the pip package manager is in how package dependencies are managed, which is a significant challenge for Python data science and the reason conda exists.

  When a package is installed by pip, it automatically installs any dependent Python packages without checking if these conflict with previously installed packages. Regardless of the state of the existing installation, it will install a package and any of its dependencies. For example, Google TensorFlow, can find that it stops working having used pip to install a different package that requires a different version of the NumPy library than the one used by TensorFlow. Package may produce different results in some cases.

- **General Description of Matplotlib:** Matplotlib is an amazing visualization library in Python for 2D plots of arrays. It is built on NumPy arrays and designed to work with the broader SciPy stack. John Hunter introduced this in the year 2002.

  Greatest benefits of visualization are that it allows us visual access to huge amounts of data in easily digestible visuals. Several plots like line, bar, scatter, histogram etc. are contained in matplotlib.

- **General Description of Pandas:** Pandas is an opensource library that allows to you perform data manipulation in Python. It is built on top of NumPy, meaning Pandas needs NumPy to operate. Provides an easy way to create, manipulate and wrangle the data. It is also an elegant solution for time series data.

Data scientists use Pandas for its following advantages:

- Easily handles missing data.
- It uses Series for one-dimensional data structure and Data Frame for multi-dimensional data structure.
- It provides an efficient way to slice the data.
- It provides a flexible way to merge, concatenate or reshape the data.

General Description of Opencv-Python: OpenCV was started at Intel in 1999 by Gary Brad sky, and the first release came out in 2000.Algorithms related to Computer Vision are supported by OpenCV and Machine Learning and are expanding day by day.

Programming languages like C++, Python, Java, etc., are supported by OpenCV and is available on different platforms including Windows, Linux, OS X, Android, and iOS.

OpenCV-Python can be considered as Python API for OpenCV, combining the best qualities of the OpenCV C++ API and the Python language.

OpenCV-Python uses **NumPy**, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the array structures of OpenCV are converted to and from NumPy arrays. This makes it easier to integrate with other libraries that use NumPy such as SciPy and Matplotlib.

5.3 SPECIFIC REQUIREMENTS:

5.3.1. S/w and H/w requirement:

1) Environment:

• **Operating System:** - Microsoft Windows 2007 or Higher

• **Data Base Server:** MySQL

• **Clients:** Microsoft Internet Explorer, Google chrome

• **Tools**: Notepad ++, WordPress, Microsoft office, Anaconda.

• **User Interface:** IDLE (Python 3.6 32-bit), Jupyter Notebook.

Hardware Requirements:

Number	Description
1	PC with 4 GB RAM and 1TB Hard disk

Table 1 Hardware Requirements

Software Requirements:

Number	Description
1	Windows 2007 or higher
2	MySQL
3	Notepad++
4	WordPress
5	Jupyter Notebook

Table 2 Software Requirements

6. DESIGN

The design phase of this paper describes how recognition system operates. It describes structure of the system and method of input, output and the various procedures by which the users interact with the currency recognition system for the purpose of detecting currency notes. This is very useful phase because it illustrates how a user of the system works using a certain type of system design method. In this section we adopted the E-R diagram and Data Flow Diagram for the proposed system showing the set of requirements to accomplish it.

6.1 SYSTEM DESIGN

The System Design describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

Input formats

Currency Convertor and Recognizer uses different input formats to process image of currency given by the user to get his/her desired result. Our System supports jpg, jpeg and png formats as input.

Output Layouts

Our System displays the Country name to the input given by user.

Processing Logic

Our System takes the image of Currency specified by the user and it applies the functionalities those are mentioned in functional requirement of our system.

Operating Environment

Our System Supports Mozilla Firefox 3.0 and above versions. Web Browser Chrome, Opera and other Browsing Machines.

6.2 DESIGN NOTATIONS

DFD: Data Flow Diagram

GUI: Graphical User Interface

CNN: Convolutional neural network

Faster R-CNN: Canonical model for deep learning-based object detection

6.3 DETAILED DESIGN

End-to-end test classification pipeline is composed of following components

1. **Training text:** It is the input image through which our supervised learning model is able to learn and predict the required class.

2. **Feature Vector:** A feature vector is a vector that contains information describing the characteristics of the input data.

3. **Labels:** These are the predefined categories/classes that our model will predict

4. **ML Algo:** It is the algorithm through which our model is able to deal with image classification (In our case: CNN, RNN, HAN)

5. **Predictive Model:** A model which is trained on the historical dataset which can perform label predictions

Faster R-CNN

Faster R-CNN is now a superior model for Machine learning-based object detection. It inspired to welcome many detection and segmentation models that came after it, including the two others, those examining today. Unfortunately, we can't really begin to understand Faster R-CNN without understanding its own predecessors, R-CNN and Fast R-CNN, so let's make a call to look into its ancestry.

R-CNN

R-CNN is the grandfather of Faster R-CNN. In other words, R-CNN really kicked things off.R-CNN, or Region-based Convolutional Neural Network, consists of three steps:

1. Scan the input image for possible objects using an algorithm called Selective Search, generating ~2000 region proposals
2. Run a convolutional neural net (CNN) on top of each of these region proposals
3. Take the output of each CNN and feed it into a) an SVM to classify the region and b) a linear regressor to tighten the bounding box of the object, if such an object exists.

These 3 steps are illustrated in the image below:

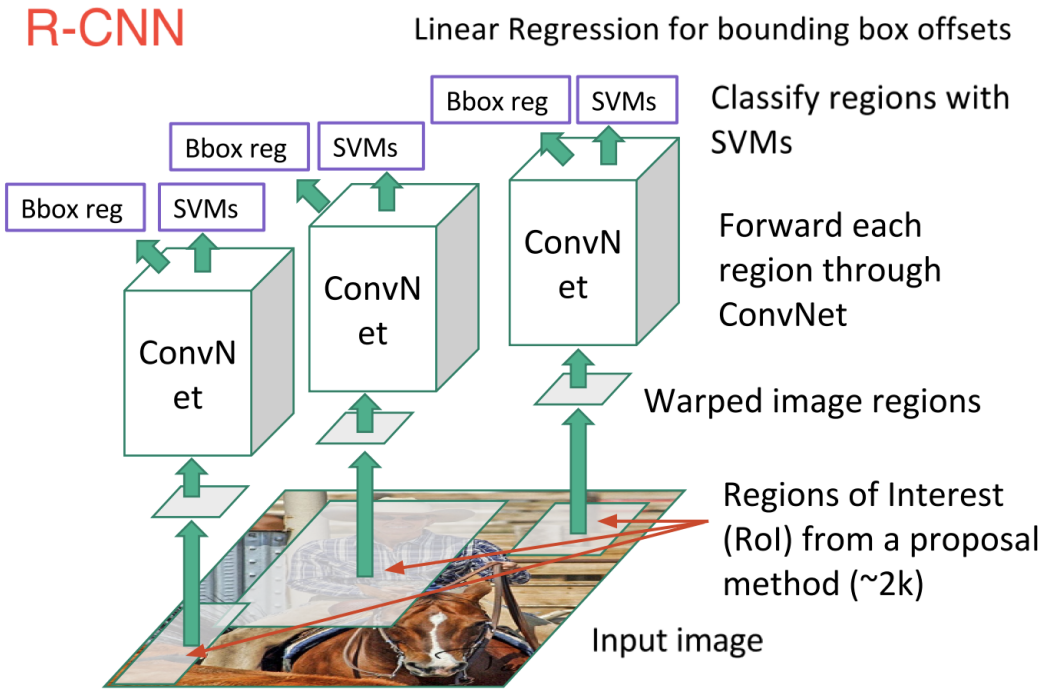


Figure 5 R-CNN

In other words, we first propose regions, then extract features, and then classify those regions based on their features. we made object detection into an image classification problem. R-CNN was very intuitive, but very slow.

Fast R-CNN

R-CNN's immediate descendant was Fast-R-CNN. Fast R-CNN resembled the original in many ways, but improved on its detection speed through two main augmentations:

1. Performing feature extraction over the image before proposing regions, thus only running one CNN over the entire image instead of 2000 CNN's over 2000 overlapping regions

2. Replacing the SVM with a SoftMax layer, thus extending the neural network for predictions instead of creating a new model

The new model looked something like this:

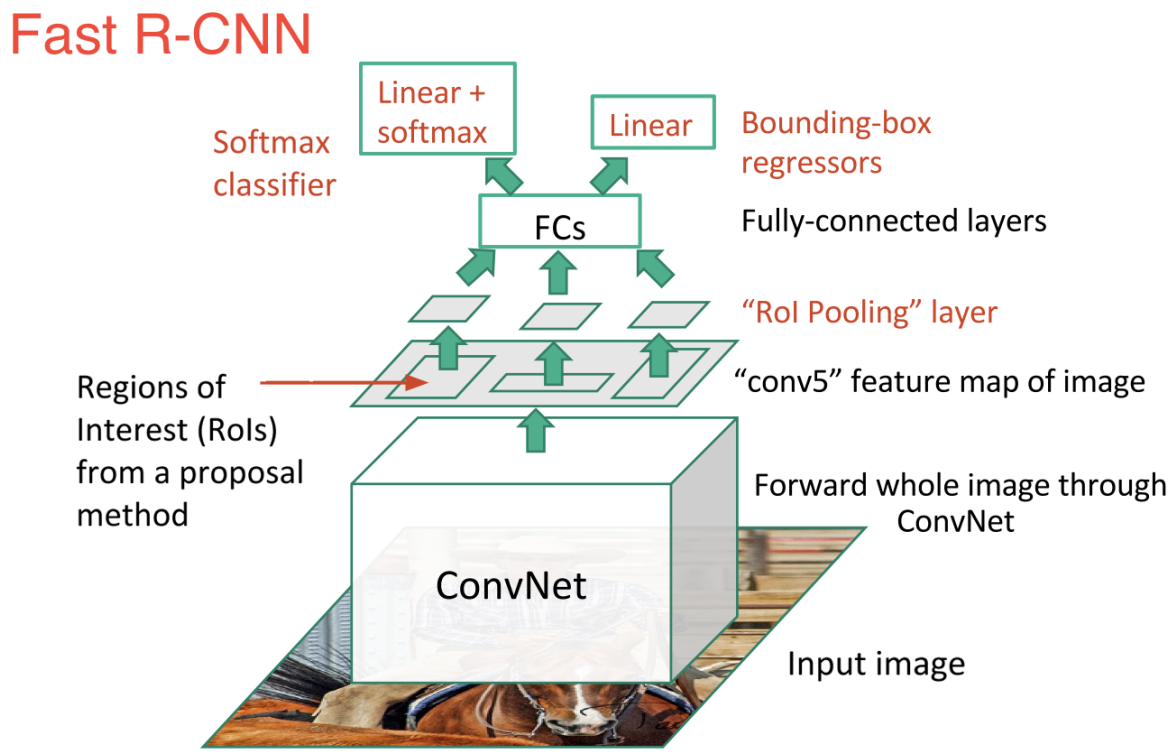


Figure 6 Fast R-CNN

As we can see from the image, we are now generating region proposals based on the last feature map of the network, not from the original image itself. As a result, we can train just one CNN for the entire image. In addition, instead of training many different SVM's to classify each object class, there is a single SoftMax layer that outputs the class probabilities directly. Now we have only one neural net to train, as opposed to one neural net and many SVM's.

Fast R-CNN performed more better in means of speed. There was only one big bottleneck left: the selective search algorithm for generating region proposals.

Faster R-CNN

Now, we're at our original target: Faster R-CNN. The main insight of Faster R-CNN was to replace the slow selective search algorithm with a fast-neural net. Specifically, it introduced the region proposal network (RPN).

Here's how the RPN worked:

- At the final layer of an initial CNN, a 3x3 sliding window moves across the feature map and maps it to a lower dimension.
- For every sliding-window location, it generates multiple possible regions based on k fixed-ratio anchor boxes (default bounding boxes)
- Each region proposal consists of

    a) an "objectless" score for that region.

    b) 4 coordinates representing the bounding box of the region.

In different means, we look at every location in our last feature map and consider *k* different boxes centered around it: a tall box, a wide box, a large box, etc. For each of those boxes, we output whether we think it contains an object, and what the coordinates for that box are. This is how it looks like at one sliding window location.

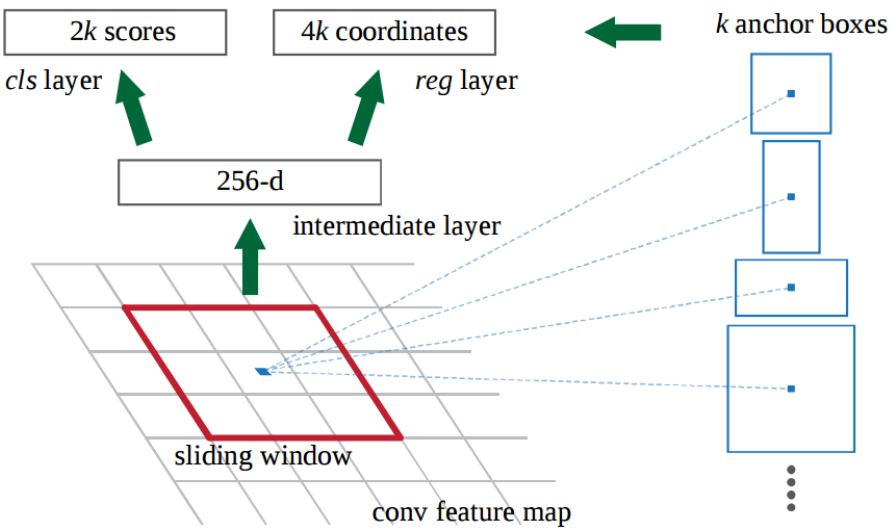


Figure 7 Faster R-CNN

The 2*k* scores represents the SoftMax possibilities of all *k* bounded boxes being on "object." Note that although the RPN outputs bounded box coordinates, it does not try to classify any strong objects: its main job is still proposing object regions. If an anchor box has an "objectless" score above a specified threshold, that box's coordinates passed forward as a region proposal.

