An Industry Oriented Mini-Project (CS705PC)

on

HARD HAT DETECTION AT CONSTRUCTION SITE

Submitted

in partial fulfilment of the requirements

for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

B. NAGA JAYAKRISHNA

(17261A0505)

Under the guidance

of

Dr. A. Nagesh

(Professor)

Department of Computer Science and Engineering

Mahatma Gandhi Institute of Technology

(Affiliated to Jawaharlal Nehru Technological University Hyderabad)

Kokapet(V), Gandipet(M), Hyderabad.

Telangana-500 075.

2020 - 2021

MAHATMA GANDHI INSTITUTE OF TECHNOLOGY

(Affiliated to Jawaharlal Nehru Technological University
Hyderabad) GANDIPET, HYDERABAD – 500 075. Telangana

CERTIFICATE

This is to certify that the thesis entitled "**Hard hat detection at construction site"** is being submitted by **B. NAGA JAYAKRISHNA** bearing **Roll no 17261A0505** in partial fulfillment for the award of **B.Tech** in **Computer Science and Engineering** to **Jawaharlal Nehru Technological University Hyderabad** is a record of bonafide work carried out by him under our guidance and supervision.

The results embodied in this project have not been submitted to any other University or Institute for the award of any degree or diploma.

Supervisor

Dr. A. Nagesh
(Professor)

Coordinators

Dr. C. R. K. Reddy

(Professor & HOD)

External Examiner

DECLARATION

This is to certify that the work reported in this project titled “**HARD HAT DETECTION AT CONSTRUCTION SITE**” is a record of work done by me in the **Department of Computer Science and Engineering, Mahatma Gandhi Institute of Technology**, Hyderabad.

No part of the work is copied from books/journals/internet and wherever the portion is taken, the same has been duly referred in the text. The report is based on the work done entirely by me and not copied from any other source.

B. NAGA JAYAKRISHNA

(17261A0505)

ACKNOWLEDGEMENT

I would like to express my sincere thanks to **Dr. K Jaya Sankar, Principal MGIT**, for providing the working facilities in college.

I wish to express my sincere thanks and gratitude to **Dr. C R K Reddy, Professor and Head of Department** of CSE, MGIT, for all the timely support and valuable suggestions during the period of project.

I am extremely thankful **to Dr. C R K Reddy, Professor & HOD, Ms.B.Prashanthi, Senior Associate Professor,** Department of CSE, MGIT, Mini Project coordinators for their encouragement and support throughout the project.

I am extremely thankful and indebted to my internal guide **Dr.A.Nagesh, Professor**, Department of CSE, for his constant guidance, encouragement and moral support throughout the project.

Finally, I would also like to thank all the faculty and staff of CSE Department who helped me directly or indirectly, for completing this project.

B. NAGA JAYAKRISHNA

(17261A0505)

Table of Contents

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Human works at construction site are hazardous and risky. These jobs include building houses, roads, dams, repair or maintenance of infrastructure. With the increase in this type of work, fatalities also increased. Taking proper measures can help mitigate the number of accidents. A hard hat is one such equipment required for a worker to enter a construction area. A hard hat is personal protective equipment, which looks like helmet to protect the head from injuries. This is predominantly used in workplace environments such as industrial or construction sites. It is necessary to stop a worker who does not wear a hard hat in construction area. It is also vital need for innovating different methods to monitor the safety of the workers at construction sites.

Main aim of this project is to improve workplace safety by detecting people and hard hats on 5k images with b-box annotations. Despite various safety inspections carried out over the years to ensure compliance with regulations and maintain acceptable and safe working conditions, construction is still among the most dangerous industries responsible for a large portion of the total worker fatalities. This project takes the initial steps and aims at evaluating existing computer vision techniques, namely object detection methods, in rapidly detecting whether workers are wearing hardhats from images captured on many indoor jobsites. We can get images from the CCTV cameras installed at construction site. We can make use of various pre trained models such as Resnet-18 to build a neural network model to detect hard hats. Finally, the extensive experiments on two datasets demonstrate that the proposed method is robust and can meet accuracy requirements for hard hat detection.

Keywords: Construction sites safety, Hardhat wearing detection, Convolutional neural network, Pytorch, Resnet-18.

1. INTRODUCTION

Despite various safety inspections carried out over the years to ensure compliance with regulations and maintain acceptable and safe working conditions, construction is still among the most dangerous industries responsible for a large portion of the total worker fatalities. Therefore, this project takes the initial steps and aims at evaluating existing computer vision techniques, namely object detection methods, in rapidly detecting whether workers are wearing hardhats from images captured on many indoor jobsites.

Due to increasing industrialization and population, there arises the need to provide jobs, shelter, and infrastructure for the smooth run of life. In order to accommodate these needs, there is a definite need to develop infrastructure in many developing and under developing countries leading to increase in construction field. Lot of manpower is deployed at construction sites who work under unsafe conditions, and many are losing their lives year on year. This great loss has been occurring to many workers families, nation and industries due to such unsafe conditions[7]. To safe guard the nation's workforce deployed at construction sites, there are methods to improve safety measures in construction fields which will be of vital importance if implemented.

