VISVESVARAYA TECHNOLOGICAL UNIVERSITY,

BELAGAVI - 590018

Internship Report on

"AUTOMATIC APPLE FRUIT DETECTION USING DEEP LEARNING"

Submitted in partial fulfillment of the award of Degree of,

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

By

JAYALAKSHMI M **4AL17CS035**

Under the Supervision of

Mr. Hemanth Kumar N P
Senior Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

ALVA'S INSTITUTE OF ENGINEERING AND TECHNOLOGY

MOODBIDRI-574225, KARNATAKA

2020– 2021

ALVA'S INSTITUTE OF ENGINEERING AND TECHNOLOGY

MIJAR, MOODBIDRI D.K. -574225

KARNATAKA

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Internship report on "AUTOMATIC APPLE FRUIT DETECTION USING DEEP LEARNING" submitted by Jayalakshmi M (4AL17CS035) is work done and is submitted during the academic year 2020 – 2021, in partial fulfilment of the requirements for the award of the degree of BACHELOR OF ENGINEERING in DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING of VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The Internship report has been approved as it satisfies the academic requirements in respect of Internship work prescribed for the Bachelor of Engineering Degree.

Internship Guide	**Internship Coordinator**
Department of CS&E	**Department of CS&E**

Head of the Department

Department of CS&E

Examiners

Name of the Examiner	**Signature with Date**
1.	
2.	

ACKNOWLEDGEMENT

First I would like to thank Sri**. PV VINOD, Scientist/Engineer 'SD', RRSC-NRSC, ISRO, ISITE Campus, Bengaluru** for giving me the opportunity to do an internship within the organization.

I would also like to thank all the people who worked along with me in **RRSC-S** with their patience and openness they created an enjoyable working environment.

I am highly indebted to Managing Trustee **Mr. Vivek Alva** and Principal **Dr.Peter Fernandes, Alva's Institute of Engineering and Technology, Mijar** for the facilities provided to accomplish this internship.

I thank my Head of the Department **Dr. Manjunath Kotari, Professor, Department of Computer Science and Engineering** for his constructive criticism throughout my internship.

I am thankful to Internship coordinator, **Mrs. Merlyn Melita Mathias**, **Assistant Professor, Department of Computer Science and Engineering** for her valuable guidance, support and advices to get and complete internship in above said organization.

I would like to thank my Internship guide **Mr. Hemanth Kumar N P Senior Assistant Professor Department of Computer Science and Engineering** for his constant help and support throughout the internship.

I am extremely grateful to my department staff members and friends who helped me in successful completion of this internship.

JAYALAKSHMI M 4AL17CS035

COMPANY PROFILE

REGIONAL REMOTE SENSING CENTRE-SOUTH, NRSC - ISRO, BANGALORE:

Regional Remote Sensing Centre - South (RRSC-S) is the southern regional node of National Remote Sensing Centre, Hyderabad. Over the past three decades the five Regional Remote Sensing Centres, RRSC-C, RRSC-E, RRSC-N, RRSC-S and RRSC-W, have contributed significantly to major National level applications projects and user projects using geospatial technology. The RRSCs also carryout application projects of regional relevance, R&D projects, Technology Development Projects (TDPs), dissemination of Geospatial Services through Bhuvan Geo-portal, Software Development, Capacity Building and Outreach of Space Technology & Applications to cater to the Geo-Spatial needs of the region. Regional themes are identified for each Centre as thrust areas. The regional themes of RRSCS are Horticulture, Urban Studies and Water Resources. Field observations and Location Based Services are also being carried out as part of several projects. RRSC-S covers the states of Karnataka, Kerala, Tamil Nadu, Andhra Pradesh, Telangana, Goa, Pondicherry, Andaman & Nicobar Islands and Lakshadweep Islands. All Regional centers have thematic scientists, Software scientists and Technical manpower to carry out different types of projects. RRSC- South has carried out large number of projects to cater to the development needs of the nation.

भारत सरकार
अंतरिक्ष विभाग
भारतीय अंतरिक्ष अनुसंधान संगठन
राष्ट्रीय सुदूर संवेदन केंद्र
क्षेत्रीय सुदूर संवेदन केंद्र- दक्षिण
आईसाईट परिसर, मारतहल्ली आउटर रिंग रोड
बेंगलूरु-560 037, **भारत**
दूरभाष : 080–61456022, **फैक्स** : 080-61456025

Government of India
Department of Space
Indian Space Research Organisation
National Remote Sensing Centre
Regional Remote Sensing Centre- South
ISITE Campus, Marathahalli Outer Ring Road
Bengaluru- 560 037, India
Phone : 080-61456022, Fax: 080-61456025

RC BG: 13.3 Dec 21, 2020

CERTIFICATE

This is to certify that the following students of ALVA's Institute of Engineering and Technology, Moodbidri, Dakshina Kannada have satisfactorily completed the project work entitled 'Automatic Apple Fruit Detection Using Deep Learning' at Regional Remote Sensing Centre-South, NRSC/ ISRO, Bengaluru during January to February 2020, in partial fulfilment of the requirements for the award of Bachelor of Technology (Computer Science and Engineering) under the guidance and supervision of the undersigned.

