

Improved Image Pre-Processing for Sharpened Object Detection in Image

Submitted in partial fulfillment of the requirements for the degree of

**Bachelor of Technology
In
Computer Science**

by

**Aditya Firoda
16BCE2184**

Under the guidance of

Dr. Jayakumar K.

(School of Computer Science and Engineering)

VIT, Vellore.



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

May, 2020

DECLARATION

I hereby declare that the thesis entitled “**Improved Image Pre-Processing for Sharpened Object Detection in Image**” submitted by me, for the award of the degree of Bachelor of Technology in Computer Science to VIT is a record of bonafide work carried out by me under the supervision of **Dr. Jayakumar K.**

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore

Signature of the Candidate

Date:

CERTIFICATE

This is to certify that the thesis entitled “Improved Image Pre-Processing for Sharpened Object Detection in Image” submitted by **Aditya Firoda (16BCE2184), School of Computer Science and Engineering, VIT University**, for the award of the degree of Bachelor of Technology in Computer Science, is a record of bonafide work carried out by him under my supervision during the period, 01. 12. 2019 to 27.05.2020, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place: Vellore

Signature of the Guide

Date:

Internal Examiner

External Examiner

Dr. Santhi V
(School of Computer Science and Engineering)

ACKNOWLEDGEMENTS

A project gives us a golden opportunity to have a greater understanding of the theoretical knowledge and helps to explore by applying what we have inherited. On this occasion, we would like to express our heartfelt gratitude to our guide, **Prof. Jayakumar K** for his guidance and encouragement throughout this project.

Additionally, we are filled with gratitude for the opportunity provided by our School Dean, **Prof. R Saravanan** and Head of Department, **Dr. V Santhi** and the Vellore Institute of Technology Management to carry out this project. It has escalated our practical knowledge and clarifies the process of making an engineer.

We would like to convey our gratitude to the chancellor of Vellore Institute of technology, **Dr. G. Viswanathan** for giving us the opportunity to pursue our Engineering

from this prestigious university.

It is our deepest sentiment to place on record our best regards, a most profound feeling of appreciation to different faculties for their mindful and valuable guidance which were extremely precious for our project both theoretically and practically.



Aditya Firoda (16BCE2184)

Executive Summary

This project primarily focuses on ways to get better object detection using deep neural networks, which will help in identifying the tanks and other armoured vehicles across different weather conditions.

This project has 4 components - Firstly a parser that was created to get all the color values from a website which will help in recognizing and recreating an image with a specific color palette.

The project further delves into the 2nd component that is the fog removal system which is based upon the technique of dark channel prior, which is able to detect fog in different regions of the image and therefore able to get better haze/fog-free images overall.

Next component of the project revolves around dominant color extraction techniques which improve upon the basic techniques such as K-means which are compute intensive in nature and do not deliver consistent results upon multiple runs. This technique of hierarchical quantization helps solve the problems of K-means.

This leads to and helps the system which has a deep neural network model which is able to detect tanks with high accuracy and precision and is able to get improved detection thanks to the help of fog removed images as part of pre-processing. The model is even able to detect more tanks from deteriorated images as it has been trained by creating images from only dominant colors of the image.

TABLE OF CONTENTS

TITLE	Page No.
Acknowledgement	4
Executive Summary	5
Table of Contents	6
List of Figures	8
List of Tables	10
Symbols and Notations	12
1. Introduction	
1.1 Theoretical Background	13
1.1 Motivation	13
1.2 Aim	13
1.3 Objective	14
1.5 Literature Survey	14
2. Project Description and Goals	
2.1 Fog Removal	19
2.2 Dominant Color Extraction	22
2.3 Residual Net Architecture	25
2.4 Google Open Image Dataset V4	27
2.5 OIdv4_Toolkit	29

3.	Technical Specification	
3.1	Functional Requirements	30
3.2	Non-Functional Requirements	30
3.3	System Requirements	31
4.	Design Approach and Details	
4.1	Design Approach/Materials and Methods	33
4.2	Constraint, Alternatives and Trade-Off	34
5.	Schedule Task and Milestones	36
6.	Project Demonstration	
6.1	Parsing the Asian Paints Library for entire color palette	37
6.2	Fog Removal from the Image using Dark Channel Prior	39
6.3	Dominant Color Extraction	44
6.4	Object Detection Model Resnet	46
7.	Result and Discussion	53
8.	Summary	54
9.	Appendix A	55

List of Figures

Fig No.	Title	Page No.
2.1	Overview of Proposed System	17
2.2	Diagram for formation of fogged photo	18
2.3	Equation for the hazy Image	19
2.4	Transformed equation	19
2.5	Dark channel of Fog Free Images	20
2.6	Dark Channel of Image with Haze	21
2.7	Splitting the image using a binary tree representation	22
2.8	Eigen Vector shown in 2D	23
2.9	Mathematical Formula for splitting the nodes	24
2.10	3D Plane separating the nodes into 2 separate nodes	24
2.11	Skip Connections	25
2.12	Index page of Google Image Dataset	27
2.13	Count of images in the v4 of dataset	27
2.14	Subset of the tank dataset	28
4.1	Flow for Prototype Model	32
6.1	Color Palette Showing all the Colors in the Database	36
6.2	Color Details of the Selected Color	37
6.3	Terminal Screen for Starting dark channel program	38
6.4	Input Foggy Image test.png	39

6.5	Formation of Dark Channel for the input Image	40
6.6	Transmission Map of the Image	41
24	Output Enhanced Image	42
6.7	RGB values which are the dominant colors in an image.	43
26	Output Image using only the dominant colors	44
6.8	Cloning the OID Toolkit	45
6.9	Installing dependencies	45
6.10	Downloading Tank Dataset	46
6.11	Tank Dataset saved in Google Drive	47
6.12	XML annotation file	48
6.13	Annotation.csv file showing the bounding boxes with the image location	48
6.14	Resnet Model Being Trained	49
6.15	Model Being Trained and saved after each epoch	50
7.1	CSV file created by scraping the website	51
7.2	Performance of models with 56 and 20 layers respectively on both the training and testing dataset.	51
7.3	Model Showing Inference in the picture	52

List of Tables

Table No.	Title	Page No.
1	Literature Survey	14
2	Milestone Table	36
3	Timeline Table	38

Symbols and Notations

I - Fogged Image Captured by the Camera

J - Fog Free Image

A - Atmospheric Light

d - Scene Depth

T - Transmission Map

β - Scattering Coefficient

D - depth of scene

$$t(x) = e^{-\beta d(x)}$$

$W^{[l]}$ - Weight of Layer l

$B^{[l]}$ - Bias of Layer l

$A^{[l]}$ - Output of Layer l

Relu(input) - Relu Function application on input

I5 - Intel i5 Processors

1. INTRODUCTION

1.1Objective - With the launch of multiple satellites by ISRO into space, India as a nation has completed the step of data collection as now it has a huge amount of data on the earth and this volume now presents a problem. The data now needs to be processed for extraction of information, this can't be done in one step as a lot of data has different types of noise and distortion which makes the feature prediction difficult and finally process intensive if humans are used for extraction of information.

1.2Motivation - While there is a lot of topographical data available for the DRDO by the Indian satellites but most are not usable due to the image quality and various weather and geographic conditions across India. These problems such as fog during winters or image distortion due to shadows in the image which decrease the efficiency of the computer vision algorithms can be reduced by employing certain image Pre-Processing pipelines which can help in improving the features of the image.

1.3Aim of the Proposed Work - To create a model where there is Pre-Processing of the image done to improve features, these Pre-Processing are removal of fog, shadows and finding the dominant color to create image from basic color palette and further after the Preprocessing using the image to train the best convolutional neural network which detects the different vehicles across the different images with high level of confidence. While creating a model from scratch is not required only that find the best existing model which has least losses and trains faster on images.

1.4Objective(s) of the Proposed Work -

- ◆ Implement the dominant color extraction techniques and find the closest color from a pre-defined library
- ◆ Create a shader library from the Asian paints library for use
- ◆ Improved algorithm for fog removal from photos based on dark channel prior.
- ◆ Find an algorithm and implement the same for enhanced shadow detection for the seamless blending of the images.
- ◆ Object/Feature detection using machine learning for enhanced classification.

1.5 Literature Survey

S. No.	Paper Name/ Model Name	Author/ Name of Organization	Year	Abstract	Adv/Disadvantages.
1.	Color Segmentation Using an Eigen Color Representation	Alaa E. Abdel-Hakim and Aly A. Farag Computer Vision and Image Processing Laboratory (CVIP Lab.) University of Louisville, Louisville/40292, KY USA	August 2005	Color segmentation is a difficult problem to solve due to the high degree of changes due to different weather and light conditions in which the image is shot. Also the existing color segmentation approaches have their results affected due to the type of color models they use, which means the results will change if we change color models. This paper suggests a linear transformation to decrease the dimensionality of feature space. It also proves the experimental proposition by	Adv - It saves from multiple iterations as compared to the k means method. It evolves from the limitations of existing color models Dis - Implementation of the proposed algorithm is difficult and only available for RGB color model

				comparison between this model and all the classical models available using real images.	
2.	Object Recognition in Images using Convolutional Neural Network	Mr.Sudharshan Duth P and Ms.Swathi Raj Dept. of Computer Science Amrita School of Arts and Sciences Mysuru, Karanataka, India	January 2018	Recognition of objects in images has been a hard problem to solve in field of computer science, this paper demonstrates the solution using cifar-10 dataset of images which is a standard dataset. They have created a convolutional neural network with the help of keras which used tensorflow in the background. This network was trained for multiple epochs on the training set of cifar-10 which consists of 60000 images and the results of the above CNN were demonstrated in the paper which has a very high percentage of accuracy.	Adv - The training part ensures minimum human interaction which helps in better results Dis - The CNN is prone to high error rate and must be run for multiple epochs for better results which is computationally expensive and at same time still has some error rate.
3.	Deep Residual Learning for Image Recognition	Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun Microsoft Research	December 2015	As we increase the number of layers the deep neural networks become difficult to train. This paper proposes an approach to ease the training. Till this point after several layers the accuracy of neural networks started decreasing, because of vanishing gradient and losing	Adv - It improves the accuracy of the model and eases the problem of vanishing and losing gradients suffered by CNN due to increase in depth of the deep neural networks.