The main reasons of construction site fatalities includes falls, slips, objects, electrocution, and being caught in between objects. In majority of the fall incidents, the workers fall off from great heights and their heads get hit on the floors. A study that was investigated showed that the mortality rate at construction sites increased because they did not use the safety equipment or had not used them properly. Head is the most critical part of a human body and is highly vulnerable to any impact that is caused due to serious injuries. Using protective hard hat in construction/industrial areas is highly recommended.

Workers are always expected to follow the occupational Safety and Health Administration (OSHA) regulations to have head protection equipment (e.g., hard hat). In spite of these recommendations, if the workers don't wear hard hats, they should be alerted at the entrance itself so as to improve safety measures.

1.1 Problem Statement

Identifying a worker if he/she wearing a hard hat or not, can be more challenging. Manually monitoring each and every worker at construction site is nearer to impossible task. Replacing this manual work with machines and models would be more feasible and more accurate with in the very less time compared to this ancient practice.

Keeping in mind, the growth in the number of construction projects that are in progress, there is a necessity of developing innovative methods which can automatically keep an eye on the safety of the workers. Safety at all levels is now more data driven. Our paper, aims at building a system to detect the use of hard hat i.e., System can check whether a worker wears the hard hat or not by 2 methods:

1. By scanning through the construction surveillance videos which are then divided into frames and hard hat identification is done.
2. By implementing a system which detects whether a worker is wearing hard hat or not when he enters the construction zone where a camera is fixed which would be integrated with our system.

1.2 Existing System

To enhance construction sites safety, the majority of existing works personally monitor the presence and proper use of hardhats. In the traditional supervision (visual monitoring) method, the safety engineers occasionally fail to enforce construction workers to use their hard hats because they could not check the workers every hour and every day.

However, the existing detection model has a poor performance when the images are not very clear, the safety helmets are too small and obscure, and the background is too complex. Moreover, the existing model is limited by the problems that some images of the dataset are less in quantity; the preprocessing operations of the images are confined to rotation, cutting, and zooming; the manual labeling is not comprehensive and may miss some objects. In some extreme cases, for example, only part of the head is visible and the safety helmet is obstructed, the model cannot detect the helmets accurately. This is the common limitation of the-state-of-art algorithms. Due to the above reasons, the detection performance is not good enough and there are some detection errors.

DISADVANTAGES

1. There will be delay in the monitoring the workers having hard hats.
2. The speed of analyzing and accuracy is very slow.

1.3 Proposed System:

We can get images from the CCTV cameras installed at construction site. Wired or Wireless network can be used to transfer these images to server.

Images of the construction site are displayed continuously on the office computers. Using real-time images, the software program detects whether the workers are using their hard hats properly.

The proposed detection method based on deep learning to detect safety helmets worn by workers provides an effective opportunity to improve safety management on construction sites. Previous studies have demonstrated the effectiveness of locating the safety helmets and workers and detecting the helmets. However, most of the studies have limitations in practical application. Sensor-based detection methods have a limited read range of readers and cannot be able to confirm the position relationship between the helmets and the workers. The machine learning-based detection methods choose features artificially with a strong subjectivity, a complex design process, and poor generalization ability. Therefore, the study proposed a method based on deep learning to detect safety helmets automatically using convolutional neural networks. The experimental results have suggested the effectiveness of the proposed method.

ADVANTAGES

1. Human intervention is not mandatory.
2. The speed of analysis is very fast.
3. Hard hats will be identified very accurate

1.4 Requirements Specification:

1.4.1 Software Requirements

Language	: Python3
Operating system	: windows 7 or 10

Tools

- Anaconda
- Jupyter Notebook
- Python 3.5
- Google Colaboratory

1.4.2 Hardware Requirements

System	:	Pentium Dual Core.
Hard Disk	:	120 GB.
Monitor	:	“14” and above LED
Input Devices	:	Keyboard, Mouse
Ram	:	1GB.

2. LITERATURE SURVEY

Related research into safety helmets detection

As of now, past investigations of wellbeing protective caps identification can be partitioned into three sections, sensor-based recognition, AI based discovery, and profound learning-based location. Sensor-based identification for the most part finds the security head protectors and laborers (Kelm et al. [4], Torres et al. [5]). The techniques typically utilize the RFID labels and perusers to find the head protectors and laborers and screen how close to home defensive gear is worn by laborers progressively. Kelm et al. [4] planned a versatile Radio Frequency Identification (RFID) entrance for checking individual defensive hardware (PPE) consistence of faculty. In any case, the working scope of the RFID perusers is restricted and the RFID perusers can just propose that the security protective caps are near the specialists yet unfit to affirm that the wellbeing head protectors are as a rule appropriately worn.