1. Ms. Jayalakshmi M (4AL17CS035)

Guided by

P.V. Vinod
Scientist/ Engineer 'SD'
RRSC-South, Bengaluru

Dr. K Ganesha Raj
General Manager
RRSC- South, Bengaluru

ABSTRACT

In this project, an efficient approach has been proposed to localize every clearly visible object from an image. For object detection we have processed every input image to overcome several complexities, to achieve better result we use object detection algorithm. We have also implemented Convolution Neural Network based on object detection model i.e YOLO-V3.Fruit Detection model that is very proficient regardless of different limitations such as high and poor image quality, same fruit but different shape, size and color, multiple overlapped fruits. This model is capable of detecting single fruit separately from a set of overlapping fruits. Nevertheless, taking a number of challenges into consideration, our proposed model is capable of detecting fruits from image with a better accuracy and average precision rate.

TABLE OF CONTENTS

CHAPTER NO.	DESCRIPTIONS	PAGE NO.
	ACKNOWLEDGEMENTS……………….	i
	COMPANY PROFILE……………………	ii
	ABSTRACT………………………………	iii
	LIST OF FIGURES……………………….	iv
	WEEKLY OVERVIEW OF INTERNSHIP ACTIVITIES………………………………	v
1.	**INTRODUCTION**	1-8
	1.1 DEEP LEARNING	2
	1.2 DEEP LEARNING VS MACHINE LERNING	4
	1.3 CONVOLUTION NEURAL NETWORK	5
	1.4 OBJECTIVES	7
2.	**METHODOLOGY**	9-12
	2.1 GOOGLE COLABORATORY	9
	2.2 YOLOV3	9
	2.3 DARKNET	9
	2.4 TRAINING A MODEL	10
3.	**IMPLIMENTATION**	12-16
	3.1 DARKNET	13
	3.2 ANNOTATION	13
	3.3 TRAIN AND TEST SETS	14
	3.4 TRAINING	16
4.	**RESULTS**	17
5.	**CONCLUSION AND FUTURE ENHANCEMENT**	18
	REFERENCES……………………………………	19

LIST OF FIGURES

Figure number	Description	Page number
1.1	PERFORMANCE OF NN	6
1.2	LAYERS OF CNN	7
2.1	TRAINING IMAGE	10
3.1	DARKNET IMPLEMENTATION	13
3.2	LABELLING OF IMAGE	14
3.3	ANNOTATION OBTAINED	14
3.4	WEIGHTS OF TRAINING DATASET	15
3.5	CONFIGURATIONS OF DATASET	16
3.6	TRAINING THE DATASET	17
4.1	APPLE DETECTION(FINAL OUTPUT)	18

ALVA'S INSTITUTE OF ENGINEERING & TECHNOLOGY
A Unit Of Alva's Education Foundation (R), Moodbidri
(Affiliated to VTU, Belgaum, Approved by AICTE, New Delhi, Recognized by Govt. Of Karnataka)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

1st WEEK

DAY	DATE	NAME OF THE TOPIC/MODULE	COMPLETED
Friday	10/01/2020	We met GM of RRSC-ISRO and he assigned a guide, rule and regulation to be followed	Yes

2nd WEEK

DAY	DATE	NAME OF THE TOPIC/MODULE	COMPLETED
Monday	13/01/2020	Discussion about basics python, ML and software we use	Yes
Tuesday	14/01/2020	Discussion about classification of cat or dog using logistic regression	Yes
Wednesday	15/01/2020	We learnt and worked or the examples on logistic regression	Yes
Thursday	16/01/2020	Learning phases of ML and sir explained briefly	Yes
Friday	17/01/2020	Viva on topics covered in this week	Yes

3rd WEEK

DAY	DATE	NAME OF THE TOPIC/MODULE	COMPLETED
Monday	20/01/2020	Difference between machine learning, deep learning and AI	Yes
Tuesday	21/01/2020	Clarification of doubts regarding logistic and linear regression	Yes
Wednesday	22/01/2020	Submitted the 1st assignment (classification of cat or dog)	Yes
Thursday	23/01/2020	Discussed about project topic	Yes
Friday	24/02/2020	Discussed about fruit detection yolo	Yes

4th WEEK

DAY	DATE	NAME OF THE TOPIC/MODULE	COMPLETED
Monday	27/01/2020	Learning how to process on project	Yes
Tuesday	28/01/2020	Learnt about object detection algorithm	Yes
Wednesday	29/01/2020	He thought us about how the model will any object	Yes
Thursday	30/01/2020	Learnt the difference between yolo version	Yes
Friday	31/01/2020	We implemented the basic simple yolo v3 model	Yes

5th WEEK

DAY	DATE	NAME OF THE TOPIC/MODULE	COMPLETED
Monday	03/02/2020	Knowing the techniques used	Yes
Tuesday	04/02/2020	Learnt about Deep Learning	Yes
Wednesday	05/02/2020	Learnt about CNN	Yes
Thursday	06/02/2020	Learnt about libraries of CNN	Yes
Friday	07/02/2020	Learnt about Darknet	Yes

6th WEEK

DAY	DATE	NAME OF THE TOPIC/MODULE	COMPLETED
Monday	10/02/2020	Learned how to run and install darknet using yolo v3	Yes
Tuesday	11/02/2020	Clarification on errors we got in installing the darknet	Yes
Wednesday	12/02/2020	Learnt on Process of detection	Yes
Thursday	13/02/2020	Learnt how to label the image	Yes
Friday	14/02/2020	Downloaded the apple image	Yes

7th WEEK

DAY	DATE	NAME OF THE TOPIC/MODULE	COMP L ETED
Monday	17/02/2020	Labelled the image using LabelImg	Yes
Tuesday	18/02/2020	Cont…on labelling the image	Yes
Wednesday	19/02/2020	Question and answering session	Yes
Thursday	20/02/2020	Processing the project	Yes
Friday	21/02/2020	Final project submission	Yes