				<p>gradient problems. This paper proposes the novel idea of residual networks which are easier to train and upon increase in layers they generate better results. The authors of this paper won almost every image recognition contest the following year.</p>	<p>Dis - This model is extremely compute intensive in nature which is bad for scaling.</p>
4.	Single image dehazing with a physical model and dark channel prior	<p>Jin-Bao Wang, Ning He, Lu-Lu Zhang, Ke Lu Beijing Key Laboratory of Information Service Engineering, College of Information Technology, Beijing Union University, Beijing 100101, China b Chinese Academy of Sciences, Beijing 100049, China</p>	February 2015	<p>The following paper has proposed a method which is based on the existing dark channel technique. The dark channel technique takes an atmospheric light as a basis, and in this paper the authors insist that the result will be dependent on the choice of the atmospheric light, they also add the improvement of fast transmission estimation algorithm which further reduces the processing time for getting the output image.</p>	<p>Adv - The dark channel prior technique works well for object detection across heavily fogged images by showing features.</p> <p>Dis - This algorithm is computationally intensive so it is not adaptable in real-time scenarios.</p>
5.	Fog Removal from Color Images using Contrast Limited Adaptive Histogram Equalization	<p>Zhiyuan Xu, Xiaoming Liu, Na Ji Information Science and Technology College Dalian Maritime University Dalian, China</p>	October 2009	<p>Most of the images that have fog are extremely bad in contrast, to curb this issue, a novel approach of Contrast Limited Adaptive Histogram Equalization is used here. The color model used for this processing is HSI based, instead of RGB. And then</p>	<p>Adv - The contrast limited adaptive histogram equalization technique works extremely well in case of real time video which requires fast processing.</p>

				following the conversion the equalization is applied to the intensity component of the image. The authors also discuss and show how the method is more useful compared to the traditional models.	Dis - The algorithm does not work as effectively as the algorithm that takes a single image into account with fog thickness as an input.
--	--	--	--	---	--

2. Project Description and Goals

Over the course of last 15 years, the research of deep neural networks has made big leaps and which in turn has made possible for lot of newer approaches and research avenues in the field of computer vision. The improvement currently is that deep neural networks are being used to identify and classify objects across various scenes without much human help.

This project has made a certain novel approach to detection of objects in a manner that is able to get better results than any other deep neural network model and has a unique solution to a difficult and important problem.

The whole project consists of following components -

- Parsing the Asian Paints Library for the entire color palette offered by them
- Pre-processing the image for removal of fog
- Find the dominant colors from the image using hierarchical quantization
- Creating a Object Detection model to get inference of objects in the image to classify them as armoured vehicles

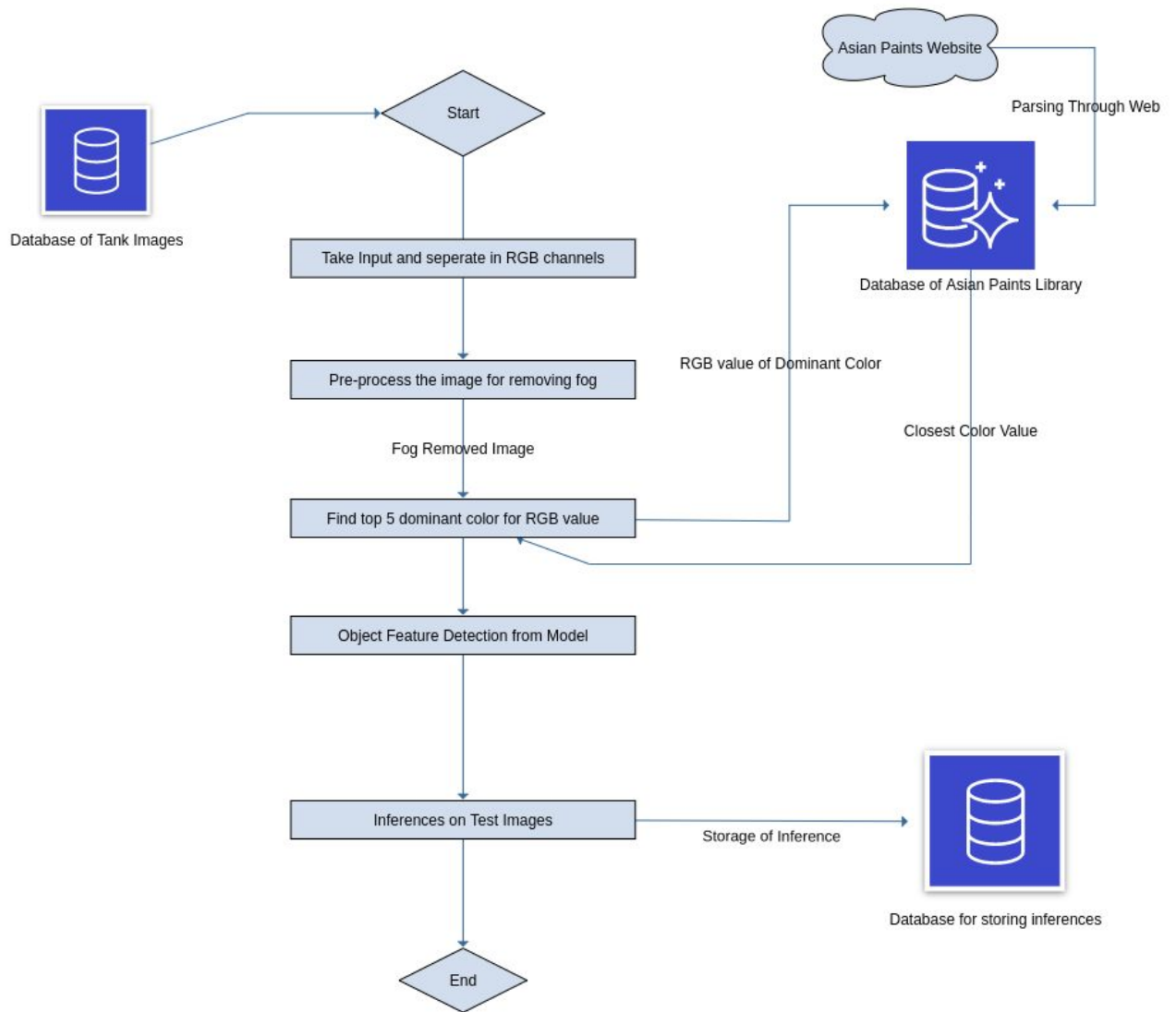


Fig 2.1- Overview of Proposed System

Project Goal -

This project helps to solve the problem of increased amount of unlabeled data which comes through the detection systems used by the army of India. This data needs to be analyzed quickly and efficiently to alert the authorities. This data also needs to detect anomalies.

Therefore the main goal of a system is to create a system which is able to make a reliable prediction across a variety of weather conditions to give precise and fast results.

Other project goals are as following -

- ❖ **Time Reduction** - Reducing the time taken to detect subjects across images, as a human takes much more time per image and also cannot process multiple images concurrently.
- ❖ **Reliability** - System should be able to detect the subjects at high accuracy and precision as it might form the part of defense measures deployed by the country.

- ❖ **Scalability** - The system should be scalable in nature that it is able to identify different types of subjects should the requirements of the system change and can work much better if provided with parallel systems.
- ❖ **Cost Reduction** - The system should be cost efficient in a manner that it should use the resources provided to the system as judiciously as possible.
- ❖ **More efficient and accurate** - The system should provide results with high accuracy and precision compared to the existing models.

2.1 Introduction and Related Concepts -

2.1.1 Fog Removal -

Removal of Fog Based on Dark Channel Prior -

This is a new method that is based on the acute observation of images in natural settings and based on the observation the de-hazing of the image is done which leads to much better, clearer images.

It arrives from the point that most pictures which capture outdoor settings have intensity value of at least one of RGB channels to be around zero, within the boundaries of the local window being considered.

The process of De-Hazing has following steps -

- 1) Atmospheric Light Estimation
- 2) Transmission Map Estimation
- 3) Transmission Map Refinement
- 4) Image Reconstruction.

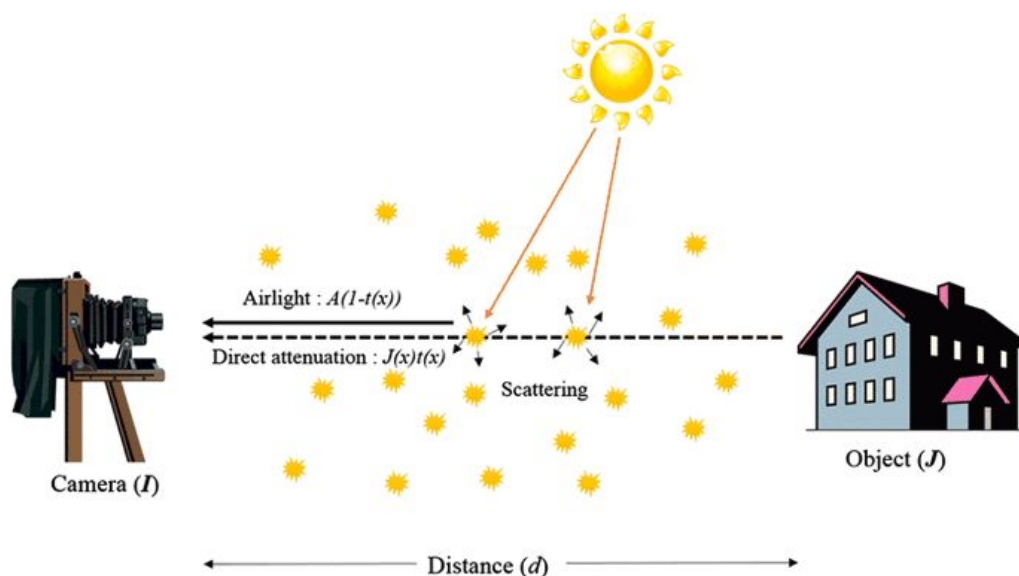


Fig2.2 - Diagram for formation of fogged photo

$$I(x) = J(x)e^{-\beta d(x)} + A (1 - e^{-\beta d(x)})$$

Fig2.3 - Equation for the hazy Image

This equation is used to get the Fog free image, if we change the equation we get

$$J(x) = \frac{I(x) - A}{t(x)} + A.$$

The effectiveness of this method was shown by K He et al. using 5000 photos taken in natural light, where around 75 percent of the images taken have some component of RGB channel around zero.

This is attributed due to the following reasons -

- Shadows, The shadows cast due to different objects in the natural world such as hills, trees etc.
- Extremely bright colors used by the different natural objects such as red rose, Dark colored animals etc.
- There are multiple dark objects and faces, such as cars, animals, rocky mountains etc.