Modern, AI based article recognition advances are generally utilized in numerous areas for its amazing item discovery and arrangement limit (e.g., Rubaiyat et al. [6], Shrestha et al. [7], Waranusast et al. [8], Doungmala et al. [9], Jia et al. [10], and Li et al. [11]). Striking investigations are made by Rubaiyat et al. [6], who proposed a programmed location technique to acquire the highlights of development laborers and security protective caps and distinguish wellbeing head protectors. The strategy consolidates the recurrence space data of the picture with the histogram of situated inclination (HOG) and the circle Hough change (CHT) extractive method to distinguish the laborers and the caps in two stages. The discovery strategies dependent on AI can recognize security caps precisely and unequivocally under different situations yet additionally have a few disadvantages. Now and then the technique can just recognize wellbeing protective caps with a particular tone and it is hard to recognize the caps with comparable tone and shape to the security caps. Besides, the strategy can't identify countenances and wellbeing head protectors altogether under certain conditions; for instance, a few specialists don't turn their appearances towards the camera at the building site.

1. **Bahaa Eddine Mneymneha, Mohamad Abbasa, Hiam Khoury, "Automated Hardhat Detection for Construction Safety Applications"**

Despite various safety inspections carried out over the years to ensure compliance with regulations and maintain acceptable and safe working conditions, construction is still among the most dangerous industries responsible for a large portion of the total worker fatalities. A construction worker has a chance of 1-in-200 of dying on the job during a 45-year career, mainly due to fires, falls, and being struck or caught-in/between objects. This in part can be attributed to how monitoring the presence and proper use of personal protective equipment (PPE) by safety officers becomes inefficient when surveying large areas and a considerable number of workers. Therefore, this paper takes the initial steps and aims at evaluating existing computer vision techniques, namely object detection methods, in rapidly detecting whether workers are wearing hardhats from images captured on many indoor jobsites. Experiments have been conducted and results highlighted the potential of cascade classifiers, in particular, in accurately and precisely detecting hardhats under various scenarios and for repetitive runs.

2. Kishor Shrestha, Pramen P. Shrestha, Dinesh Bajracharya, Evangelos A. Yfantis, "Hard-Hat Detection for Construction Safety Visualization"

In 2012, 775 fatalities were recorded, and many more were injured at construction sites in the United States. Of these, 415 fatalities (54%) were due to fall, slips, and trips as well as being struck by falling objects. In order to decrease fatalities at construction sites to these types of events, the Occupational Safety and Health Administration (OSHA) provides Fall Prevention and OSHA-10 trainings to construction workers. Moreover, safety personnel monitor whether the workers use personal protective equipment (PPE) properly. Data shows that construction fatalities have decreased by 2% annually since 1994; however, the owners still are not satisfied with this result. Various studies have shown that fall is the biggest contributor for construction fatalities. One study showed that half of the fall fatalities were because the workers either had not used PPEs or had not used them properly. In addition, studies showed that, with proper use of hard hats, the fatalities due to fall, slips, trips, and being struck by falling objects could be reduced. This study developed and tested a hard-hat detection tool that uses image-processing techniques to identify whether workers are wearing hard hats. The tool dispatches warning messages if the workers do not use hard hats.

SNO	YEAR	AUTHOR	TITLE	TECHNIQUE
1	2017	Bahaa Eddine Mneymneha, Mohamad Abbasa, Hiam Khoury.	Automated Hardhat Detection for Construction Safety Applications.	This section describes and evaluates existing computer vision techniques deemed useful for detecting hardhats. (1) Feature detection, extraction and matching, (2) template matching, and (3) cascade classifiers models.
2	2015	Kishor Shrestha, Pramen P. Shrestha, Dinesh Bajracharya, Evangelos A. Yfantis.	Hard-Hat Detection for Construction Safety Visualization	In this research, the software tool was developed in Microsoft Visual Studio 2012
3	2015	Jixiu Wu, Nian Cai, Guotian Wang	Automatic detection of hardhats worn by construction personnel.	Reverse progressive attention boosts performance for small object detection
4	2020	Yange Li, Han Wei, Zheng Han, Jianling Huang, Weidong Wang	Deep Learning-Based Safety Helmet Detection in Engineering Management Based on Convolutional Neural Networks	In the paper, the SSD-MobileNet algorithm is used to build the model. A dataset of 3261 images containing various helmets is trained and tested on the model.