8th WEEK

DAY	DATE	NAME OF THE TOPIC/MODULE	COMPLETED
Monday	24/02/2020	Report Preparation	Yes
Tuesday	25/02/2020	Final report submission.	Yes

CHAPTER 1

INTRODUCTION

Computer vision acts as a tool to perceive and grasp knowledge. The three basic works undertaken by a computer vision algorithm include classification, detection and localization. These three fundamental tasks provide both opportunities and challenges to the field of computer vision. They have started seeking the attention of researches and there came a tremendous increase in the work over these fields. One reason for this tremendous growth is the integration of deep convolutional neural network to the field of computer vision. Image classification intends to convert an image to a label. The process of object detection, another important task in computer vision, points to detect the presence of entire objects within an image. For example, driverless cars otherwise called as selfdriving cars are a couple that use a combination of software and sensors, not only to detect the presence of other cars but also the presence of trees, human beings, animals, other vehicles etc. on the road. The third main task in computer vision is called object localization. The localization task is similar to the task of detection.

Object detection deals with detection of a variable number of object classes whereas localization deals with the identification of a fixed number of object classes. The localization task is considered to be a superset of the computer vision task of image classification. With this, we are not only required to know whether there is an object in the image of the given class but also to know where exactly the object is in the image. Object detection is regarded as the first process in most computer vision systems. It has several applications and some of them include security systems, human-computer interactions, robotics, product detection in manufacturing industries, people counting and so on. From the last decade, the field of machine learning has started to move forward in a purposeful way, as a result, the researches have started to concentrate more over these techniques especially for the matter of object detection. Machine learning-based object detection consists of two phases, a training phase as well as a testing phase. An extensively large number of images are used for the purpose of training. Increasing the number of images enables the computer to learn the class of objects correctly and helps to properly identify the objects belonging to different classes. Testing phase involves the task of testing whether the machine responds correctly by giving different inputs or test cases. By giving an image to a computer it tries to learn every bit of object features within the image by a process called feature extraction.

1.1. DEEP LEARNING

Deep learning (also known as deep structured learning) is part of a broader family of machine learning methods based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised.

Deep-learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases surpassing human expert performance. Artificial neural networks (ANNs) were inspired by information processing and distributed communication nodes in biological systems. ANNs have various differences from biological brains. Specifically, neural networks tend to be static and symbolic, while the biological brain of most living organisms is dynamic (plastic) and analog.

The adjective "deep" in deep learning comes from the use of multiple layers in the network. Work showed that a linear perceptron cannot be a universal classifier, and then that a network with a non-polynomial activation function with one hidden layer of unbounded width can on the other hand so be. Deep learning is a modern variation which is concerned with an unbounded number of layers of bounded size, which permits practical application and optimized implementation, while retaining theoretical universality under mild conditions. In deep learning the layers are also permitted to be heterogeneous and to deviate widely from biologically informed connectionist models, for the sake of efficiency, trainability and understandability, whence the "structured" part.

While deep learning was first theorized in the 1980s, there are two main reasons it has only recently become useful:

- Deep learning requires large amounts of labeled data. For example, driverless car development requires millions of images and thousands of hours of video.

- Deep learning requires substantial computing power. High-performance GPUs have a parallel architecture that is efficient for deep learning. When combined with clusters or cloud computing, this enables development teams to reduce training time for a deep learning network from weeks to hours or less.

Deep learning applications are used in industries from automated driving to medical devices.

- Automated Driving: Automotive researchers are using deep learning to automatically detect objects such as stop signs and traffic lights. In addition, deep learning is used to detect pedestrians, which helps decrease accidents.
- Aerospace and Defense: Deep learning is used to identify objects from satellites that locate areas of interest, and identify safe or unsafe zones for troops.
- Medical Research: Cancer researchers are using deep learning to automatically detect cancer cells. Teams at UCLA built an advanced microscope that yields a high-dimensional data set used to train a deep learning application to accurately identify cancer cells.
- Industrial Automation: Deep learning is helping to improve worker safety around heavy machinery by automatically detecting when people or objects are within an unsafe distance of machines.
- Electronics: Deep learning is being used in automated hearing and speech translation. For example, home assistance devices that respond to your voice and know your preferences are powered by deep learning applications.

Most deep learning methods use neural network architectures, deep learning models are often referred to as deep neural networks. The term “deep” usually refers to the number of hidden layers in the neural network. Traditional neural networks only contain 2-3 hidden layers, while deep networks can have as many as 150.

Deep learning models are trained by using large sets of labeled data and neural network architectures that learn features directly from the data without the need for manual feature extraction. One of the most popular types of deep neural networks is known as convolutional neural networks (CNN or ConvNet). A CNN convolves learned features with input data, and uses 2D convolutional layers, making this architecture well suited to processing 2D data, such as images.

CNNs eliminate the need for manual feature extraction, so you do not need to identify features used to classify images. The CNN works by extracting features directly from images. The relevant features are not pretrained; they are learned while the network trains on a collection of images. This automated feature extraction makes deep learning models highly accurate for computer vision tasks such as object classification.

1.2. DEEP LEARNING VS MACHINE LEARNING

The easiest takeaway for understanding the difference between machine learning and deep learning is to know that deep learning *is* machine learning. More specifically, deep learning is considered an evolution of machine learning. It uses a programmable neural network that enables machines to make accurate decisions without help from humans. Deep learning is just a subset of machine learning. In fact, deep learning technically is machine learning and functions in a similar way (hence why the terms are sometimes loosely interchanged).