Which confirms the observation that in all the images

$$J^{\text{dark}} \rightarrow 0$$

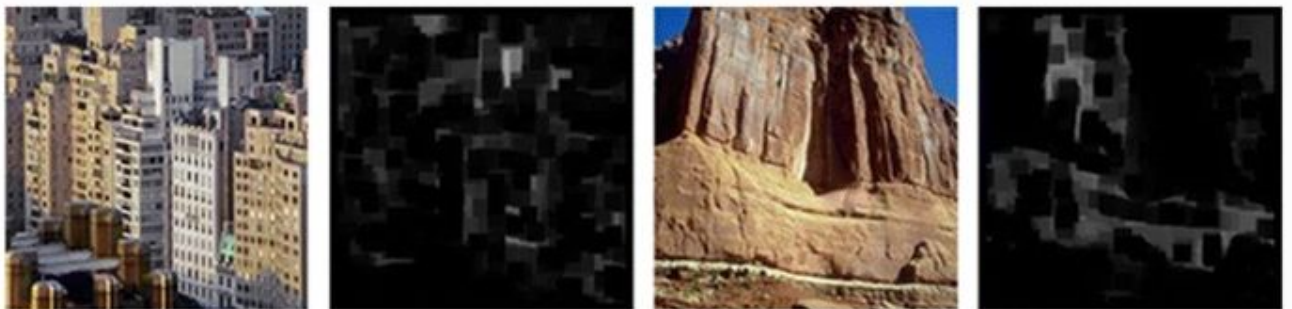


Fig2.5 - Dark Channel of Fog Free Images

The above figure shows the same where all the fog free images have dark channel prior has values tending to zero.

Also when this is compared with images that do contain the fog it causes the dark channel to give values which will have values far above zero. And this in turn helps us in discovering the density of haze at each point in the image.



Fig 2.6 - Dark channel of Foggy Images

As you can see in the above image the dark channel of the foggy image causes a lot of pixels to be not that dark.

Now from the dark channel formed we now take the brightest pixels of the dark channel, and use their color for the purpose of atmospheric light. Which helps in getting better results for fog removal.

2.2 Dominant Color Extraction -

Hierarchical Quantization - The idea of hierarchical quantization is that every image contains a millions of colors but the displays on which they are projected only have 256 shades of RGB, so if we dial down the number of colors to 4-5 colors we will be able to get the dominant colors of the image.

Process -

First we take an image “I” where each pixel in the image has an RGB value corresponding to that pixel. Now initially, we start by considering the entire image of one type or same color. We then keep dividing the image to get the number of dominant colors required. (If we want 5 colors we divide the image 5 times)

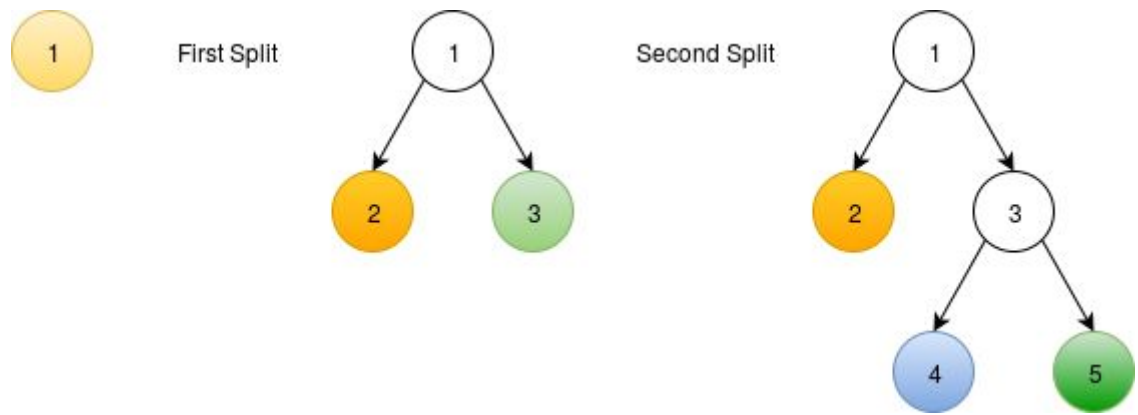


Fig2.7 - Splitting the image using a binary tree representation

The division of image is done by assigning each node in the tree with a unique class id and later after the split the new class id is given to the new nodes. Starting we just have a single node with the class id of 1.

The Splitting Criteria -

We split the matrix with the help of a covariance matrix. The matrix stores the information of how colors vary across the entire image. For e.g. A colorful image would have crazy numbers in the covariance matrix due to the huge variety of colors. As we divide down the image of colors the covariance matrix starts to have lower variances.

The covariance matrix helps us in creating a plane, which is the splitting criteria for the image.

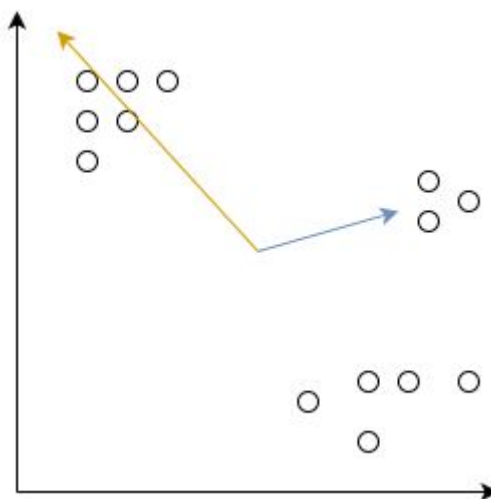


Fig2.8 - Eigen Vector shown in 2D

We now take the help of eigenvectors, these help us in identifying the plane where the covariance is maximum. In the figure 2 shown above we see that there are 2 vectors shown; the arrowhead shows the direction of the eigenvector and length of the arrow shows the eigen vector magnitude. But for our purposes we will be using a unit magnitude vector for direction and storing the magnitude separately as eigenvalues.

This approach is viable because the axis which shows the maximum change in co-variance will probably have two or more colors.

Splitting Process -

We split the nodes based on this mathematical formula -

$$C_{n+1} = \{s \in C_n : \mathbf{e}_n \mathbf{x}_s^t \leq \mathbf{e}_n \mathbf{q}_n^t\}$$

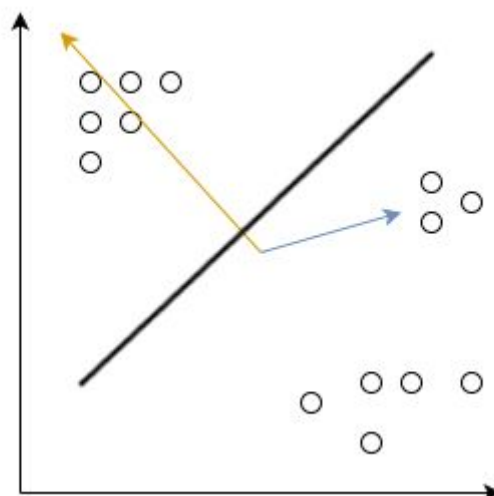
$$C_{n+2} = \{s \in C_n : \mathbf{e}_n \mathbf{x}_s^t > \mathbf{e}_n \mathbf{q}_n^t\}$$

Fig2.9 - Mathematical Formula for splitting the nodes

\mathbf{e}_n represents the normal of 3D-plane. Then we define the thick line in figure 4 as $\mathbf{e}_n \mathbf{q}_n^t$. The \mathbf{x}_s represents the actual value of the pixel at the location s .

Here we split the node N into $N+1$ and $N+2$ with the following conditions as stated in figure 3.

Fig2.10 - 3D Plane separating the nodes into 2 separate nodes



2.3 Residual Net Architecture -

In the beginning when the research on deep neural networks was being conducted, there was a belief that as we increase the depth of the networks we will be able to gain more accuracy as there are more parameters to change and hence the model will get better at prediction. But as models such as VGG16 and others were stacked with more layers the model in actual reality started performing poorly as compared to models with less layers.

This was later found out that with the increase in number of layers other problems start creeping in, which causes the model to perform so badly that it does not work well even with the training data set.

To overcome this problem Kaiming He et al. Came with the concept of skip connections

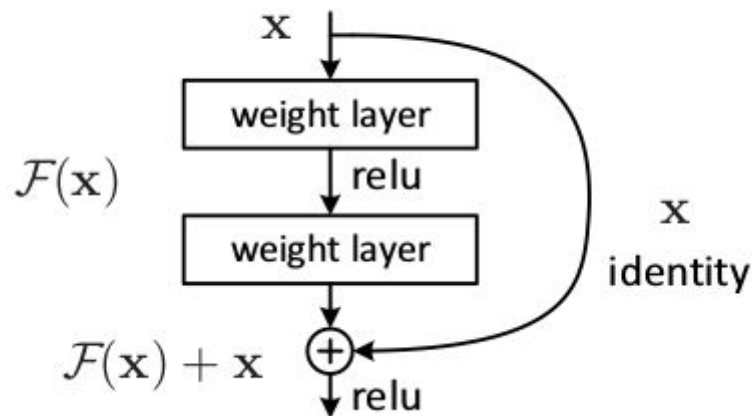


Fig2.11 - Skip Connections

$$A^{[l]} = x \text{ (Output of layer l)}$$

$$Z^{[l+1]} = W^{[l+1]} \cdot A^{[l]} + B^{[l+1]}$$

$$A^{[l+1]} = \text{relu}(Z^{[l+1]})$$

$$Z^{[l+2]} = W^{[l+2]} \cdot A^{[l+1]} + B^{[l+2]}$$

$$A^{[l+2]} = \text{relu}(W^{[l+2]} \cdot A^{[l+1]} + B^{[l+2]} + A^{[l]})$$

In basic terms we add the first layer pooling output to the third layer linear output and then take relu of that and pass that as input to the next layer.

This system works for all depth of layers such as 34 layers and even 152 layers

Skip Connections -

1) Helps in back propagation due to these connections also there is the compounded effect of regularization, helps in less overfitting, if $A^{[l+2]} = \{\text{first term is zero} + A^{[l]}\}$ then it helps training and test error reduces very much.

This architecture due to the way it is made helps us that efficiency of a deeper network will always be at least equal to the network with less layers, and can even perform better.

But overall this helps us in cases where the model started performing even worse than less layered models.

2.4 Google Open Image Dataset V4 -

This is an open source database project created by google for use by everyone trying to find relevant dataset to train their models. This dataset currently contains in its latest version around 9 million images. These images also have their annotation with them that means they can be use directly to train the model without having the user to create labels for each image using different tools such as hyperlabel etc.