Once we are having our region proposals, we feed them directly into what is essentially a Fast R-CNN. We add a pooling layer, fully connected layers, SoftMax classification layer and bounding box regressor. In a sense, Faster R-CNN = RPN + Fast R-CNN.

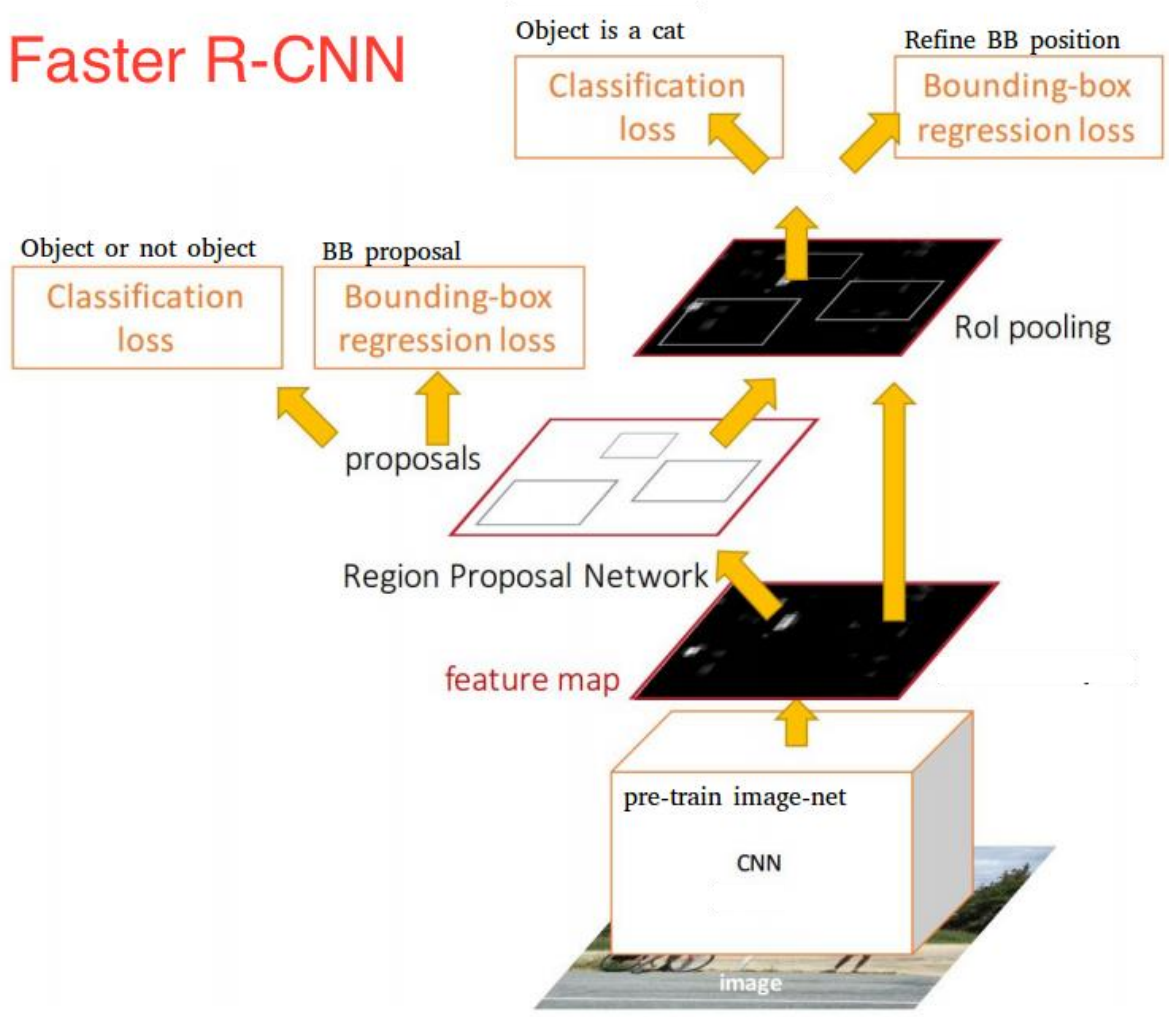


Figure 8 Faster R-CNN Training Model

Altogether, Faster R-CNN achieved better speeds and a state-of-the-art accuracy. It great to mention that, although future models did a lot to increase detection speed, few models unable to perform as Faster as R-CNN by a significant margin. In other words, Faster R-CNN may not be the simplest or fastest method for object detection, but it is still one of the best performing. Case in point, TensorFlow's Faster R-CNN with Inception ResNet is their slowest bust accurate model.

Finally, Faster R-CNN may look complicated, but its core design is the same as the original R-CNN: **hypothesize object regions and then classify them**. This is now the predominant pipeline for many object detections models, including our next one.