While basic machine learning models do become progressively better at whatever their function is, they still need some guidance. If an AI algorithm returns an inaccurate prediction, then an engineer has to step in and make adjustments. With a deep learning model, an algorithm can determine on its own if a prediction is accurate or not through its own neural network. In the flashlight example: it could be programmed to turn on when it recognizes the audible cue of someone saying the word *"dark"*. As it continues learning, it might eventually turn on with any phrase containing that word. Now if the flashlight had a deep learning model, it could figure out that it should turn on with the cues "I can't see" or "the light switch won't work," perhaps in tandem with a light sensor. A deep learning model is able to learn through its own method of computing—a technique that makes it seem like it has its own brain.

Machine learning offers a variety of techniques and models you can choose based on your application, the size of data you're processing, and the type of problem you want to solve. A successful deep learning application requires a very large amount of data (thousands of images) to train the model, as well as GPUs, or graphics processing units, to rapidly process your data.

When choosing between machine learning and deep learning, consider whether you have a high-performance GPU and lots of labeled data. If you don't have either of those things, it may make more sense to use machine learning instead of deep learning. Deep learning is generally more complex, so you'll need at least a few thousand images to get reliable results. Having a high-performance GPU means the model will take less time to analyze all those images.

1.3. CONVOLUTION NEURAL NETWORK(CNN)

A convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analyzing visual imagery. They are also known as shift invariant. CNNs are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "fully-connectedness" of these networks makes them prone to overfitting data. Typical ways of regularization include adding some form of magnitude measurement of weights to the loss function. CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns. Therefore, on the scale of connectedness and complexity, CNNs are on the lower extreme. Convolutional layers convolve the input and pass its result to the next layer. This is similar to the response of a neuron in the visual cortex to a specific stimulus. Each convolutional neuron processes data only for its receptive field. Although fully connected feedforward neural networks can be used to learn features as well as classify data, it is not practical to apply this architecture to images.

A very high number of neurons would be necessary, even in a shallow (opposite of deep) architecture, due to the very large input sizes associated with images, where each pixel is a relevant variable. For instance, a fully connected layer for a (small) image of size 100 x 100 has 10,000 weights for *each* neuron in the second layer. The convolution operation brings a solution to this problem as it reduces the number of free parameters, allowing the network to be deeper with fewer parameters. For instance, regardless of image size, tiling regions of size 5 x 5, each with the same shared weights, requires only 25 learnable parameters. By using regularized weights over fewer parameters, the vanishing gradient and exploding gradient problems seen during backpropagation in traditional neural networks are avoided. Pre-trained models are usually shared in the form of the millions of parameters/weights the model achieved while being trained to a stable state. Pre-trained models are available for everyone to use through different means.

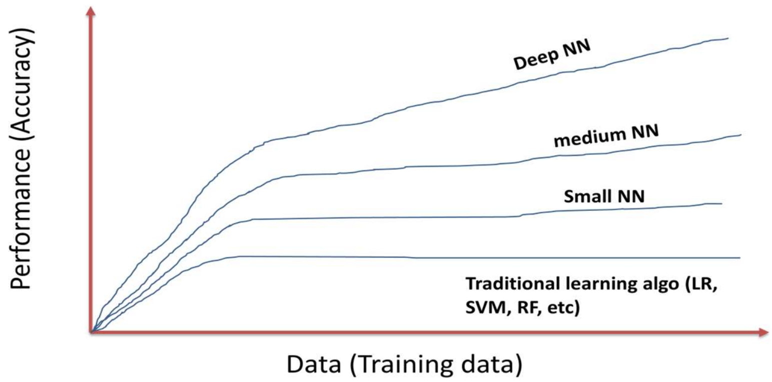

Figure 1.1: Performance of NN

The famous deep learning Python library, keras, provides an interface to download some popular models. You can also access pre- trained models from the web since most of them have been open-sourced.

Some popular pre-trained models: -

- VGG-16
- VGG-19
- Inception V3
- XCeption
- ResNet-50
- Convolutional neural networks require large datasets and a lot of computational time to train. Some networks could take up to 2-3 weeks across multiple GPUs to train.
- Transfer learning is a very useful technique that tries to address both problems. Instead of training the network from scratch, transfer learning utilizes a trained model on a different dataset, and adapts it to the problem that were trying to solve.
- Utilize the trained model as a fixed feature extractor: In this strategy, we remove the last fully connected layer from the trained model, we freeze the weights of the remaining layers, and we train a machine learning classifier on the output of the remaining layers.
- Fine-tune the trained model: In this strategy, we find tune the trained model on the new dataset by continuing the back propagation. We can either fine-tune the whole network or freeze some of its layers.