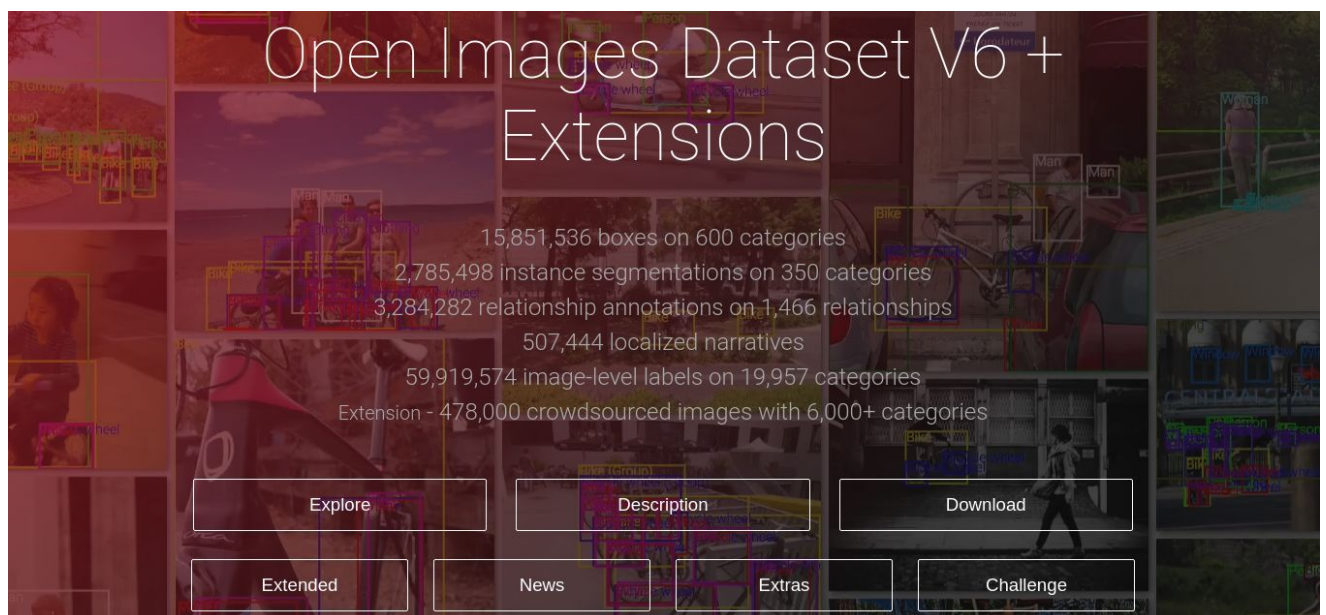


Fig2.12 - Index page of Google Image Dataset

This is the repository link - “<https://storage.googleapis.com/openimages/web/index.html>”

This dataset was used for development of the Resnet Model with the images of the tank, this model consists of 1000s of images of tank with around 1200 images for test dataset.

Object Detection

	Train	Validation	Test	#Classes
Images	1,743,042	41,620	125,436	-
Boxes	14,610,229	204,621	625,282	600

Image Classification

	Train	Validation	Test	#Classes
Images	9,011,219	41,620	125,436	-
Machine-Generated Labels	78,977,695	512,093	1,545,835	7,870
Human-Verified Labels	27,894,289	551,390	1,667,399	19,794

Fig2.13 - Count of images in the v4 of dataset

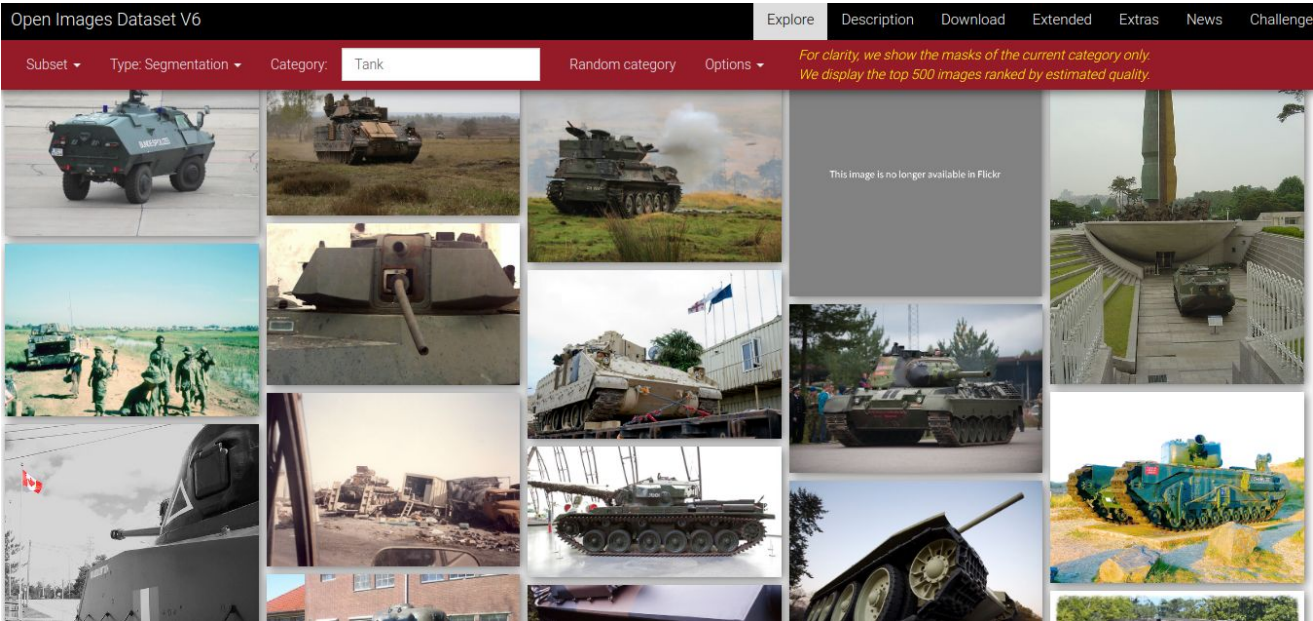


Fig2.14 - Subset of the tank dataset

2.5 OIDv4_Toolkit

While there are millions of images and around 19000 classes on the dataset if somebody wants only a subset of the all images for their use there is no proper way to download the subset other than downloading the entire annotations and then comparing the classes of each image to get the desired results.

This toolkit helps in getting the subset classes on which we want to train our model, this toolkit downloads the entire images dataset with the specific classes as specified and stores them with images and their labels in separate folders. The annotations are in the form of .txt file format which can be changed accordingly to the needs of the model that the user is training.

The toolkit can also be used to see the images pertaining to the specific class or in general otherwise.

3. Technical Specification

3.1 Functional Requirements -

3.1.1 **Product Perspective** - The product offers a way of better detection of features such as armed vehicles during bad weather conditions. This helps in detecting threats without any human intervention and also helps in reducing the workload and costs involved in assigning a human

3.1.2 **Product Features** - The Product requires the system to be computationally inexpensive and at the same time be able to get all the required objects and features from the image.

3.1.3 **Assumptions and Dependencies** - For the proper working of the system, we can list our assumptions and dependencies as follows.

- 1)Python installed on the computer
 - 2)TensorFlow installed on the system
 - 3)Keras installed on the system
- Availability of the images in the required format.

3.2 Non-Functional Requirements -

3.2.1 **Efficiency** -The model created should be able to infer the results within 4-5 seconds on a machine with the required capability. The model should be able to perform even with images of low resolution and heavily dense fogged images. Efficiency of the model should not drop below 70% percent detection rate of objects (i.e.) off all the images being tested it should be able to detect the correct label in 70% of the images

3.2.2 **Reliability** - The model must be reliable as to give good confidence scores of the objects detected by the model and not be confused in images with low resolution. The model should be secure enough that if employed in an application it should not be vulnerable to attacks which try to access the images for testing and training purposes.

3.2.3 Portability - The model should be portable given the case once the model is trained and tested it should be able to perform the same on different sets of machine and should not require an intense setup or retraining of the model.

3.2.4 Usability - Application should be usable and cater to the needs of the personnel and be controlled by a commands if possible should contain a graphical user interface which helps them drag and drop the image for inference from the model.

3.3 System Requirements

3.3.1 Hardware Requirements -

For Online -

- 1) A System with good internet connection

For Offline - (Min Requirements)

- 1) A System with i5 or greater processor
- 2) 10GB of RAM
- 3) 100GB of storage for data set
- 4) Operating System - Windows / Mac / Linux

3.3.2 Software Requirements -

For Online -

- 1) Internet Browser
- 2) Permissions for caching of different library
- 3) Google Co-lab Account
- 4) Google Drive Account with Ample Storage for data set

For Offline

- 1) Python - This is a high-level language which was created by Guido Van Rossum, it is extremely popular for the ease of usage and

comfortability in reading the code due to easy to understand syntax. It has automatic garbage collection procedures and is dynamically typed which means the variables do not require their data-type and size known in advance. It also has a variety of libraries which help in fast production of applications.

2)A Text Editor (PyCharm) - This is an integrated development environment which is used to develop large scale projects, it particularly works really good for python based projects.

3)Keras - This is a neural network library, which was initially released in Fall 2015 and has been the most popular library for neural network development. This works as a middle layer between tensorflow and user, and due to its user-friendlies it is preferred compared to directly working on tensorflow.

4)TensorFlow - This is an open-source library primarily used for neural networks and machine learning. It is also used for dataflows and differentiable programming.

5)Numpy, Matplotlib - These are famous python library which are used in a whole variety of applications, numpy has support for multidimensional and very large size arrays manipulation. It also has multiple functions that makes the job of programmer much easier and he/she is able to achieve the goal in fewer lines of code. Matplotlib serves as an extra hand to the numpy library by adding support for GUI based interface which help in easy navigation and much better visualizations of the data.

4.DESIGN APPROACHES AND DETAILS

4.1 Design Approach -

The project follows the agile model of prototyping, where there is quick prototype build for the purpose of demonstration, to the client. If the design is approved then the final implementation was done.

This model ensured that there was no difference between user requirements and the actual product being developed.

Advantages -

- 1) This helps in avoiding errors which are generated due to bad requirements analysis.
- 2) It also helps in increased user feedback which helps in getting right specifications.
- 3) More coherence, and less costs if any changes in requirements at a later stage.