Table 2.1 Literature Survey For the identification of hard hats

3. SYSTEM DESIGN for Hard Hat detection

3.1 Architectural Design

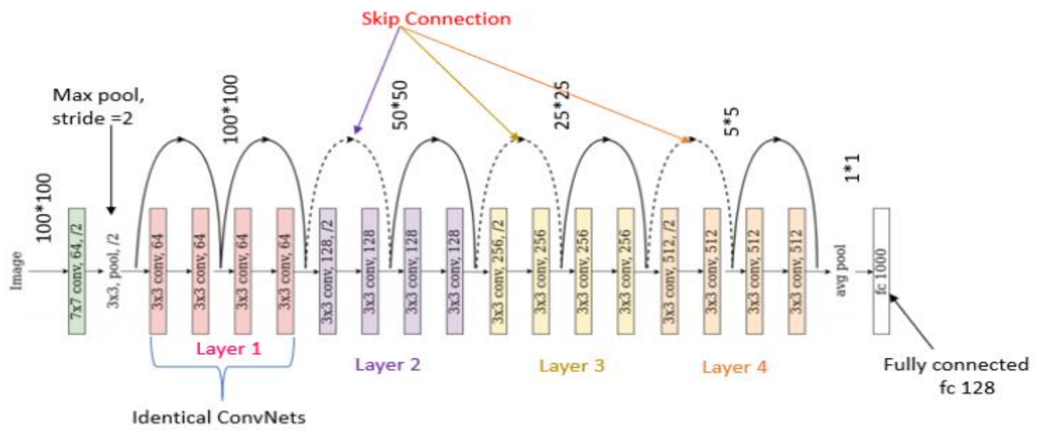


Figure 3.1: System Architecture

The above figure 3.1 represents the architecture of the ResNet 18 network. It is used in this project. A pretrained model has been previously trained on a dataset and contains the weights and biases that represent the features of whichever dataset it was trained on. Learned features are often transferable to different data. For example, a model trained on a large dataset of any object detection will contain learned features like edges or horizontal lines that you would be transferable to our dataset.

Pre-trained models are beneficial to us for many reasons. By using a pre-trained model you are saving time. Someone else has already spent the time and compute resources to learn a lot of features and your model will likely benefit from it.

The key technology of CNN is the local receptive field, sharing of weights, sub sampling by time or space, so as to extract feature and reduce the size of the training parameters. The advantage of CNN algorithm is that to avoid the explicit feature extraction, and implicitly to learn from the training data .Same neuron weights on the surface of the feature mapping, thus network can learn parallel, and reduce the complexity of the network Adopting sub sampling structure by time robustness, scale and deformation displacement .Input information and network topology can be a very good match, It has unique advantages in image processing and object detection. The Convolution Neural Network involves these steps.

Convolution Layer:

The convolution layer is the core building block of a CNN. The convolution layer comprises of set of independent feature detectors. Each Feature map is independently convolved with the images.

Pooling Layer:

Pooling layer function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network. Pooling layer operates on each feature map independently. The approaches used in pooling are:

- Max Pooling
- Mean Pooling
- Sum Pooling

Fully Connected Layer:

Neurons in the fully connected layer have full connections to all activations in the previous layer. In this the output obtained from max pooling is converted to one dimensional array and that should be the input layer and the process continues same as ANN model.

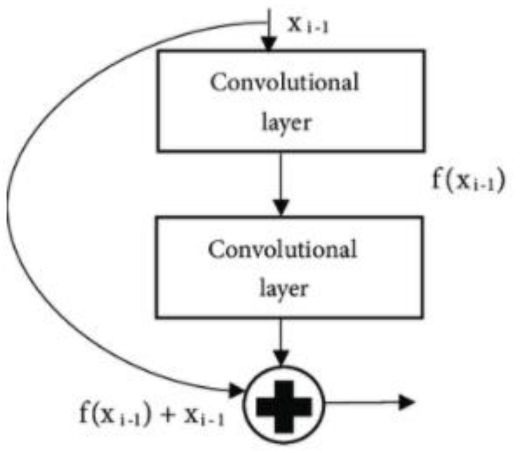
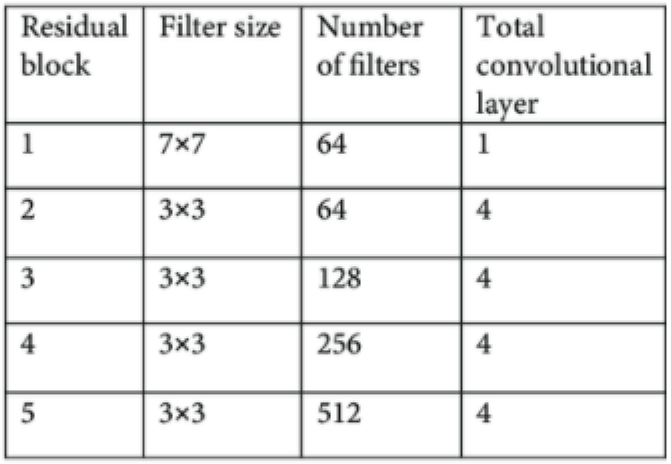


Figure 3.2 Basic building block of residual network



Residual block	Filter size	Number of filters	Total convolutional layer
1	7×7	64	1
2	3×3	64	4
3	3×3	128	4
4	3×3	256	4
5	3×3	512	4

Table 3.1 Filter information of different convolution layers

Image classification is often used to classify images into a single category, usually corresponding to the most salient object. Photos taken with mobile phones are usually complex and contain multiple objects.

Object detection models are therefore more appropriate to identify multiple relevant objects in a single image. The second significant advantage of object detection models is that localization of the objects is provided. Researchers often use different datasets like PASCAL VOC, COCO, ImageNet for object detection. First of all, to assess the spatial precision we need to remove the boxes with low confidence (usually, the model outputs many more boxes than actual objects). Then, we use the Intersection over Union (IoU) area, a value between 0 and 1. It corresponds to the overlapping area between the predicted box and the ground-truth box. The higher the IoU, the better the predicted location of the box for a given object. Usually, we keep all bounding box candidates with an IoU greater than some threshold.