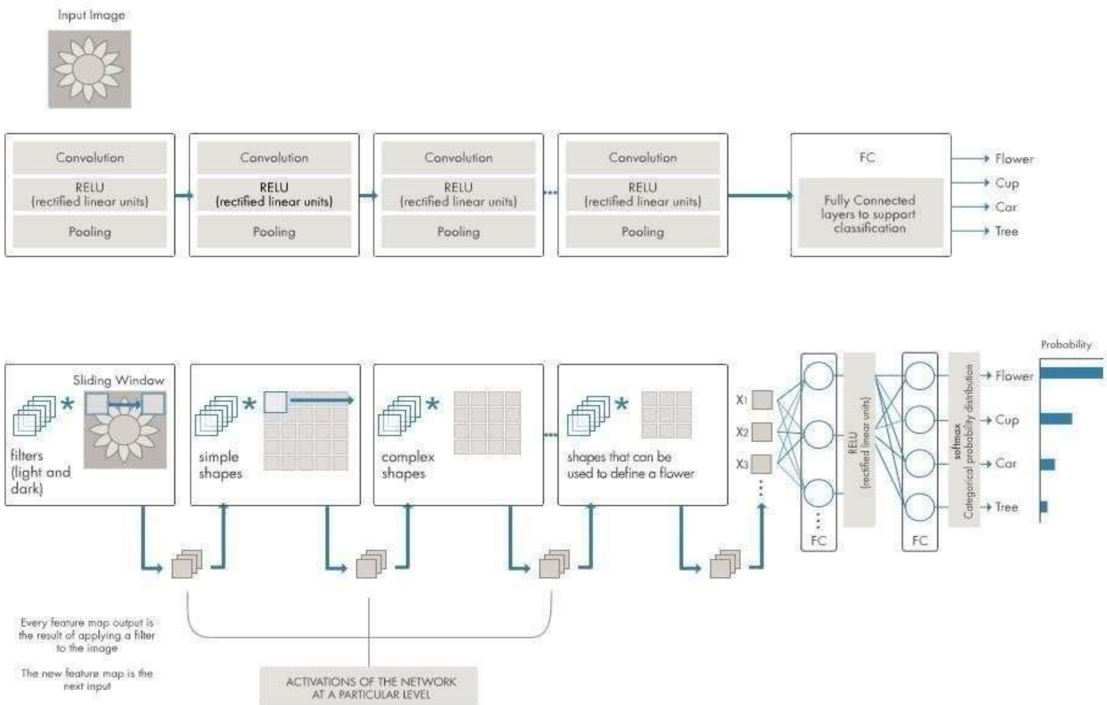

Figure 1.2: Layers of CNN.

Getting the best results in deep learning requires experimenting with different values for training parameters, an important step called hyper parameter tuning. Since Deep Learning Pipelines enables exposing deep learning training as a step in Spark's machine learning pipelines, users can rely on the hyper parameter tuning infrastructure already built into Spark. Pre-trained models are usually shared in the form of the millions of parameters, weights the model achieved while being trained to a stable state. Pre-trained models are available for everyone to use through different means. The famous deep learning Python library, keras, provides an interface to download some popular models. You can also access pre-trained models from the web since most of them have been open-sourced.

1.4. OBJECTIVES

- To overcome the computational expense and to provide more accurate bounding boxes, has introduced the YOLO (You Only Look Once) method.
- Identification of fruits is by using YOLO-V3 algorithm.
- We proposed a solution to help the people to determine the type of apple more accurately and to predict the type of images of apple.

- The quantity and the quality of the images have increased, it leads to both opportunities and challenges in this field.
- Images are high quality images taken from the top of the earth at varying heights, at varying light conditions for variety of applications. The accuracy of the object detection results can be improved by increasing the count of the images.

CHAPTER 2

METHODOLOGY

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos. Object detection has applications in many areas of computer vision, including image retrieval and video surveillance. To detect apples, we use YOLOv3 algorithm which contains 1000 images and we were guided by the round shape of objects, and the categorized into “apples”.

2.1. GOOGLE COLABORATORY

Google Colab is a free cloud service for machine learning education and research. It provides a runtime fully configured for deep learning and free-of-charge access to a robust GPU. Colab is the perfect place to do your research, tutorials or projects. Obviously, not everything can be wonderful. Colab has some limitations that can make some steps a little bit hard or tedious. In this tutorial I compiled some tips and tricks to mitigate these limitations. Notebooks are not very handy to program in. I keep as much work as possible to be done on your computer transparently and leave the notebook the training tasks.

2.2. YOLOV3

YOLOv3 is an object detection algorithm (based on neural nets) which can be used detect objects in live videos or static images, it is one of the fastest and accurate object detection method to date. YOLO uses a training set comprised of images and their corresponding bounding boxes (of target objects). YOLO runs at its best when utilizing an NVIDIA GPU and CUDA, if you run YOLO on CPU alone, you will experience significantly slower training times and debugging times. At the time of writing this post I used a macbook pro to train and test my model.

2.3. DARKNET

Darknet is the part of the Internet below the private deep web that uses custom software and hidden network superimposed on the architecture of the Internet. “Darknet” was often associated with the Tor network, when the infamous drug bazaar Silk Road once headlines. Darknet was coined in the 1970s to designate networks isolated by ARPANET.

Darknets in general may be used for various reasons, such as

- Computer_vision(cracking, file corruption, etc.)
- Protecting dissidents from political reprisal
- File sharing
- Sale of restricted goods on darknet markets.