6.4 FLOW CHART

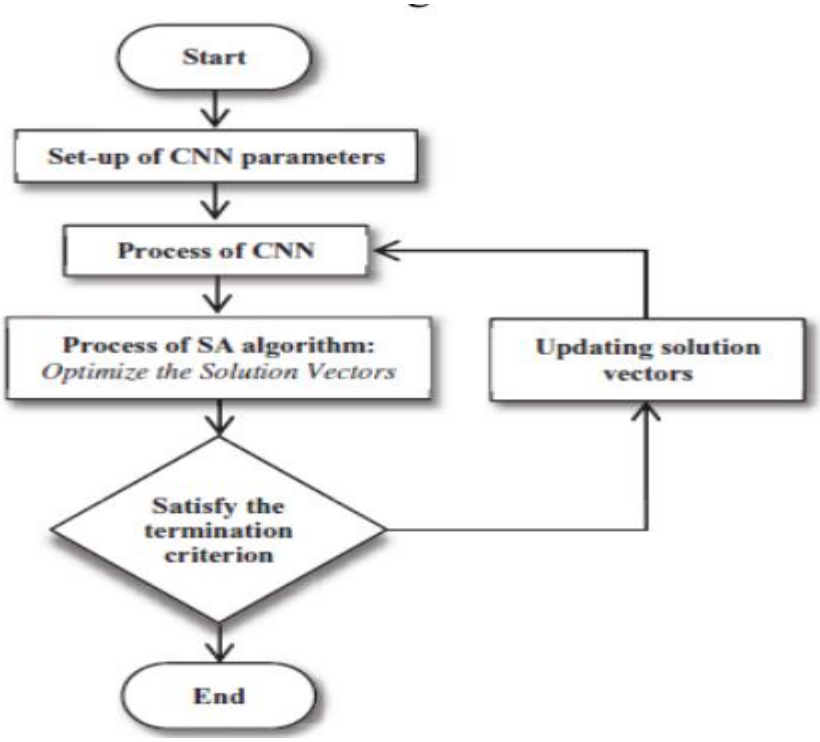


Figure 9 Currency Detection Flow Chart

6.5 PSEUDO CODE

1. Pre-processing of dataset

    1.1 gray scaling and resizing and other preprocessing of every image in dataset manually.

2. Load dataset from local disk

2.1 training_data = [ ] # list

2.2 specifying path

2.3 reading image and image class label training_data. append[images,labels]

3. Separating labels and images

X= [ ] # images

Y= [ ] # labels

4. Creating and training the model (CNN classifier)

5.1 Creating Sequential Model

5.2 Adding Conv2D layer

5.3 Adding Activation Relu layer

5.4 Adding MaxPooling2D layer

5. Prediction

5.1 Browse Input Image

5.2 Input image preprocessing

5.2.1 Input image gray scaling and resizing and other preprocessing.

5.2.2 Line and character segmentation.

5.2.3 Giving each segmented character to give model to classify.

5.2.4 Prediction

6. Predict the accurate output.

7. TESTING

7.1. Introduction

The process of software engineering can be viewed as spiral. Initially system engineering tells the role play by software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Going inside the spiral, we came to design and after coding. To develop recognition system, we spiral in along rationalizes that reduce the level of abstraction on each turn.

A plan for software testing can also be seen in the spiral context. Unit testing initiates at the vertex of the spiral and focusses on each unit of the system as implemented in source code. Testing growths by moving outward along the spiral to integration testing, where the concentration is on the design and the construction of the system architecture. By considering outward on the spiral we come across validation testing where requirements evolved as part of software requirements analysis, are validated opposing the system that has been constructed. Finally, we come to system testing, where the software and other system elements are tested.

7.2. Functional Testing:

Functional testing is a quality determination procedure and a type of black box testing that bases its test cases on the specifications of the system component under test. Functions are tested by serving them input and examining the output, and internal program structure is rarely considered (not like in white-box testing). Functional Testing usually describes what the system does.
Functional testing typically involves five steps:

1. The identification of functions that the system is expected to perform.

2. The input data creation based on the function's specifications.

3. The determination of output based on the function's specifications.

4. The execution of the test case.

5. The comparison of actual and expected outputs.

Black-box testing is a type of software testing that examines the functionality of an application (e.g. what the system does) denying of looking into its internal structures or workings (see white-box testing).This method of test can be applied to virtually every level of software testing: unit, integration, system and acceptance. It comprises of all higher-level testing.
Black box testing intend to explore different types of errors than white box testing:

- Missing functions
- Usability problems
- Performance problems
- Concurrency and timing errors
- Initialization and termination errors.

7.3. Structural Testing:

White-box testing, known as clear box testing, transparent box testing, structural box testing, and glass testing is a type of testing software that tests structures or working of an application, as opposed to its functionality. In white box testing an internal observation of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to train paths through the code and determine the corresponding outputs. This is comparable to testing nodes in a circuit. While white-box testing can be applied at the unit, integration and system levels of the testing process of software, it is usually done at the unit level. It can test paths inside unit, paths between units during integration and subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it might not notice unimplemented parts of the specification or missing requirements.

White-box test design techniques include:

• Control flow testing

• Data flow testing

• Branch testing

• Path testing

• Statement coverage

• Decision coverage

7.4. Levels

7.4.1. Unit Testing

White-box testing is done during performing unit testing to confirm that the code is working as intended, before any integration takes place with previously tested code. White box testing of during performing unit testing catches any faults or errors early on and aids in any defects that happen later on after the code is integrated with the rest of the application and therefore prevents any type of errors that occurs later on.

7.4.2. Integration Testing

At this level, White box testing are performed to test the interactions of each interface with each other. The Unit level testing made sure that every line in code was tested and working correspondingly in an isolated environment and integration examines the correctness of the behavior in an open environment by using white box testing for any interactions of interfaces known to the programmer.

7.5 Project Testing

Now we moved on to project testing after completion of design and coding testing.

Following are the steps performed in testing phase:

1. We labelled the images that were collected from various sources. Labelling is done to make it easy for the system to grasp the images easily.
2. After labelling we started to test the code by giving the training dataset to the model. Around 5-6-hour training was given to the system to train itself from the given set of images.
3. After the training is done, we have done the testing using the test data that was separated previously. The test data also consists of labelled images that will be used and input for the system. And the system needs to predict the output correctly.
4. If the system predicts the output wrong then the testing part can be considered as failed. Again, the model, dataset and system need to be rechecked.
5. System has predicted the output correctly and exactly. Now instead of giving the input through test data we have started giving the input through webcam. System will be able to predict the output correct if the image through webcam is visible and clear.
6. At the we have followed the same way for remaining datasets also.

8. IMPLEMENTATION

8.1 Design Approach

The first step in the development phase is design for any techniques and principles for defining a device, a process or system in enough detail to realize its physical realization. After the analyzation and specification of software requirements, the software design involves three activities design, coding, implementation and testing that are required to build and verify every software. The design activities are most important in this phase, because in this activity, decisions will be affecting the successful implementation of the software and its ease of maintenance are made. These decisions have responsibility of reliability and maintainability of the system. Design is the best way to transform the user requirements into finished software or a system. Design is the phase where quality is measured in development. Software design is a process through which requirements are translated into a software representation.

Software design involves two steps. Preliminary design is concerned with the transformation of requirements into data.

8.1.1Software Architectural Design

8.1.1.1 Implementing Front End

Our System follows the single tier Architecture, which includes code and its related libraries inside the python Idle 3.5.2. The code for currency Recognition System is accomplished by using TensorFlow library, which is used for predicting the model sample.

8.1.1.2 Implementing Back End

Tensor Flow provides its own Backend, which is quite easy to use and offers quick solutions to the back-end problems. However, in our case we not only needed Tensor Flow backend, but also, we need Anaconda prompt to run and execute the code to get desired result.