The use of anchor boxes improves the speed and efficiency for the detection portion of a deep learning neural network framework. Anchor boxes are a set of predefined bounding boxes of a certain height and width. These boxes are defined to capture the scale and aspect ratio of specific object classes you want to detect and are typically chosen based on object sizes in your training datasets. During detection, the predefined anchor boxes are tiled across the image. The network predicts the probability and other attributes, such as background, intersection over union (IoU) and offsets for every tiled anchor box. The predictions are used to refine each individual anchor box. You can define several anchor boxes, each for a different object size. Anchor boxes are fixed initial boundary box guesses.

Using the ResNet 18 model and freezing it for the last two layers from it and applying the required functions to it to acquire the required results. Here I used the Retina Net Focal Loss function to calculate the loss function of model trained at each epoch by considering 6 number of anchors. It learned the required data by using the pascalVOC metric and BB metrics and retina net focal loss as the loss function.

To develop the tool that automatically detects workers who are not wearing hard hat in the real-time video, a visualization approach was used. The visualization approach is an innovative software tool that monitors the workers in real time and dispatches warning messages to concerned personnel once the safety rules (wearing hard hats properly at this stage) are violated. This approach consists of

(1) a closed-circuit television (CCTV) camera installed at construction site,
(2) a wired/wireless network to transfer videos taken by CCTV camera to the server (office computer),
(3) a server at the nearest office,
(4) Images of the construction site are displayed continuously on the office computers. Using real-time images, the software program detects whether the workers are using their hard hats properly. This tool was developed using Object detection techniques.

3.2 DATASET

A dataset is a collection of data. The dataset contains 5000 images with bounding box annotations in the PASCAL VOC format for these 3 classes:

- Helmet
- Person
- Head

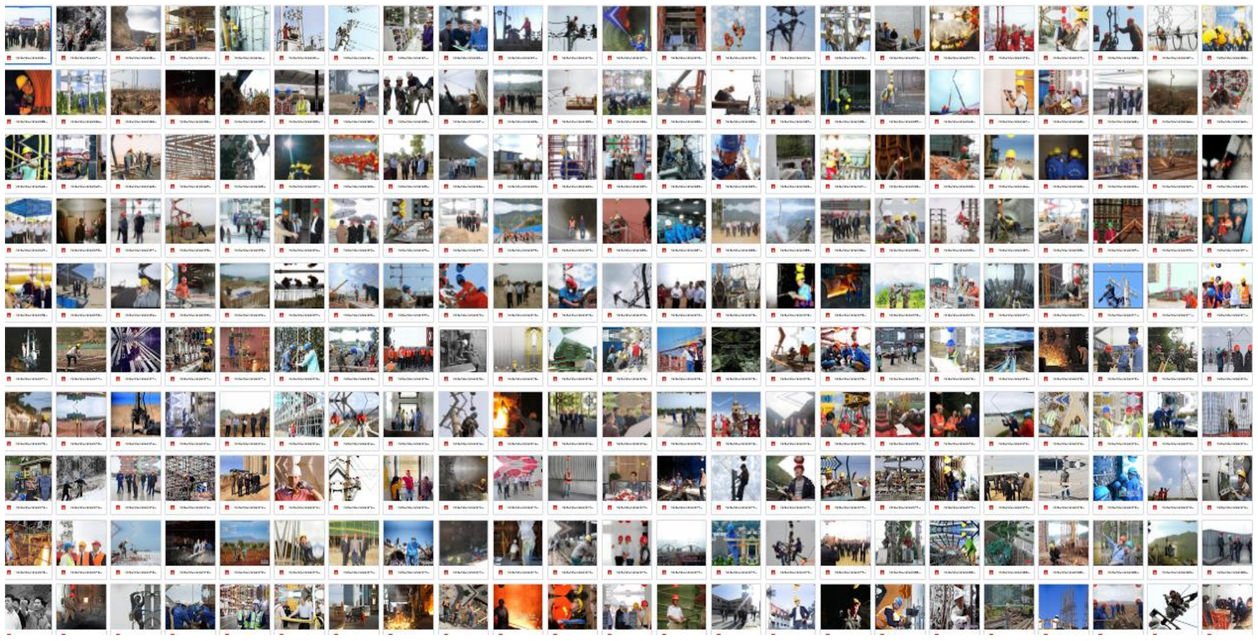
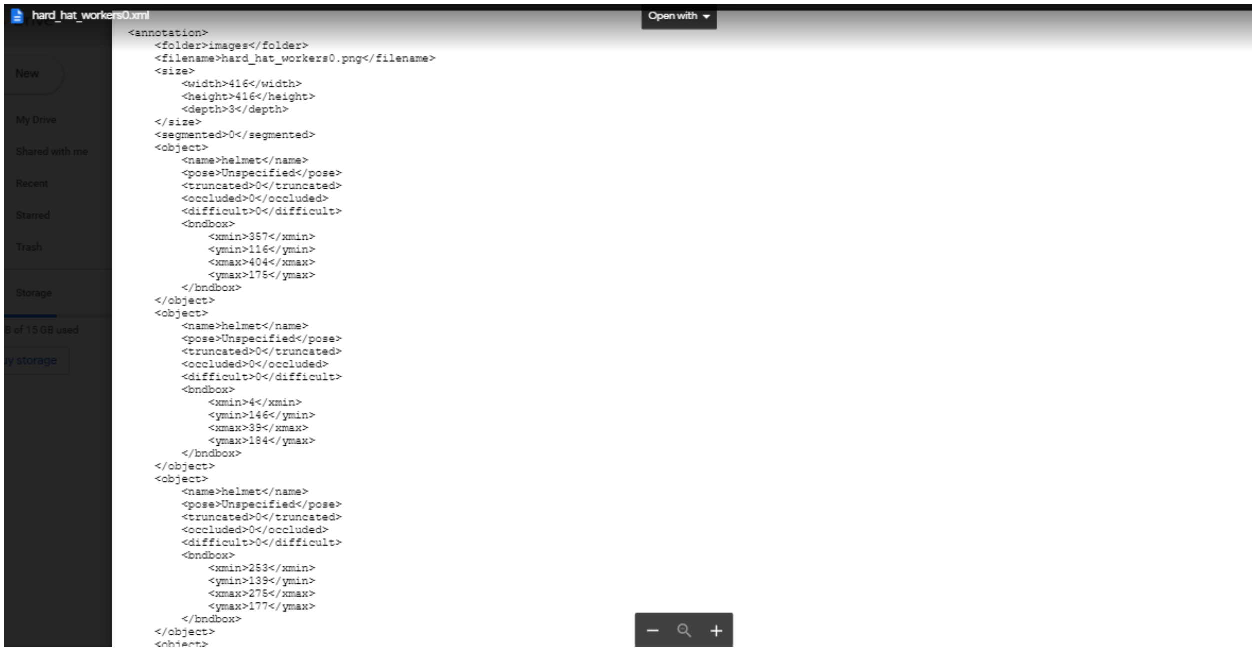