2.3.1. TO RUN DARKNET

- Configure Runtime type to use GPU. (Only the first time)
- Get access to Google Drive and map as network drive
- Install cuDNN
- Clone and compile Darknet

2.3.2. TO TRAIN YOLO

- YOLO model configuration file (yolo-...-.cfg)
- An Image data set
- Data configuration file (obj.data)
- The pre-trained weights file (file.weights)

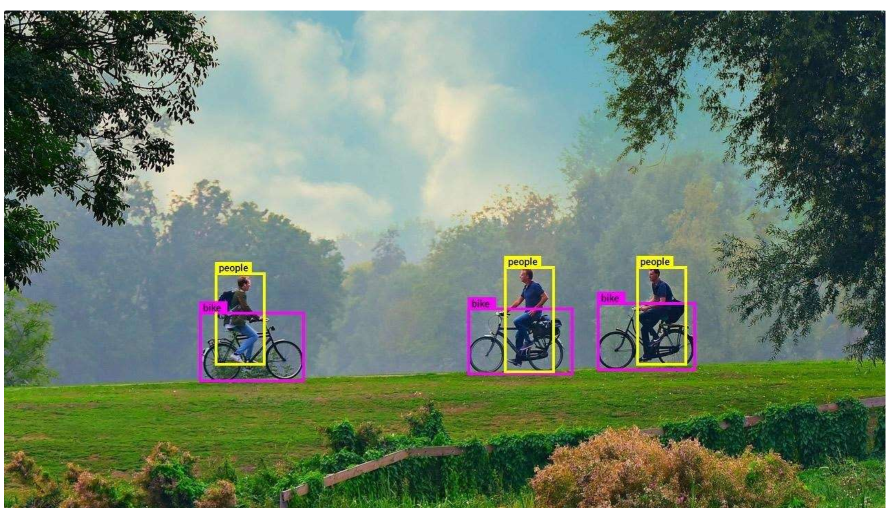

Figure 2.1: Training image

2.4. TRAINING A MODEL

- Input an image we know what contains.
- Process this through our neural network and get a result.
- Compare the result to what we know it really contains.

- Adjust the weights in order to minimize the difference between what we obtained and we know we had. This process is repeated until we get the best weights we can. Here are some terms involved in this process. To know that these terms are, is necessary for start training your own object detection model.
- **Weights** - The parameters we need to train in our neural network in order to, given an input, having a desired output.
- **Train** - Apply a defined number of images to our model to get and correct the best weights for our purpose.
- **Data Set** - A pairs of images and labels.
- **Labels** - The annotations we'll prepare for each image to indicate to the model what it has to be found in each image.
- **Train Data Set -** The data set of images we'll use to train our NN.
- **Test Data Set -** The data set we'll use to validate our NN.
- **Classes -** The number of objects we want to detect in our model.

2.4.1. THE STEPS ARE FOLLOWED ARE:

PART 1: Configuration setup

STEP 0. Configure runtime to work with GPU.

STEP 1. Connect your files to Google Drive.

STEP 2. Connect the Colab notebook to Google Drive.

STEP 3. Check CUDA release version.

STEP 4. Install cuDNN according to the current CUDA version.

STEP 5. Installing Darknet.

STEP 5-A. Cloning and compiling Darkent. ONLY NEEDS TO BE RUN ON THE FIRST EXECUTION!!

STEP 5-B. Copying the compiled version of Darknet from Drive. UNCOMMENT AFTER FIRST EXECUTION.

STEP 6. Runtime configuration finished!

PART 2. Training YOLO

STEP 0. Preparing your data and configuration files.

STEP 1. Loading files to VM local drive.

STEP 1 - Option 1. Using files from Google Drive directly.

STEP 1 - Option 2A. Copying files from Google Drive to VM local file system.

STEP 1 - Option 2B. Copying files zipped from Google Drive to VM local file system and unzip locally.

STEP 1 - Option 3. Clone your image dataset from a git repo. Seems the fastest one. Finally, Train your model!

CHAPTER 3

IMPLEMENTATION

3.1. DARKNET

We will use Darknet, an open source neural network framework to train the detector. Download and build.

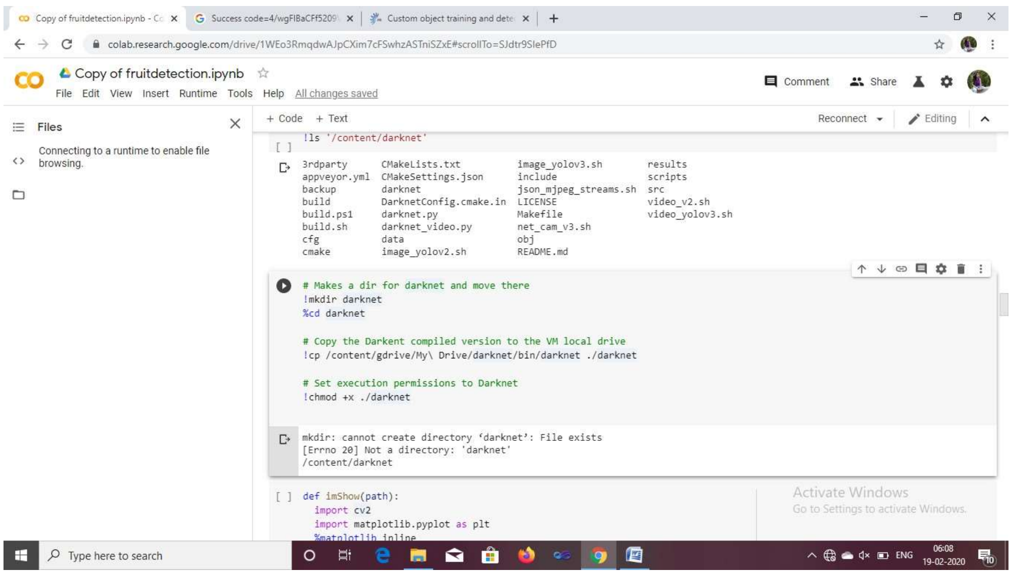

Figure 3.1: Darknet Configuration

3.2. ANNOTATION

After we collect the images containing our custom object, we will need to annotate them. For YOLOv3, each image should have a corresponding text file with the same file name as that of the image in the same directory.

