



# Object Detection

INT 248 PROJECT BY :

MANISH SHARMA

11806127

---

RKM073B59

[GITHUB LINK](#)

# TABLE OF CONTENTS

---

1) Introduction

2) Problem Statement

3) Datasets

4) Implementation

5) Result

6) Screenshots

7) References

# Introduction

---

Object classification is a critical task in computer vision applications. It is the task of classifying objects from different object categories. It is useful for Duckie bot to classify the objects in the received images and it can be helpful in tasks such as object detection and tracking. In Duckie town, there are many informative objects such as traffic signs, Duckie bot, duckies, house models, etc. Convolutional Neural Networks (CNN) is one of the variants of neural networks used heavily in the field of Computer Vision

# Continue...

---

It derives its name from the type of hidden layers it consists of. The hidden layers of a CNN typically consist of convolutional layers, pooling layers, fully connected layers, and normalization layers. It is a great method to use deep learning. Deep Learning is becoming a very popular subset of machine learning due to its high level of performance across many types of data. The Keras library in Python makes it pretty simple to build a project.

# Problem Statement

---

In this project, the classification model of cars (5 classes) is built using tensorflow keras python.

# Datasets

---

Name	Object-images
Training set	4024 images
Testing set	423 image
Image size	1280 x 856
No. of Classes	5
classes -	tree, car, wheel, chair, Table

# Implementation

---

The model starts with downloading the dataset from the drive followed by importing related libraries which include TensorFlow Keras, NumPy, matplotlib.pyplot and some more.

**Build the Model** : The images that will go into convnet are 1280X856 color images on Data Preprocessing, we'll add handling to resize all the images to 1280X856 before feeding them into the neural network).

# Continue...

---

We will stack 3 {convolution + relu + maxpooling} modules. Our convolutions operate on 3x3 windows and our maxpooling layers operate on 2x2 windows. Our first convolution extracts 16 filters, the following one extracts 32 filters, and the last one extracts 64 filters. Before the output layer, two fully connected layers are added. The output layer is using softmax activation to classify images.



# Continue...

---

Creating accurate machine learning models capable of localizing and identifying multiple objects in a single image remains a core challenge in computer vision. The TensorFlow Object Detection API is an open source framework built on top of TensorFlow that makes it easy to construct, train and deploy object detection models. At Google we've certainly found this codebase to be useful for our computer vision needs, and we hope that you will as well.

# Standard HyperParameters

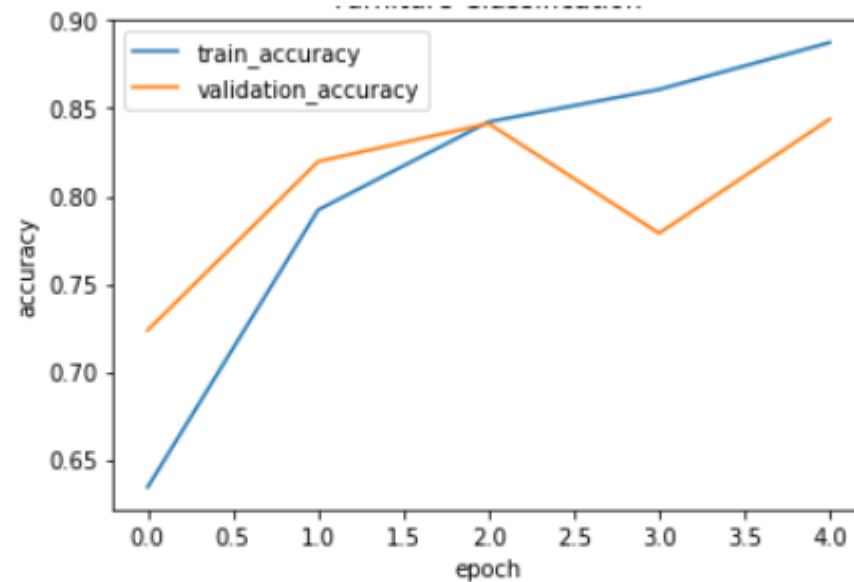
---

Conv Layers	3 Layers
Filter Size	3
Epochs	5
L2 Regularisation	0 (REMOVE)
Dropout	0 (REMOVE)
Batch size	16
Optimizer	adam
Loss Categorical	crossentropy

# Result

---

Standard results are the results removing all the additional hypermeters (consider standard hyperparameters)



**Graph between the Accuracies and Epochs**

# Analysis

---


By changing the hyperparameters of the model, we can see the changes in the results and analyse for the better model. However, changes may have different effects on different datasets and hence it must be chosen carefully. The accuracy tweaks with epochs are shown below as per the changes made. Other hyperparameters in each comparison are standard parameters of the model.

• • •


---

From the above results, we can analyse that adding or removing L2 regularization, changing the layer count or change in filter is not making much changes in the last epoch. Accuracies are in the range [0.83, 0.89]. Although adding dropout of 0.2 is increasing the train accuracy of the model up to 0.945 and validation accuracy up to 0.885 in last epoch.

# ScreenShots


 object-detection-using-tensorflow-hub.ipynb ☆

File Edit View Insert Runtime Tools Help [All changes saved](#)

Comment Share Settings 

+ Code + Text

Connect Editing



Saved successfully! ✕





# References

---

- Deep Learning Tutorial | [deeplearning.ai](https://deeplearning.ai)
- A friendly introduction to CNN
- CNN Tutorial | SimpliLearn
- Understanding Neural Networks | Towards Data Science



# Thank You!

---