8.1.2 Database Design

Database required for the long green detection and conversion system is gathered from Wikipedia, google and on live camera basis. Gathered Data quirpele is done using Microsoft paint.

8.2 Conversion Plan

The idea of building this system stroked us when we observed many people go to different countries for their respective works, they cannot able to find and recognize the currency of that country for their uses. It makes difficult to accommodate without handing long green in hands.

In order to avoid these situations, we developed a system, which helps them to find and recognize the currency of different countries.[2]

8.3 Post Implementation and Software Maintenance

The main focus after the implementation is to increase the user base and to involve more and more people into our project, since our project is prediction based it becomes very important to have good and meaningful predictive practices on the system in order to run it well.

A post implementation review measures the systems performance against pre-defined requirements of the system. It determines how well the system will continues to work to meet performance specifications.

In software maintenance we must have to check that whether the software have their correct working or not. It is used to improve performance of the software and provide better results. It is concerned with modifying software once it is delivered to the users or we can say that the process of modifying the software or components after delivery.

9. PROJECT LEGACY

9.1 Current status of the project

The project is successfully converted into an executive software where user can choose the currency denominations of different countries to recognize its value and the country name. The system is secure and does not require any personal information of the user or any legitimate credentials. So, the user need not to hesitate before installing it as it is free of any charges to use the system. It can also addition of new features.

9.2 Remaining Areas of Concern

The System can be made better by adding more features like-

- Developing an application to make the system ease to users while they use it.
- Adding Upload option to choose the user choice file from database by the user him/herself.
- Making the system, mobile supportive. So that the user can easily carry the system to everywhere.
- Adding fake currency detection feature to prompt the message to make awareness in users that they are using fake currency.

9.3 Technical and Managerial Lessons Learnt

9.3.1 Technical Lessons Learnt

We got to learn more about Currency Detection and Recognition System development required concepts like-

- Collecting Database
- Database Training

- About TensorFlow library
- Learned Some Machine learning concepts and models
- Get to know about Anaconda in python
- Basic Knowledge on Open-CV python

9.3.2 Managerial Lessons Learnt

- Learned Team Management.
- Learned how to deal with different situations when stuck.
- Learned how to work in group of developers.
- Learned to distribute tasks to members.
- Learned about the leadership quality.

10. User Manual: A complete document (Help Guide) of the software developed.

This section will familiarize you with the overall working of the Image recognition system. This system is very much user friendly to use**.** The following steps guide a new user to use this system. A complete document for users to operate this system

1. Start executing the Python programme from the anaconda environment.
2. Give the input image path in the programme which the image is tested.
3. When we run the programme, it gives the input image output with the prediction as (10 rupees note: 99%).
4. There are different ways that the user can see the output:
    - Image (object_detection_image.py)
    - Webcam (object_detection_webcam.py)
    - Video (object_detection_video.py)
5. You can select any option and run that programme and get the output accordingly.

11. Source Code (wherever applicable) or System Snapshots

Set up TensorFlow Directory and Anaconda Virtual Environment

The TensorFlow Object Detection API requires some instructions to install and is present in our system. It requires several additional Python packages, specific additions to the PATH and PYTHONPATH variables, and a few extra setup commands to get everything set up to run or train an object detection model.

TensorFlow version	GitHub Models Repository Commit
TF v1.7	https://github.com/tensorflow/models/tree/adfd5a3aca41638aa9fb297c5095f33d64446d8f
TF v1.8	https://github.com/tensorflow/models/tree/abd504235f3c2eed891571d62f0a424e54a2dabc
TF v1.9	https://github.com/tensorflow/models/tree/d530ac540b0103caa194b4824af353f1b073553b
TF v1.10	https://github.com/tensorflow/models/tree/b07b494e3514553633b132178b4c448f994d59df
TF v1.11	https://github.com/tensorflow/models/tree/23b5b4227dfa1b23d7c21f0dfaf0951b16671f43
TF v1.12	https://github.com/tensorflow/models/tree/r1.12.0
TF v1.13	https://github.com/tensorflow/models/tree/r1.13.0
Latest version	https://github.com/tensorflow/models

Download the Faster-RCNN-Inception-V2-COCO model from TensorFlow's model zoo

TensorFlow provides several object detections models (pre-trained classifiers with specific neural network architectures). Other models (such as the SSD-Mobile Net model) have an architecture that allows faster detection but with less accuracy rate, while some models (such as the Faster-RCNN model) give slower detection but with more accuracy rate. I initially started with the SSD-MobileNet-V1 model, but it didn't do a very good job identifying the currency notes in my images.

I re-trained my detector on the Faster-RCNN-Inception-V2 model, and the detection worked considerably better, but with a noticeably slower speed.

Figure 10 Detecting Malaysian Currency

Object Detection Repository

This repository contains the images, annotation data, .csv files, and TF Records needed to train all currency notes detector. You can use these images and data to practice by making your own currency notes Detector, also contains python scripts that used to generate the training data. It has python scripts to test the object detection classifier of currency notes on images, videos, or a webcam feed.

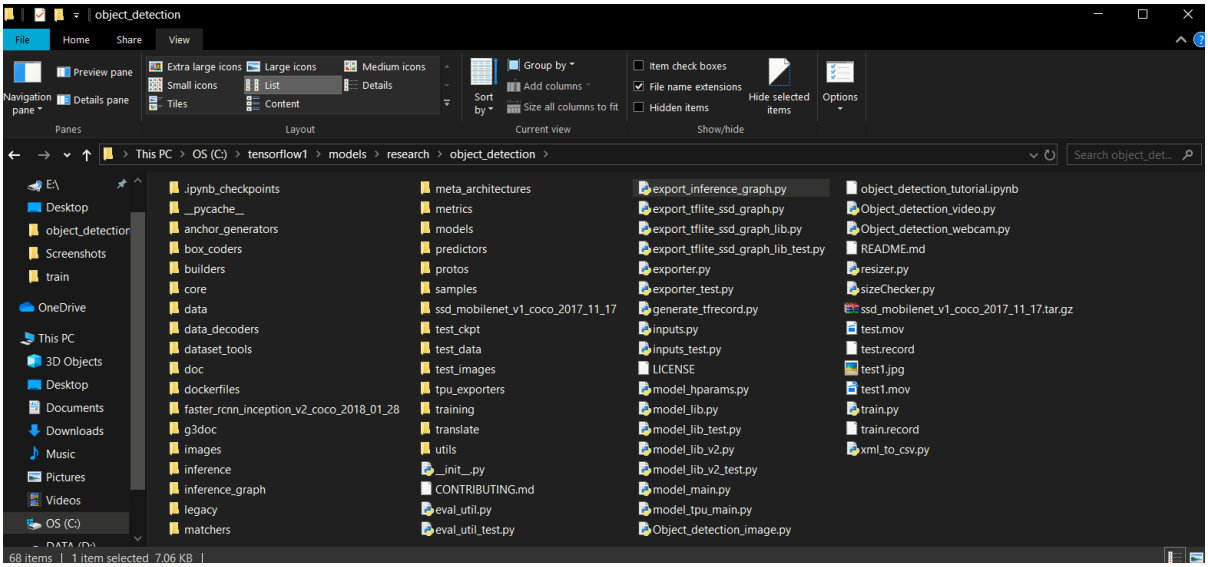


Figure 11 Currency Detection Data Base

If you want to train your own object detector, you need to delete the following files (do not delete the folders):