In our case text files should be saved in data/train directory. For e.g. image1.jpg should have a text file image1.txt.Each row in the text file corresponds to a single bounding box of the object and should have the following information

classes=1

train=/content/gdrive/My\Drive/darknet/train.txt

valid=/content/gdrive/My\Drive/darknet/test.txt

names=/content/gdrive/My\Drive/darknet/obj.names

backup=/content/gdrive/My\ Drive/darknet/backup

This can also be done with LabelImg, a graphical image annotation tool which creates .txt files for the images in YOLO format as shown in Figure 3.3.

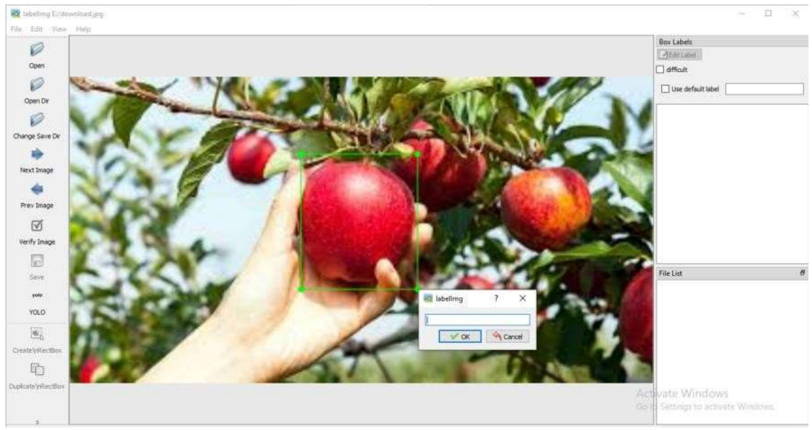

Figure 3.2: Labelling of image.

Figure 3.3: Annotation obtained.

3.3. TRAIN AND TEST SETS

We can then randomly split the annotated images into train and test sets in the ratio of 80:20 data/train.txt Each row in the file should have the location of train dataset. data/test.txt Each row in the file should have the location of test dataset.

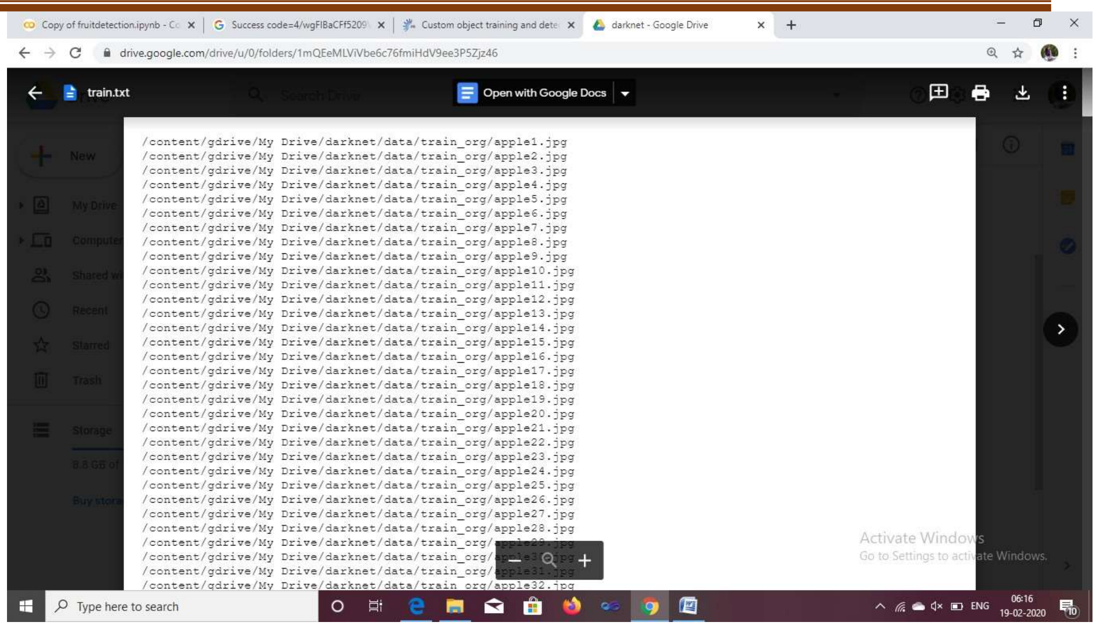

Figure 3.4: Weights of training dataset

3.3.1. PRE-TRAINED WEIGHTS

To train our object detector we can use the existing pre trained weights that are already trained on huge data sets. From here we can download the pre trained weights to the root directory.

3.3.2. YOLO DATA FILE

Create a detector data file in the data directory which should contain information regarding the train and test data sets backup is the location where newly trained weights saved.