Prototyping

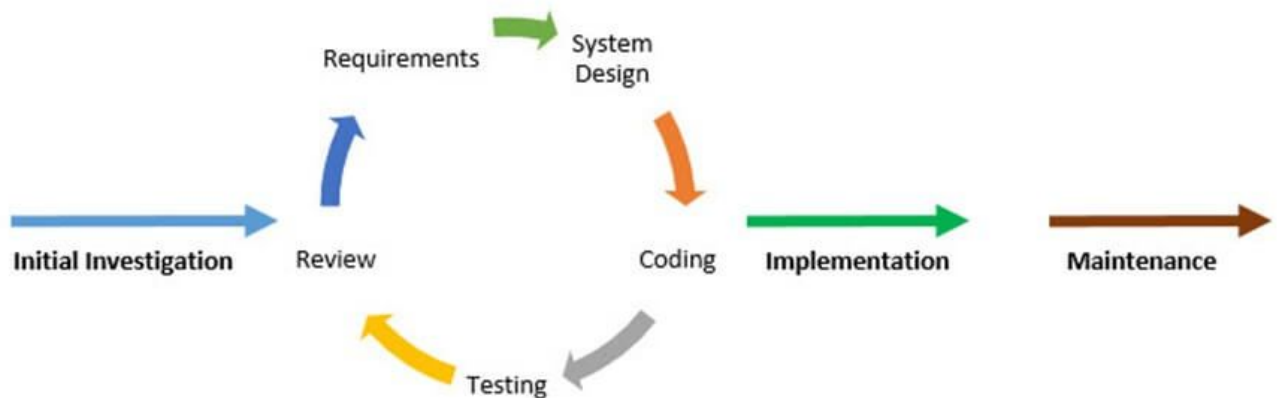


Fig4.1 - Flow for Prototype Model

Disadvantages -

- 1) This also paves path for creating a broken system then fixing the system as time progresses.
- 2) Requires a lot of interaction between the development team and customers.

3.2.1 Understanding the Problem -

As a developer the problem lies in the fact a lot of data is being generated and most of it has to be marked in by a human which lead to

lot more time consumption and a lot of human resources. The images due to the condition in which they are being shot, have a lot of noise which makes the process of identification harder.

3.2.2 Defining Requirements -

The system needs an automatic way of labeling the images that are being generated. The classified images should have a high degree of confidence. Also the system should preprocess the image to remove fog from the image which leads to the better detection and labeling.

3.2.3 Developing Prototype -

Based on the requirements we propose the following model -

3.2.4 Testing Prototype -

Once the prototype is initially made it is checked against a test data set and the accuracy of the model is computed if the model is able to achieve the accuracy percentage set by user then, the model is accepted, otherwise further changes are made

3.3 Constraints, Alternatives and Trade-off -

Constraints Include -

- > The system cannot work without the weights file which contains the individual weights of each neuron and their bias.
- > There is only a limit to the accuracy of the system, and training the model more will cause overfitting.
- > The system requires a high performance computer to operate as it is very computationally intensive.
- > If the system is run online then it requires a high bandwidth reliable connection to work properly.

Alternatives Includes -

- > Taking a pre-computed weight file for the model.
- > Changing the model using a less computational model

Trade-off -

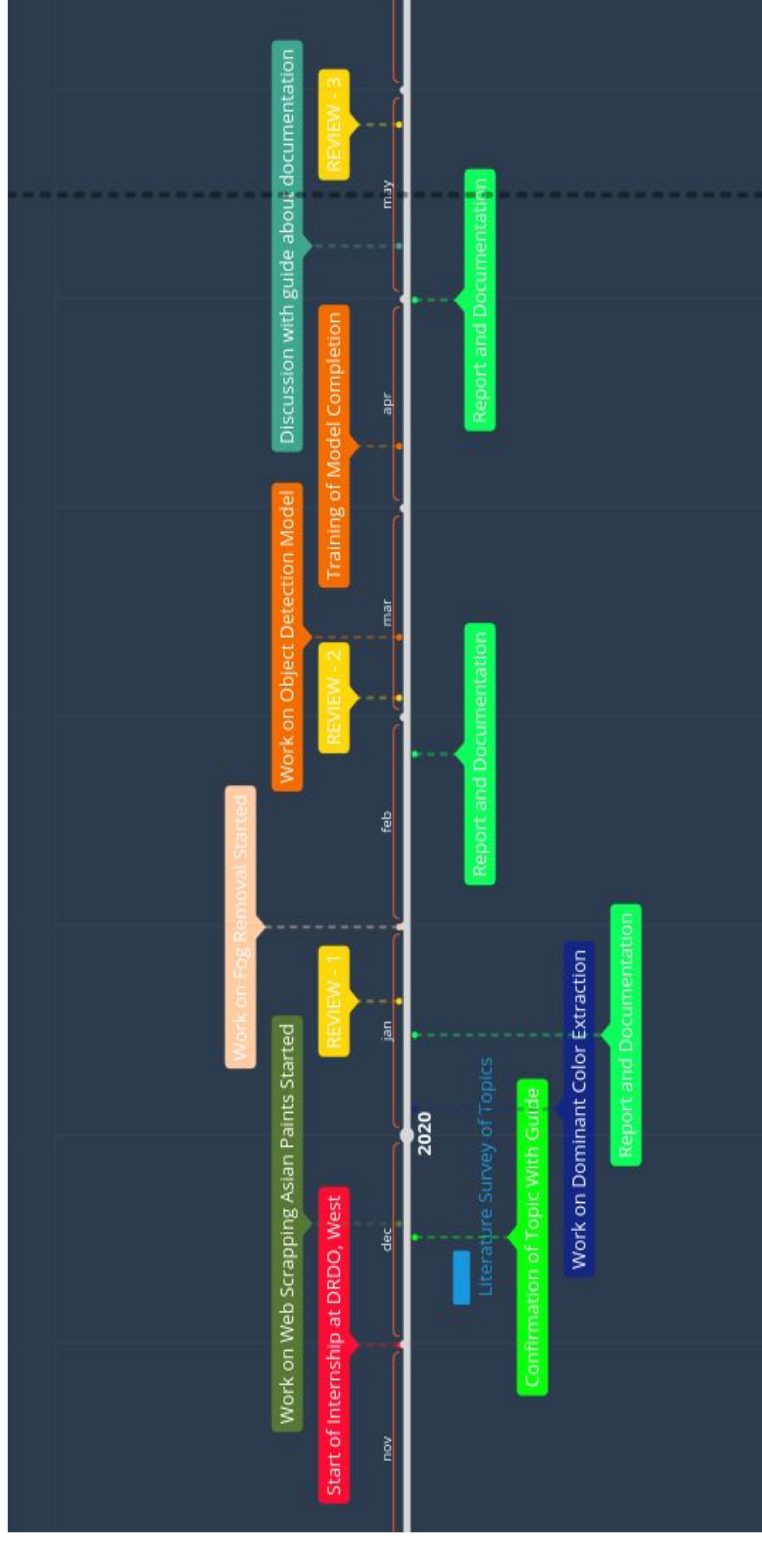
- > The model will only be able to identify the type of objects that are used as training objects, at any point if there is an introduction of new objects which the model has not seen, it will either fail or give bad results.
- > The model will reduce the manpower but always needs a constant supervision because it is not 100% accurate in its findings.
- > Model will consume a lot of power which if used on a low performance computer might either increase the time taken in producing the result or worse it might overwhelm the system causing the entire system failure. This would cause even more problems as it might affect other systems running on the same computer.

5. SCHEDULE TASKS AND MILESTONES

The beginning of my internship started at 1st December 2019 at DRDO Jodhpur. At the starting of the project only the milestones and objectives for the project were laid out in rough form for me to work upon.

Milestone	Tasks	Date of Completion
1. Creating a list of objectives to be completed during the phase of internship	<ul style="list-style-type: none"> - Create list of objectives - Approval of objectives - Study of Respective Tools to be used 	7 th Dec 2019
2. Acceptance by guide	<ul style="list-style-type: none"> - Acceptance of objectives by mentor at DRDO - Acceptance of objectives by Dr. Jayakumar K 	10 th Dec 2019
3. Literature Survey	<ul style="list-style-type: none"> - Study of Research Survey Papers for Different Technology 	14 th Dec 2019
4. Work on Web Scraping	<ul style="list-style-type: none"> - Creating CSV File for library 	25 th Dec 2019
5. Final Confirmation by project Guide	<ul style="list-style-type: none"> - Feasibility and Minimum functional and non-functional requirements acceptance 	30 th Dec 2019
6. Dominant Color Extraction	<ul style="list-style-type: none"> - Completion of Dominant Color extraction - Testing the model 	7 th Jan 2020
7. Review - 1	<ul style="list-style-type: none"> - Discussion with the guide - Completion of documentation and PPT 	22 nd Jan 2020
8. Fog Removal	<ul style="list-style-type: none"> - Study of different Technique - Implementation of Fog Removal - Acceptance of prototype by 	10th Feb 2020

	mentor - Documentation	
9. Model Detection	- Study of Different Models - Literature Survey of Different Models	15 th Feb 2020
10. Review 2	- Discussion with guide - Documentation of Work Completed - Plagiarism Check of Report - Creating Presentation	4 th Mar 2020
11. Work on Object Detection Model	- Start studying the different models - Approval of the architecture - Search Dataset	12 th Mar 2020
12. Training Model Completion	- Complete the Model with multiple epochs - Return Inference Report	14 th Apr 2020
13. Report and Documentation	- Discussion with Guide - Completion of Report for submission - Check for Plagiarism in Report created for final review.	1 st May 2020



6. PROJECT DEMONSTRATION

Here the demonstration of the project is done according to the components and each component is explained separately -

1) Parsing the Asian Paints Library for entire color palette -

The parsing was done using python library, to get the following column fields there were 1800 values -

- Color URL
- Shade Code
- Shade Name
- Color Family
- Red Value
- Green Value
- Blue Value