- All files in \object_detection\images\train and \object_detection\images\test
- The “test_labels.csv” and “train_labels.csv” files in \object_detection\images
- All files in \object_detection\training
- All files in \object_detection\inference_graph

Set up new Anaconda virtual environment

In this we'll work on setting a new virtual environment in Anaconda for tensorflow-gpu. Go to the Start menu in Windows and search for the Anaconda Prompt, right click on it, and click “Run as Administrator”. If Windows asks you if you would like to allow it to make changes to your computer, click Yes.

In the command prompt that is popped up, create a new virtual environment called “tensorflow” by issuing the following command:

C:\> conda create -n tensorflow pip python=3.6

C:\> activate tensorflow

(tensorflow) C:\>python -m pip install --upgrade pip

(tensorflow) C:\> pip install --ignore-installed --upgrade tensorflow-gpu

(tensorflow) C:\> conda install -c anaconda protobuf

(tensorflow) C:\> pip install pillow

(tensorflow) C:\> pip install lxml

(tensorflow) C:\> pip install Cython

(tensorflow) C:\> pip install contextlib2

(tensorflow) C:\> pip install jupyter

(tensorflow) C:\> pip install matplotlib

(tensorflow) C:\> pip install pandas

(tensorflow) C:\> pip install OpenCV-python

Configure PYTHONPATH environment variable

A **PYTHONPATH** variable must be created and that should be pointing to the \models, \models\research, and \models\research\slim directories. Do this activity by giving the following commands:

(tensorflow1)C:\>set PYTHONPATH=C:\tensorflow1\models;C:\tensorflow1\models\research;C:\tensorflow1\models\research\slim

Compile Protobufs and run setup.py

Compile the Protocol buffer files, which are used by TensorFlow to configure model. The short protoc command present on TensorFlow's Object Detection API does not work on Windows. The .proto file in the object_detection\protos directory must be compiled individually by the following command.

cd C:\tensorflow1\models\research

protoc --python_out=..\object_detection\protos\anchor_generator.proto.\object_detection\protos\argmax_matcher.prot.\object_detection\protos\bipartite_matcher.proto.\object_detection\protos\box_code

r.proto.\object_detection\protos\box_predictor.proto.\object_detection\protos\eval.proto.\object_detection\protos\faster_rcnn.proto.\object_detection\protos\faster_rcnn_box_coder.proto.\object_detection\protos\grid_anchor_generator.proto.\object_detection\protos\hyperparams.proto.\object_detection\protos\image_resizer.proto.\object_detection\protos\input_reader.proto.\object_detection\protos\losses.proto.\object_detection\protos\matcher.proto.\object_detection\protos\mean_stddev_box_coder.proto.\object_detection\protos\model.proto.\object_detection\protos\optimizer.proto.\object_detection\protos\pipeline.proto.\object_detection\protos\post_processing.proto.\object_detection\protos\preprocessor.proto.\object_detection\protos\region_similarity_calculator.prot.\object_detection\protos\square_box_coder.proto.\object_detection\protos\ssd.proto.\object_detection\protos\ssd_anchor_generator.proto.\object_detection\protos\string_int_label_map.proto.\object_detection\protos\train.proto.\object_detection\protos\keypoint_box_coder.proto.\object_detection\protos\multiscale_anchor_generator.proto.\object_detection\protos\graph_rewriter.proto.\object_detection\protos\calibration.proto.\object_detection\protos\flexible_grid_anchor_generator.proto

This creates a name_pb2.py file from every name. The proto file in the object_detection\protos folder.

(tensorflow1) C:\tensorflow1\models\research> python setup.py build
(tensorflow1) C:\tensorflow1\models\research> python setup.py install

Test TensorFlow setup to verify it works

The TensorFlow Object Detection API is now all set up to use pre-trained models for object detection, or to train a new one. You can test it and verify your installation is working by compiling the object_detection_tutorial.ipynb script with Jupyter. From the object_detection directory, give this command:

(tensorflow1) C:\tensorflow1\models\research\object_detection> jupyter notebook object_detection_tutorial.ipynb

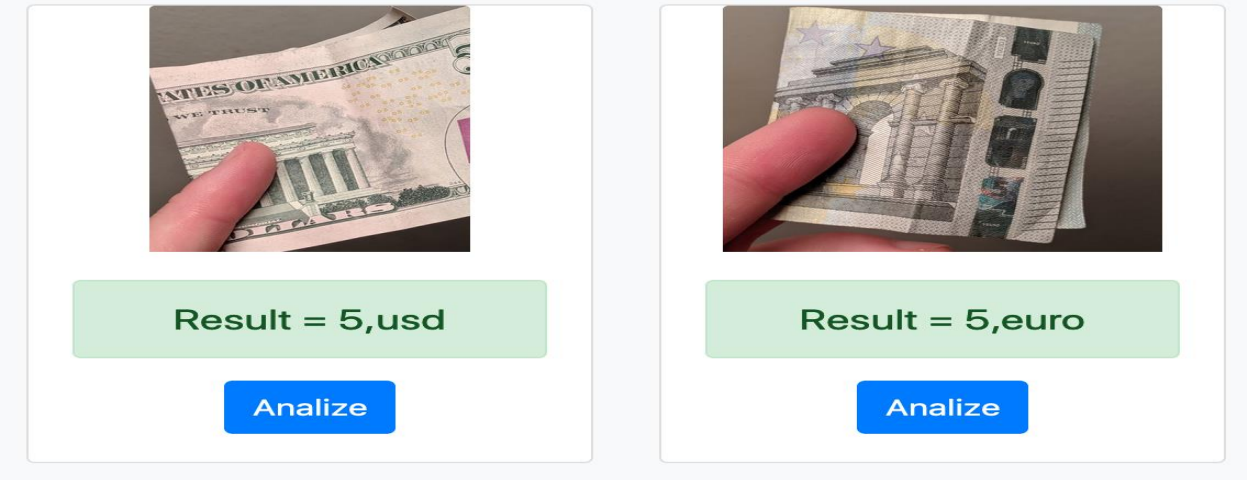


Figure 12 Analysing Currency

Gather and Label Pictures

Gather Pictures

TensorFlow needs a greater number of images of an object to be trained and get good detection classifier. To train a powerful classifier, the training images should have random objects along with the desired objects and should have a variety of backgrounds and lighting effects. There should be some images where the desired object is partially overlapped with something else or only halfway in the picture.