Figure 3.3 : Dataset

There are annotations available for each and every image. Annotations are in the PASCAL VOC format, saved as XML files.



```xml
<annotation>
    <folder>images</folder>
    <filename>hard_hat_workers0.png</filename>
    <size>
        <width>416</width>
        <height>416</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>helmet</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <occluded>0</occluded>
        <difficult>0</difficult>
        <bndbox>
            <xmin>357</xmin>
            <ymin>116</ymin>
            <xmax>404</xmax>
            <ymax>175</ymax>
        </bndbox>
    </object>
    <object>
        <name>helmet</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <occluded>0</occluded>
        <difficult>0</difficult>
        <bndbox>
            <xmin>4</xmin>
            <ymin>146</ymin>
            <xmax>39</xmax>
            <ymax>184</ymax>
        </bndbox>
    </object>
    <object>
        <name>helmet</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <occluded>0</occluded>
        <difficult>0</difficult>
        <bndbox>
            <xmin>253</xmin>
            <ymin>139</ymin>
            <xmax>275</xmax>
            <ymax>177</ymax>
        </bndbox>
    </object>
    <object>
```

Figure 3.4: Annotations

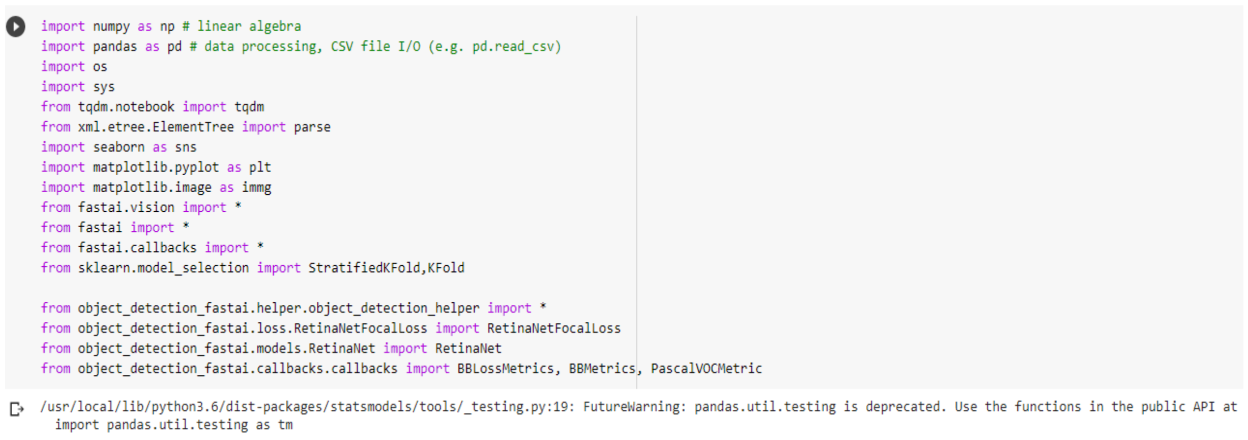
The above figure 3.4 shows the annotations file created for one of the images among the dataset comprising of 5000 images. Similarly each image consists of each annotation file saved in XML format.