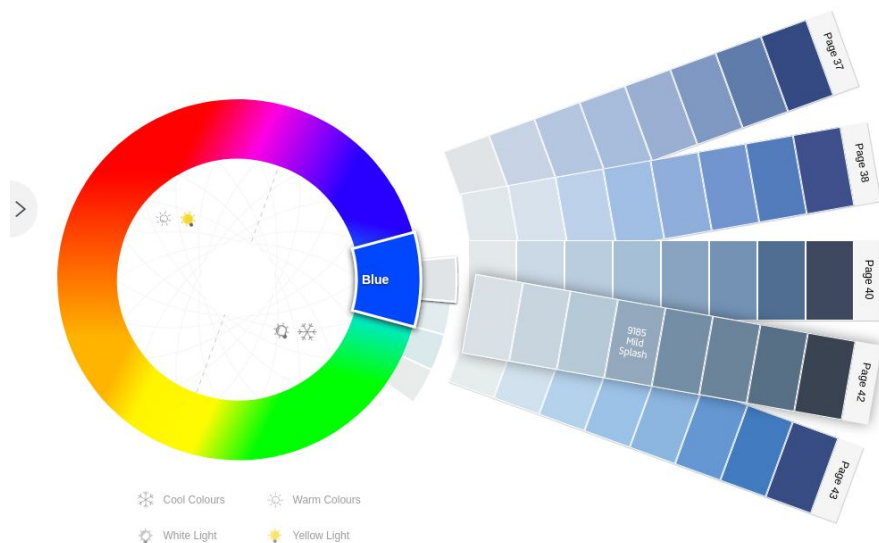


Fig6.1 - Color Palette Showing all the Colors in the Database

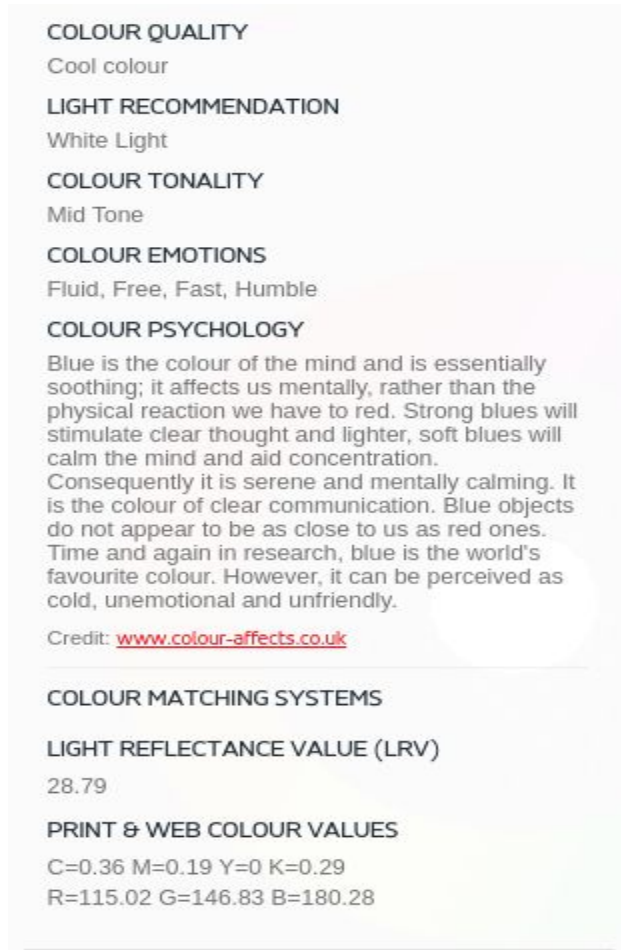


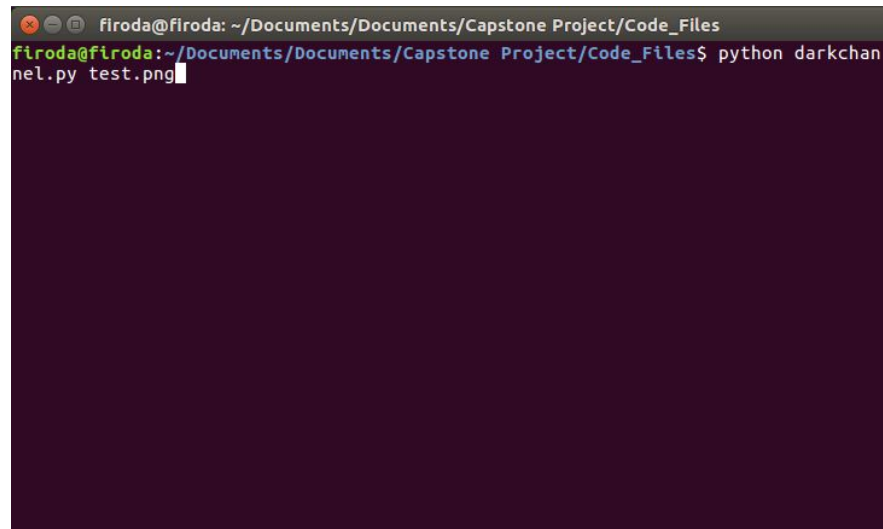
Fig6.2 - Color Details of the Selected Color

The figure above shows the various values shown for each color shade selected from the color palette. It shows values such as color quality, tonality, LRV Value, RGB Value, CMYK values etc.

Due to the nature of requirements only the relevant fields were scrapped from the website.

2. Fog Removal from the Image using Dark Channel Prior

Fig 6.3 - Terminal Screen for Starting darkchannel program



The python program takes the program file and the image path as input and it also requires that the system has python 2.x or higher installed on the system.



Fig 6.4 - Input Foggy Image test.png

We start by giving a foggy input image, the image provided does not give any information about the depth of the fog or the atmospheric light in the scene.



Fig 6.5 - Formation of Dark Channel for the input Image

As we can see the dark channel of the image, we observe that certain parts of the image have extremely dark color, these are the regions which will have near zero fog and we do not need to modify such regions, while others have a much lower color. We still have to take into account the atmospheric light but we have a general sense of depth of the smoke.



Fig 6.6- Transmission Map of the Image

We calculate the refined transmission map of the image, which shows the atmospheric light in the region. Using this and dark channel we will be able to calculate the fog free image.

3. Dominant Color Extraction -



Fig6.7 - Input Image

We take the following input image which has multiple colors and find the dominant colors.

```
[[186 149 140]
 [ 40  60 100]
 [238 206 171]
 [114 105 121]]
```

Fig - RGB values which are dominant colors in image.

We find the dominant colors in an image and store their RGB values in a list for our inspection, or further usage.

4. Object Detection Model Resnet -

```
[ ] !git clone https://github.com/EscVM/0IDv4_ToolKit.git

Cloning into '0IDv4_ToolKit'...
remote: Enumerating objects: 422, done.
remote: Total 422 (delta 0), reused 0 (delta 0), pack-reused 422
Receiving objects: 100% (422/422), 34.08 MiB | 36.58 MiB/s, done.
Resolving deltas: 100% (146/146), done.
```

Fig 6.8- Cloning the OID Toolkit

We clone the toolkit directory using the git clone command which copies the entire directory into the local runtime of google colab VM.

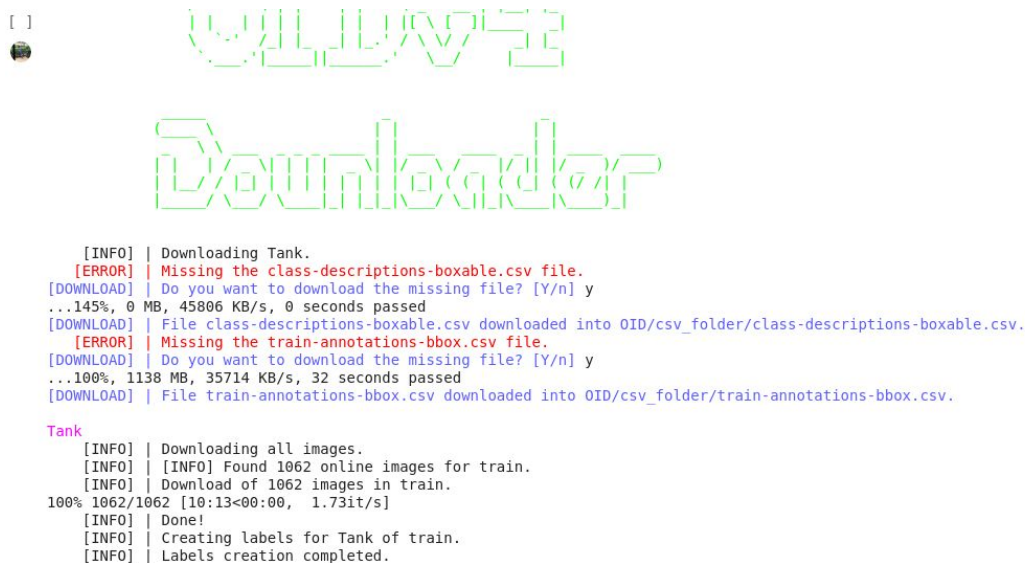
If we intend to use this toolkit multiple times then we can consider copying it into our google drive for further use.

```
!pip install -r requirements.txt

/content/gdrive/My Drive/OIDv4 Toolkit
Requirement already satisfied: pandas in /usr/local/lib/python3.6/dist-packages (from -r requirements.txt (line 1)) (1.0.3)
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from -r requirements.txt (line 2)) (1.18.4)
Collecting awscli
  Downloading https://files.pythonhosted.org/packages/db/79/d35f7a279ca09ede8b219549fee3c3195fef3d242d089aa8b2a678c8dc06/awscli-1.18.61-py2.py3-no
    3.0MB 4.8MB/s
Requirement already satisfied: urllib3 in /usr/local/lib/python3.6/dist-packages (from -r requirements.txt (line 5)) (1.24.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.6/dist-packages (from -r requirements.txt (line 7)) (4.41.1)
Requirement already satisfied: opencv-python in /usr/local/lib/python3.6/dist-packages (from -r requirements.txt (line 9)) (4.1.2.30)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-packages (from pandas->-r requirements.txt (line 1)) (2018.9)
Requirement already satisfied: python-dateutil>=2.6.1 in /usr/local/lib/python3.6/dist-packages (from pandas->-r requirements.txt (line 1)) (2.8.1)
Collecting botocore==1.16.11
  Downloading https://files.pythonhosted.org/packages/b0/4c/36f5203d7c9a7a9c25fc8185db523483168e44363fe9bc6e757533163b5e/botocore-1.16.11-py2.py3-no
    6.2MB 34.5MB/s
Collecting rsa<=3.5.0,>=3.1.2
  Downloading https://files.pythonhosted.org/packages/e1/ae/baedc9cb175552e95f3395c43055a6a5e125ae4d48a1d7a924baca83e92e/rsa-3.4.2-py2.py3-none-an
    51kB 6.9MB/s
Requirement already satisfied: PyYAML<5.4,>=3.10; python version != "3.4" in /usr/local/lib/python3.6/dist-packages (from awscli->-r requirements.txt (line 1)) (4.2.1)
Collecting colorama<0.4.4,>=0.2.5; python version != "3.4"
  Downloading https://files.pythonhosted.org/packages/c9/dc/45cdef1b4d119eb96316b3117e6d5708a08029992b2fee2c143c7a0a5cc5/colorama-0.4.3-py2.py3-no
    51kB 6.9MB/s
Requirement already satisfied: s3transfer<0.4.0,>=0.3.0 in /usr/local/lib/python3.6/dist-packages (from awscli->-r requirements.txt (line 3)) (0.3.0)
Requirement already satisfied: docutils<0.16,>=0.10 in /usr/local/lib/python3.6/dist-packages (from awscli->-r requirements.txt (line 3)) (0.15.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from python-dateutil>=2.6.1->pandas->-r requirements.txt (line 1)) (1.12.0)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /usr/local/lib/python3.6/dist-packages (from botocore==1.16.11->awscli->-r requirements.txt (line 1)) (0.9.4)
Requirement already satisfied: pvasn<=0.1.3 in /usr/local/lib/python3.6/dist-packages (from rsa<=3.5.0,>=3.1.2->awscli->-r requirements.txt (line 1)) (0.1.3)
```