For my currency notes Detection classifier, I have twenty different objects. I used my iPhone to take about 40 pictures of each currency note on its own, with various other non-desired objects in the pictures. Then, I took about another 100 pictures with multiple notes in the picture. I know I want to be able to detect the notes when they're overlapping, so I made sure to have the notes be overlapped in many images.

Figure 13 Currency Dataset Sample

Make sure the images aren't too large. They should be less than 200KB each, and their resolution shouldn't be more than 720x1280. The larger the images are, the longer it will take to train the classifier. You can use the resizer.py python script to reduce the size of the images.

You have all the pictures, now keep twenty percent of the images in the object_detection\images\test directory and remaining eighty percent of the images in object_detection\images\train directory. Make sure there are a different type of pictures in both the test and train directories.

Label Pictures

Now comes the fun part with all the pictures gathered, it's time to label the desired objects in every picture. LabelImg is a great tool for labeling images, and it contains very clear instructions on google how to install and use it.

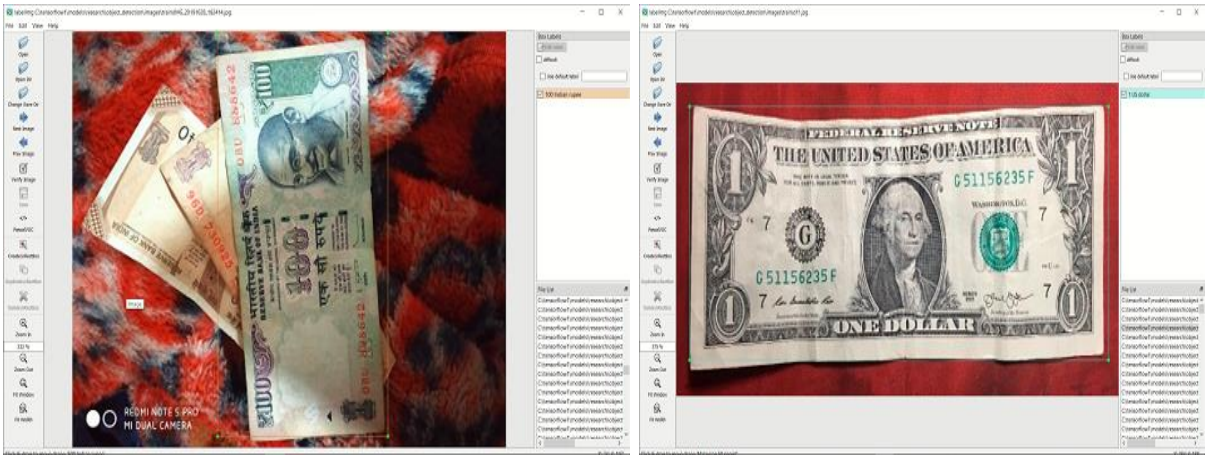


Figure 14 Currency Labelling

Labeling saves a .xml file containing the label data for each image. These .xml files are used to generate TF Records, which are one of the inputs to the TensorFlow trainer. Once you have labelled and saved each image, there will be one .xml file for each image in the test and train directories.

Generate Training Data

First, all the images with .xml data will be converted to create .csv files that contains all the data for the train and test images. From the object_detection folder, give the following command in the Anaconda command prompt:

(tensorflow1) C:\tensorflow1\models\research\object_detection> python xml_to_csv.py

Next, open the generate_tfrecord.py python file. Replace the label map with your own label map, where each object is assigned an ID number. This same number will be assigned when configuring the labelmap.pbtxt file.

```
def class_text_to_int(row_label):
if row_label == '10 Indian rupee:   return 1
    elif row_label == '20 Indian rupee':
        return 2
    elif row_label == '50 Indian rupee':
        return 3
    elif row_label == '100 Indian rupee':
        return 4
    elif row_label == '200 Indian rupee':
        return 5
    elif row_label == '500 Indian rupee':
        return 6
    elif row_label == '2000 Indian rupee':
        return 7
    elif row_label == '1 US dollar':
        return 8
    elif row_label == '2 US dollar':
        return 9
    elif row_label == '5 US dollar':
        return 10
    elif row_label == '10 US dollar':
        return 11
    elif row_label == '20 US dollar':
```

```
    return 1
elif row_label == '50 US dollar':
    return 13
elif row_label == '100 US dollar':
    return 14
elif row_label == 'Malaysian 1 ringgit':
    return 15
elif row_label == 'Malaysian 5 ringgit':
    return 16
elif row_label == 'Malaysian 10 ringgit':
    return 17
elif row_label == 'Malaysian 20 ringgit':
    return 18
elif row_label == 'Malaysian 50 ringgit':
    return 19
elif row_label == 'Malaysian 100 ringgit':
    return 20
else:
    return 0
```

Create Label Map and Configure Training

Label map

The label map tells each image is defined and mapped with different classes and have a unique ID of that class. Use a text editor to create a new file and save it as labelmap.pbtxt in the C:\tensorflow1\models\research\object_detection\training folder. Make sure that the file type is .pbtxt, not the .txt file. In the text editor, copy or type in the label map in the format(sample):

```
 item
{
  id: 1
  name: '5 Indian rupee’
 }
```

Configure training

Finally, the object detection training must be configured. This model defines which model and what parameters will be used for training. This is the last step before running the training model**.**

Run the Training

Here we go to train the objects from the object detection directory, issue the following command to begin training:

python train.py --logtostderr --train_dir=training/ --pipeline_config_path=training/faster_rcnn_inception_v2_pets.config