3.3.3. CONFIGURATIONS

We set the DNN backend to OpenCV here and the target to CPU.

```
[net]
# Testing
#batch=1
#subdivisions=1
# Training
batch=64
subdivisions=16
width=416
height=416
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.001
burn_in=1000
max_batches = 4000
policy=steps
steps=3200,3600
scales=.1,.1

[convolutional]
batch_normalize=1
filters=32
size=3
stride=1
pad=1
activation=leaky
```

Figure 3.5: Configurations of dataset

3.4. TRAINING

With all the required files and annotated images we can start our training./darknet detector train custom_data/detector.data custom_data/cfg/yolov3- custom.cfg darknet53.conv.74

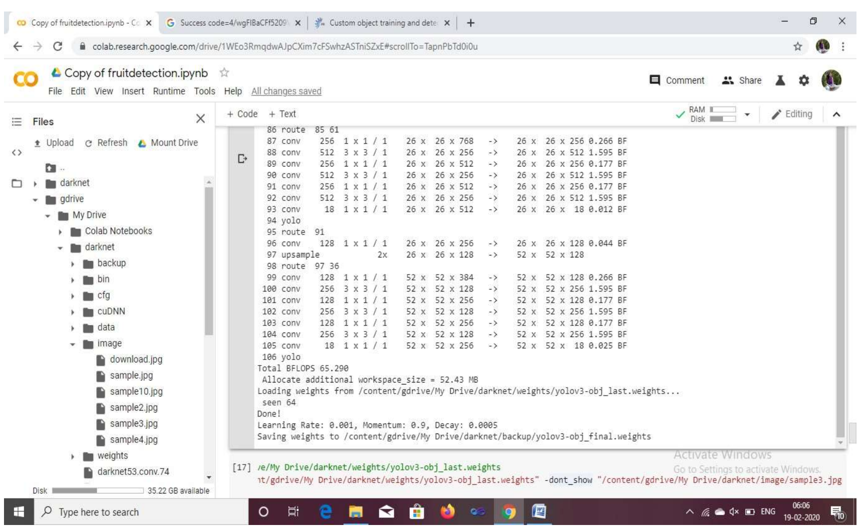

Figure 3.6: Training the dataset

Once the training is complete we can use the generated weights to perform detection.

CHAPTER 4

RESULTS

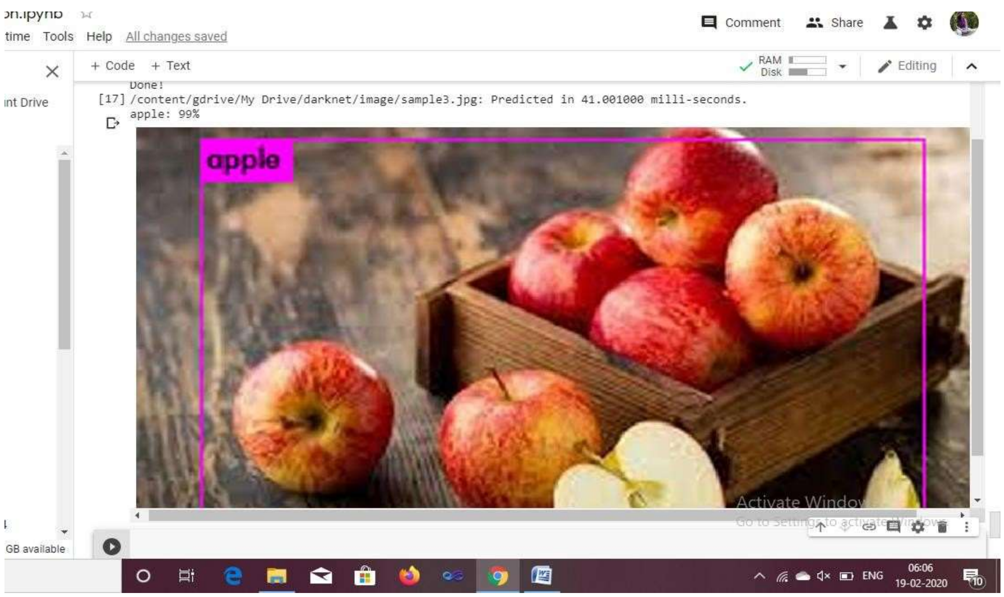

Figure 4.1 Apple detection(Final output).

Finally, our proposed model is capable of detecting fruits from image with a better accuracy and average precision rate.

CHAPTER 5

CONCLUSION AND FUTURE ENHANCEMENT

The present approaches are for a vision-based fruit detection system that can detect only a fruit with label. This can be implemented for multiple fruits also. From our point of view one of the main objectives for the future is to improve the accuracy of the neural network. This involves further experimenting with the structure of the network. In the near future we plan to create a mobile application which takes pictures of fruits and labels them accordingly.

REFERENCES

[1] H. Chen, Y. Wang, G. Wang and Y. Qiao, "LSTD: A Low-Shot Transfer Detector for Object Detection," arXiv:1803.01529v1, 2018.

[2] H. Xu, X. Lv, X. Wang, Z. Ren and R. Chellappa, "Deep Regionlets for Object Detection," arXiv:1712.02408v1, December 2017.

[3] "Deep learning to frame objects for visual target tracking," Engineering Applications of Artificial Intelligence, vol. 65, pp. 406- 420, October 2017.

[4] C.-C. Kao, P. Sen, T.-Y. Lee and M.-Y. Liu, "Localization-Aware Active Learning for Object Detection," arXiv:1801.05124v1, January 2018.

[5] J. P. N. Cruz, M. L. Dimaala, L. G. L. Francisco, E. J. S. Franco, A. A. Bandala and E. P. Dadios, "Object Recognition and Detection by Shape and Color Pattern Recognition Utilizing Artificial Neural Networks," in 2013 International Conference of Information and Communication Technology (ICoICT), Bandung, Indonesia, 2013.

[6] A. Gulli and S. Pal, Deep Learning with Keras, Birmingham: Packt, 2017.