3.3 DATA PRE-PROCESSING

Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behavior or trends, and is likely to contain many errors. Data pre-processing is a proven method of resolving such issues. Steps performed in pre-processing are:

STEP 1: IMPORTING THE LIBRARIES

As shown in Fig 3.5 we import libraries in Python using import keyword and these are the most popular libraries which any Data Scientist uses.



```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import os
import sys
from tqdm.notebook import tqdm
from xml.etree.ElementTree import parse
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.image as immg
from fastai.vision import *
from fastai import *
from fastai.callbacks import *
from sklearn.model_selection import StratifiedKFold,KFold

from object_detection_fastai.helper.object_detection_helper import *
from object_detection_fastai.loss.RetinaNetFocalLoss import RetinaNetFocalLoss
from object_detection_fastai.models.RetinaNet import RetinaNet
from object_detection_fastai.callbacks.callbacks import BBLossMetrics, BBMetrics, PascalVOCMetric
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprecated. Use the functions in the public API at
  import pandas.util.testing as tm
```

Figure 3.5: Importing libraries

NumPy:

NumPy is the fundamental package for scientific computing with Python. As it is used to divide the given image from construction site into n dimensional objects such that it can be easily compared with the dataset .It contains another things such as:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions.
- Useful Linear Algebra, Fourier transform and random number capabilities.

Pandas:

Pandas is for data manipulation and analysis. The main purpose that we are using the pandas in neural network because it analyse the given input data and it will recognize that the given thing is an object and it will further proceed, whereas Pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. Pandas is a Num FOCUS sponsored project. This will help ensure the success of development of pandas as a world-class open-source project, and makes it possible to donate to the project.

Matplotlib:

Matplotlib is a Python 2D plotting library which it is used to plot given image in number of small images and it will check with the dataset and it will produce publication quality figures in a variety of hard copy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits.

Sklearn:

In python, sklearn is a machine learning package which include a lot of ML algorithms. Here, we are using some of its modules like train_test_split, and accuracy_score.

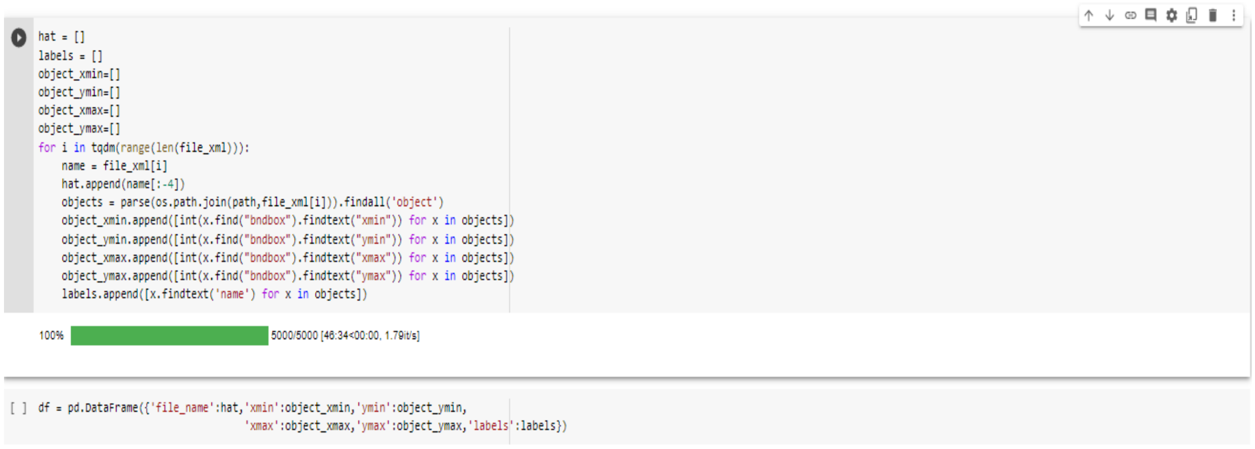
Fastai:

Fastai is a deep learning library which provides practitioners with high-level components that can quickly and easily provide state-of-the-art results in standard deep learning domains, and provides researchers with low-level components that can be mixed and matched to build new approaches. It aims to do both things without substantial compromises in ease of use, flexibility, or performance. This is possible thanks to a carefully layered architecture, which expresses common underlying patterns of many deep learning and data processing techniques in terms of decoupled abstractions. These abstractions can be expressed concisely and clearly by leveraging the dynamism of the underlying Python language and the flexibility of the PyTorch library. fastai includes: a new type dispatch system for Python along with a semantic type hierarchy for tensors; a GPU-optimized computer vision library which can be extended in pure Python.

Seaborn:

Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures. Seaborn helps you explore and understand your data. Its plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots. Its dataset-oriented, declarative API lets you focus on what the different elements of your plots mean, rather than on the details of how to draw them.