Fig 6.9- Installing dependencies

We install dependencies required for functioning of the toolkit such as numpy, pandas, urllib3, tqdm etc. We use the requirements.txt as a reference to install the correct version of the library for proper functioning and avoiding any issues due to different version of the library.



```
[ ]
[INFO] | Downloading Tank.
[ERROR] | Missing the class-descriptions-boxable.csv file.
[DOWNLOAD] | Do you want to download the missing file? [Y/n] y
...145%, 0 MB, 45806 KB/s, 0 seconds passed
[DOWNLOAD] | File class-descriptions-boxable.csv downloaded into OID/csv_folder/class-descriptions-boxable.csv.
[ERROR] | Missing the train-annotations-bbox.csv file.
[DOWNLOAD] | Do you want to download the missing file? [Y/n] y
...100%, 1138 MB, 35714 KB/s, 32 seconds passed
[DOWNLOAD] | File train-annotations-bbox.csv downloaded into OID/csv_folder/train-annotations-bbox.csv.

Tank
[INFO] | Downloading all images.
[INFO] | [INFO] Found 1062 online images for train.
[INFO] | Download of 1062 images in train.
100% 1062/1062 [10:13<00:00, 1.73it/s]
[INFO] | Done!
[INFO] | Creating labels for Tank of train.
[INFO] | Labels creation completed.
```

Fig6.10 - Downloading Tank Dataset

We download the tank dataset from the Google Open Image Dataset using this toolkit it downloads all the images of the class specified and also download the annotations of the specific class. At this point we mount the google drive to save the copy of our Tank dataset as we will be using it multiple times so we give a path to Google drive for saving.

Saving can also be done as a local run-time variable but every time the run-time is reset or restarted then the entire dataset have to be downloaded again which leads to wastage of the bandwidth.

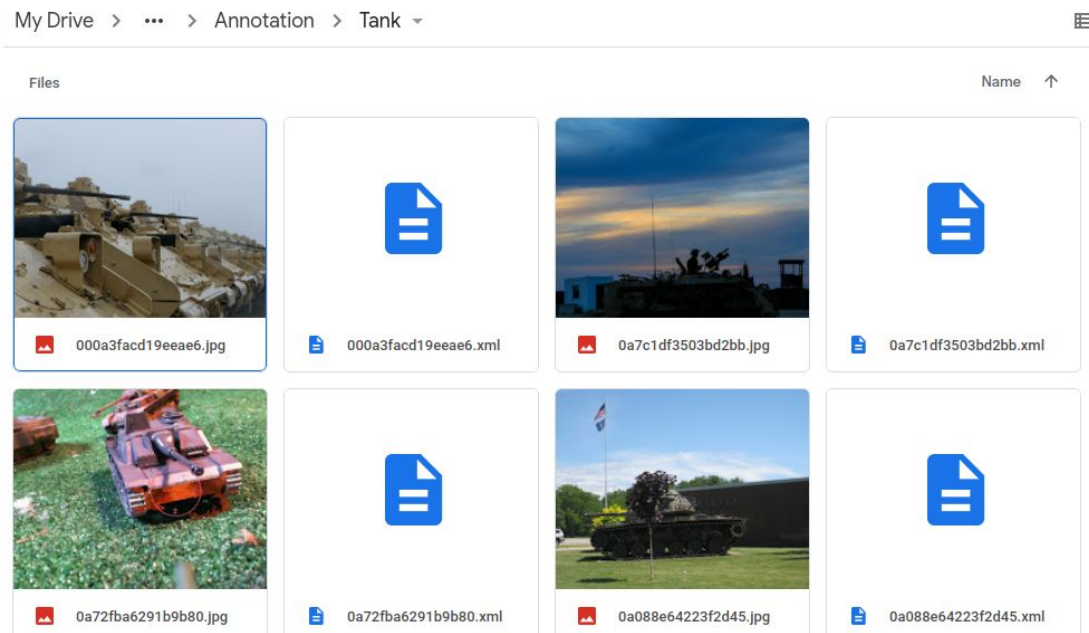


Fig 6.11- Tank Dataset saved in Google Drive

This is dataset stored in the google drive it consists of 3 part directory -

Tank

Test

Image

Annotation

Train

Image

Annotation

Validation

Image

Annotation

```
!python OIDv4_to_VOC.py --sourcepath '/content/gdrive/My Drive/OID_DATASETTank/train/Tank' --dest_path '/content/gdrive/My Drive/OID_DATASETTank/tr
```

Fig - Conversion of .txt file to Pascal VOC format

```
<annotation><folder>open_images_volume</folder><filename>000a3facd19eeae6.jpg</filename>
<path>/mnt/open_images_volume/000a3facd19eeae6.jpg</path><source><database>Unknown</database></source><size>
<width>5472</width><height>3648</height><depth>3</depth></size><segmented>0</segmented><object><name>Tank</name>
<pose>Unspecified</pose><truncated>0</truncated><difficult>0</difficult><bndbox><xmin>0</xmin><ymin>188</ymin>
<xmax>5389</xmax><ymax>3644</ymax></bndbox></object><object><name>Tank</name><pose>Unspecified</pose>
<truncated>0</truncated><difficult>0</difficult><bndbox><xmin>5208</xmin><ymin>1912</ymin><xmax>5468</xmax>
<ymax>2248</ymax></bndbox></object></annotation>
```

Fig6.12 - XML annotation file

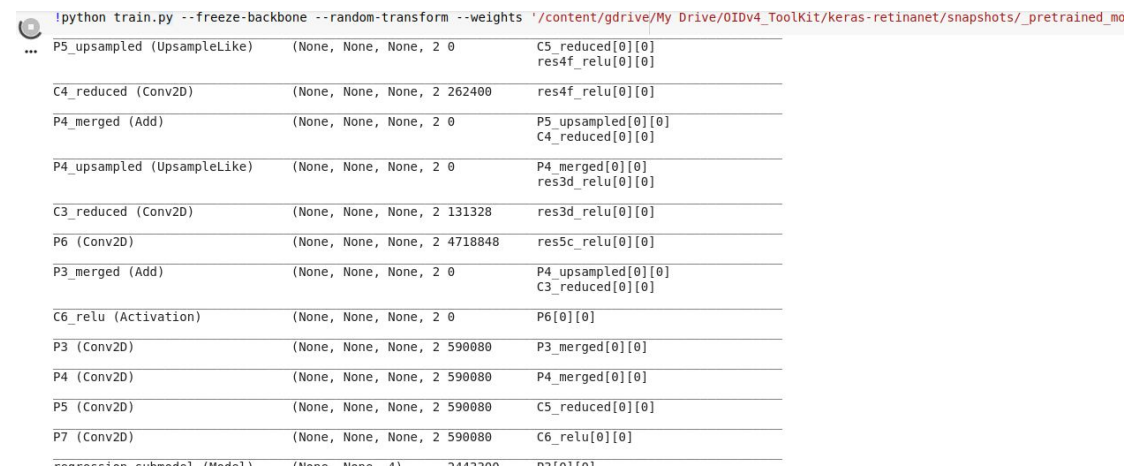
The resnet model takes the image annotation as a XML file in Pascal VOC format whereas the dataset downloaded from google is having the annotation in txt format so first the files have to be converted.

A1	fx	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/be886c1972ef0da3.jpg					
			B	C	D	E	F
1	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/be886c1972ef0da3.jpg		173	0	876	620	Tank
2	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/be886c1972ef0da3.jpg		798	64	958	292	Tank
3	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/61f54f030e93f5d0.jpg		0	0	1023	423	Tank
4	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/5dfbdbc915692e6.jpg		160	19	628	561	Tank
5	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/00b7dfa6afa343d0.jpg		182	92	692	412	Tank
6	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/4fe42df40579a9c4.jpg		0	0	482	359	Tank
7	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/a1d2ecf8bd302c27.jpg		0	126	766	655	Tank
8	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/a1d2ecf8bd302c27.jpg		835	264	1023	488	Tank
9	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/d1ecd5e92561ed65.jpg		79	153	670	532	Tank
10	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/6bf0658e3e157848.jpg		16	293	348	565	Tank
11	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/6bf0658e3e157848.jpg		611	308	992	555	Tank
12	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/6bf0658e3e157848.jpg		983	339	1023	430	Tank
13	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/84ac4bdf03061cc5.jpg		414	389	914	765	Tank
14	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/e7062a2feb412e04.jpg		0	37	1023	767	Tank
15	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/d339a4a95eb9b935.jpg		19	118	185	240	Tank
16	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/d339a4a95eb9b935.jpg		473	67	587	139	Tank
17	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/d339a4a95eb9b935.jpg		899	38	1022	129	Tank
18	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/8bec5f2dbfea2ca2.jpg		142	371	836	550	Tank
19	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/01d793b58ee81477.jpg		0	299	119	419	Tank
20	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/01d793b58ee81477.jpg		0	296	121	418	Tank
21	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/01d793b58ee81477.jpg		188	289	297	406	Tank
22	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/01d793b58ee81477.jpg		188	286	298	407	Tank
23	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/01d793b58ee81477.jpg		273	258	542	451	Tank
24	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/01d793b58ee81477.jpg		275	259	542	459	Tank
25	/content/gdrive/My Drive/OID_DATASETTank/train/Annotation/Tank/01d793b58ee81477.jpg		515	318	656	385	Tank

Fig6.13- Annotation.csv file showing the bounding boxes with the image location

The xml files are then parsed to create a csv file with the bounding box description of the objects in the image, each box on every image get a single entry.