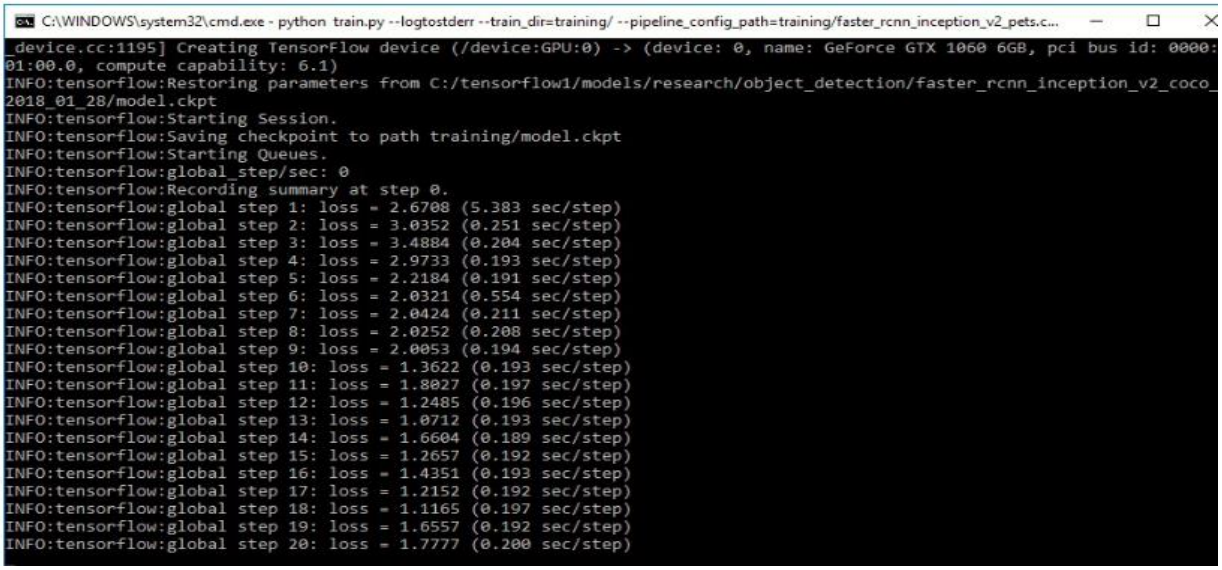
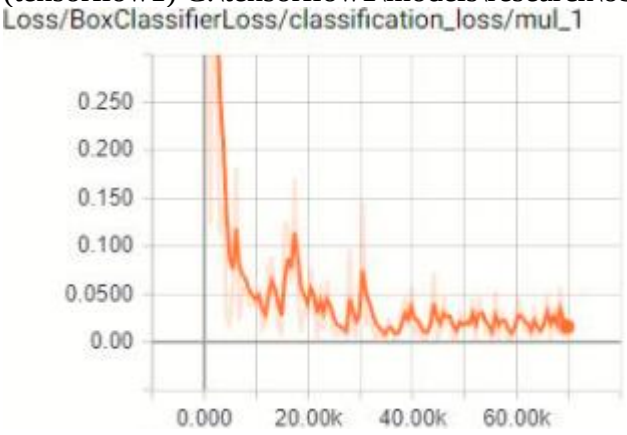


Figure 15 TensorFlow Prompt

You can view the progress of the training by using Tensor Board. To do this, open new Anaconda Prompt, activate the tensorflow virtual environment, change to the C:\tensorflow1\models\research\object_detection directory, and give the following command:

(tensorflow1) C:\tensorflow1\models\research\object_detection>tensorboard –logdir=training



The training periodically saves checkpoints of every five minutes. You can terminate the training by pressing Ctrl+C in the command prompt window.

Export Inference Graph

Now the training is complete, the last step is to generate the frozen inference graph (.pb file). From the object_detection folder, give the following command, where "XXXX" in "model.ckpt-XXXX" should be replaced with the highest-numbered .ckpt file in the training folder:

python export_inference_graph.py --input_type image_tensor --pipeline_config_path training/faster_rcnn_inception_v2_pets.config --trained_checkpoint_prefix training/model.ckpt-20549 --output_directory inference_graph.

Use Your Newly Trained Object Detection Classifier

The object detection classifier is now ready to predict and give accurate output. We've written Python scripts to test it out on an image, video, or webcam feed.

Before running this Python scripts, you need to modify the NUM_CLASSES variable in the script which should be equal to the number of classes you want to detect. (For my currency notes Detector, there are twenty cards I want to detect, so NUM_CLASSES = 20.)

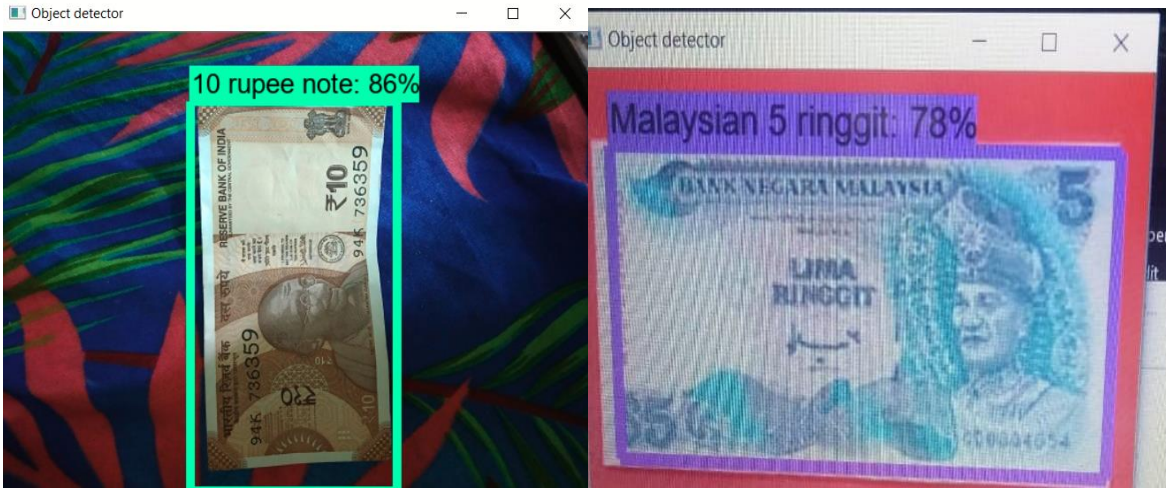


Figure 16 Currency Detection Output Model

12.BIBLIOGRAPHY

Some of the resources from the following website made our work to take best path by overcoming the obstacles like error in the code and helped us to move towards desired output of the model.

[1] Vedasamhitha Abburu, Saumya Gupta, S.R. Rimitha, Manjnath Mulimani, Shashidhar G.Koolagudi. **"**Currency recognition system using image processing" IEEE.

[2] Vipin Kumar Jain, Ritu Vijay, Indian currency denomination identification using image processing technique, 2013.

[3] Binod Prasad Yadav, "Indian Currency Recognition and Verification System Using Image Processing", International Journal of Advanced Research in Computer Science and Software Engineering, vol. 4, no. 12, 2014.

[4] Muhannad Alfarras, "Bahraini paper currency recognition", Journal of Advanced Computer Science and Technology Research, vol. 2, no. 2, pp. 104-115, 2012.

[5] Muhammad Sarfraz, "An Intelligent Paper Currency Recognition System", Procedia Computer Science, vol. 65, pp. 538-545, 2015.