At the time of training images were ~1200 and the csv file has ~1700 entries for the images. The csv file had more entries due to cases where there are multiple tanks in a single image.



```
!python train.py --freeze-backbone --random-transform --weights '/content/gdrive/My Drive/0IDv4_ToolKit/keras-retinanet/snapshots/_pretrained_mo
```

P5_upsampled (UpsampleLike)	(None, None, None, 2 0	C5_reduced[0][0] res4f_relu[0][0]
C4_reduced (Conv2D)	(None, None, None, 2 262400	res4f_relu[0][0]
P4_merged (Add)	(None, None, None, 2 0	P5_upsampled[0][0] C4_reduced[0][0]
P4_upsampled (UpsampleLike)	(None, None, None, 2 0	P4_merged[0][0] res3d_relu[0][0]
C3_reduced (Conv2D)	(None, None, None, 2 131328	res3d_relu[0][0]
P6 (Conv2D)	(None, None, None, 2 4718848	res5c_relu[0][0]
P3_merged (Add)	(None, None, None, 2 0	P4_upsampled[0][0] C3_reduced[0][0]
C6_relu (Activation)	(None, None, None, 2 0	P6[0][0]
P3 (Conv2D)	(None, None, None, 2 590080	P3_merged[0][0]
P4 (Conv2D)	(None, None, None, 2 590080	P4_merged[0][0]
P5 (Conv2D)	(None, None, None, 2 590080	C5_reduced[0][0]
P7 (Conv2D)	(None, None, None, 2 590080	C6_relu[0][0]

Fig6.14 - Resnet Model Being Trained

The model is then being trained with the annotation and csv file path as input.

The model has following values -

Batch Size - 8

Steps - 75

Epochs - 10

The model uses the weights of a model which was trained on a COCO dataset this helps in speeding up the process.

The COCO model is a dataset which consists of around 330000 images with around more than 20000 images labeled. This model contains around images with ~80 classes and is frequently used as a starting point for training custom models. The model itself can be used to identify objects but is much more prone to error as compared to models that are specifically trained to identify specific types of objects.

Further this model also does not work fast enough compared to specifically trained models which are used as they have a much limited set of labels to identify from.(Generally speaking)

```
75/75 [=====] - 8675s 116s/step - loss: 2.6380 - regression_loss: 1.6486 - classification_loss: 0.9894

Epoch 00001: saving model to ./snapshots/resnet50_csv_01.h5
/usr/local/lib/python3.6/dist-packages/keras/callbacks/callbacks.py:846: RuntimeWarning: Early stopping conditioned on metric `mAP` which is
not available in the monitor: `'.join(list(logs.keys()))'`, RuntimeWarning
Epoch 2/10

50/75 [=====>.....] - ETA: 48:12 - loss: 2.2372 - regression_loss: 1.5034 - classification_loss: 0.7338
```

Fig6.15 - Model Being Trained and saved after each epoch

Further after each epoch the model is saved so that in case there is need the shut down the system to continue later, which can be done on such systems. Also if we find that the model is being increasingly overfitting in nature then the model needs to be stoped as overall performance will decrease.

Multiple saves means we can test which of the (.h5) models can work best for the user requirements.

7. RESULT AND DISCUSSION

After rigorous testing of the prototype model, which has multiple components the results of the project are shown in this section. There is also inclusion of discussion about the results obtained through the system.

1. Parsing the Asian Paints Library for entire color palette

	A	B	C	D	E	F	G
1	url	Shade Code	Shade Name	Color Family	Red	Green	Blue
2	https://www	7276	Brush Stroke	blue	224.52	228.33	231.66
3	https://www	7275	Blue Dawn	blue	200.64	212.19	229.82
4	https://www	7274	Summer Sky	blue	181.28	198.42	224.88
5	https://www	7273	Scuba Blue	blue	168.61	188.74	220.69
6	https://www	7272	Washed Indigo	blue	154.84	175.1	209.88
7	https://www	7271	Blue Pottery	blue	127.92	153.31	195.84
8	https://www	1218	Blue Clover	blue	96.94	124.35	171.28
9	https://www	1306	Oriental Blue	blue	53.1	74.96	130
10	https://www	9172	Blue Smoke	blue	225.3	232.08	235.77
11	https://www	9171	Blue Blaze	blue	215.17	226.12	236.41
12	https://www	9170	Cloud Blue	blue	190.1	209.8	234.97
13	https://www	9169	Pristine Blue	blue	161.6	190.04	229.02
14	https://www	9168	Blue Effect	blue	143.84	174.83	220.88
15	https://www	9167	Blue Illusion	blue	114.35	149.97	207.75
16	https://www	9166	Rare Blue	blue	82.92	124.27	187.46
17	https://www	9165	Ink Blue Sky	blue	63.79	80.02	141.17
18	https://www	7300	Jet Stream	blue	227.25	233.23	234.99
19	https://www	1243	Classic	blue	202.61	217.44	229.5
20	https://www	7298	Sail Away	blue	185.41	205.25	223.57
21	https://www	7297	Blissfull Blue	blue	165.28	191.02	215.62
22	https://www	7296	Far Horizons	blue	136.84	164.7	193.58
23	https://www	7295	Picture Perfect	blue	115.02	146.83	180.28

Fig7.1 - CSV file created by scrapping the website

The parser is able to get all 1800 entries from the asian paints website with their shade name, color family and RGB values respectively.

2. Fog Removal from the Image using Dark Channel Prior



Fig7.2 - Output Enhanced Image

By using the above images we calculate the fog-free image which has fog removed from the areas where there are objects, while you can see that it acts peculiarly in areas where there was sky because it has problems in removing fog from light coloured areas such as blue skies.

3. Dominant Color Extraction -



Fig7.3 - Output Image using only the dominant colors

We use the dominant colors to recreate the image and we see that we lose very less detail which helps us in creating image while using a limited set of colors.

4. Object Detection Resnet -



Fig7.4 - Model Showing Inference in the picture

Here the model is able to detect armoured vehicle in 5.74 seconds of processing time. This model will be helpful in case where there is a lot of data coming and the model can alert if it sees the anomaly such as an armoured vehicle approaching.

Comparison of Models -

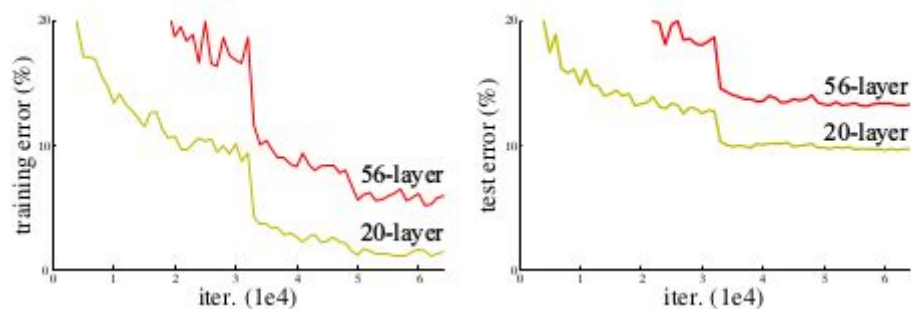


Fig7.5 - Performance of models with 56 and 20 layers respectively on both the training and testing dataset.

Here we can see the training error rate of conventional neural network models when they are added more layers they start performing even worse.

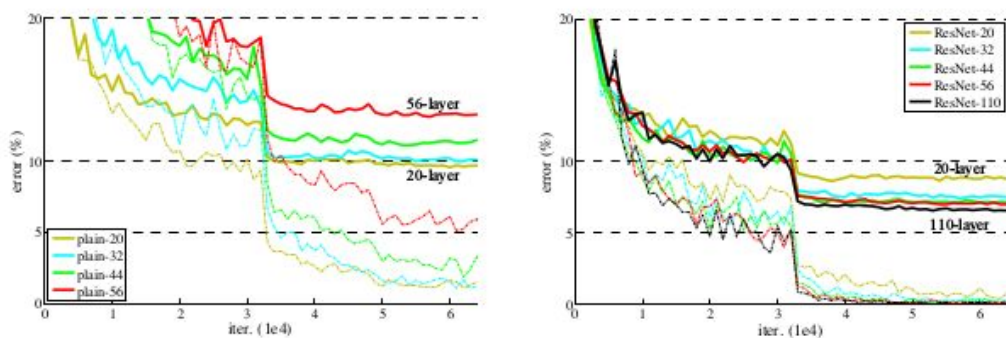


Fig 7.6 - Results of Vanilla DNN compared to Resnet Model when trained on CIFAR-10 dataset

As we see there is a sharp drop in training error percentage in the vanilla networks as more and more layers are stacked against the neural network. When this model is tested on the CIFAR-10 the models with less layers are performing better than models with more layers.

This compared to the ResNet Model which is performing equally well compared to lower less deep neural networks due to its property of skip connections which help it get better results with more number of layers.

8. SUMMARY

After the complete development of all the components of the projects. This project serves as an addition to the detection systems to detect armoured vehicles across a variety of landscapes. This project will help in reducing the workload of humans which have to spend more hours in classifying and analyzing the data very frequently and this project can be scaled at large. Using this project's components such as the removal of fog can help in detection of vehicles even in heavily dense fogged images which otherwise would make the model not able to detect or at worst overwhelm the system or simply mark for human read.

The dominant color detection will help in further training the model in cases where the training dataset sizes are small as further images with less colors can be used to train.

In a nutshell these improved pre-processing techniques coupled with object detection models with large numbers of layers can easily be trained and used in place of an actual human.

9. APPENDIX A

[1]. Color Segmentation Using an Eigen Color Representation

Authors: Alaa E. Abdel-Hakim and Aly A. Farag Computer Vision and Image Processing Laboratory (CVIP Lab.) University of Louisville, Louisville/40292, KY USA

[2] Object Recognition in Images using Convolutional Neural Network

Authors: Mr.Sudharshan Duth P and Ms.Swathi Raj Dept. of Computer Science Amrita School of Arts and Sciences Mysuru, Karanataka, India

[3] towardsdatascience.com

[4] https://github.com/AtriSaxena/OIDv4_to_VOC.git

[5] Single Image Fog Removal Using Bilateral Filter

[6] Authors: Abhishek Kumar Tripath and Sudipta Mukhopadhyay Dept. of Electronics & Electrical Communication Engineering, Indian Institute of Technology, Kharagpur, INDIA 721302

[7] Single image de hazing with a physical model and dark channel prior Authors: Jin-Bao Wanga, Ning Hea,n, Lu-Lu Zhanga, Ke Lub Beijing Key Laboratory of Information Service Engineering, College of Information Technology, Beijing Union University, Beijing 100101, China b Chinese Academy of Sciences, Beijing 100049, China

[8] <https://colab.research.google.com/drive/1v3nzYh32q2rm7aqOaUDvqZVUmShicAsT>