STEP 2: PROCESSING XML FILES



```
hat = []
labels = []
object_xmin=[]
object_ymin=[]
object_xmax=[]
object_ymax=[]
for i in tqdm(range(len(file_xml))):
    name = file_xml[i]
    hat.append(name[:-4])
    objects = parse(os.path.join(path,file_xml[i])).findall('object')
    object_xmin.append([int(x.find("bndbox").findtext("xmin")) for x in objects])
    object_ymin.append([int(x.find("bndbox").findtext("ymin")) for x in objects])
    object_xmax.append([int(x.find("bndbox").findtext("xmax")) for x in objects])
    object_ymax.append([int(x.find("bndbox").findtext("ymax")) for x in objects])
    labels.append([x.findtext('name') for x in objects])
```

```
100% 5000/5000 [46:34<00:00, 1.79it/s]
```

```
[ ] df = pd.DataFrame({'file_name':hat,'xmin':object_xmin,'ymin':object_ymin,
                                        'xmax':object_xmax,'ymax':object_ymax,'labels':labels})
```

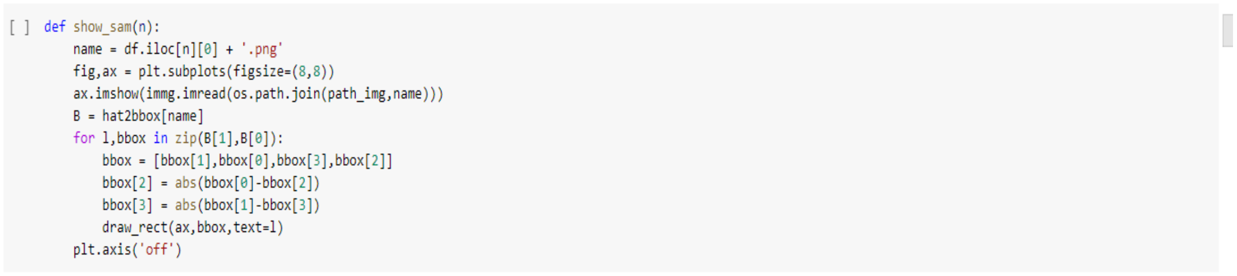
Figure 3.6 : XML files processing

Before starting the training of the model, we need to make sure that the data is preprocessed correctly so that the training of the model can be done without any interruptions.

Here we need to find out the exact dimensions of the bounding boxes from the annotations and the images of the dataset so that our model can know what to identify from the image and what to neglect.

STEP 3: Findings and Suggestions

The concept of findings and suggestions is important to understand in order to successfully find image data. If the suggestion values are not handled properly by the researcher, then he/she may end up drawing an inaccurate inference about the data. Due to improper handling, the result obtained by the researcher will differ from ones where the suggestions are present.



```
[ ]  def show_sam(n):
         name = df.iloc[n][0] + '.png'
         fig,ax = plt.subplots(figsize=(8,8))
         ax.imshow(immg.imread(os.path.join(path_img,name)))
         B = hat2bbox[name]
         for l,bbox in zip(B[1],B[0]):
             bbox = [bbox[1],bbox[0],bbox[3],bbox[2]]
             bbox[2] = abs(bbox[0]-bbox[2])
             bbox[3] = abs(bbox[1]-bbox[3])
             draw_rect(ax,bbox,text=l)
         plt.axis('off')
```

Figure 3.7 : displaying sample image with bounding boxes

```
show_sam(random.randint(2,4800))
```

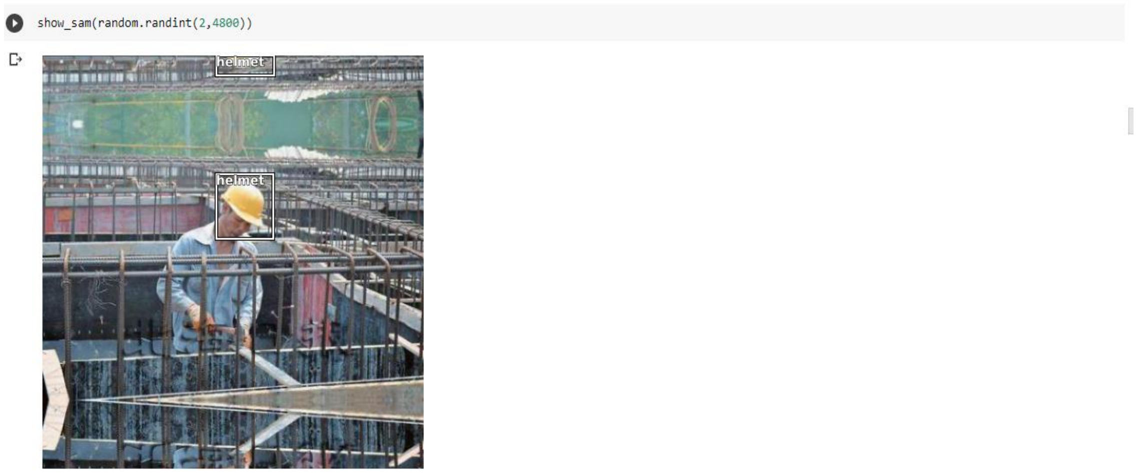
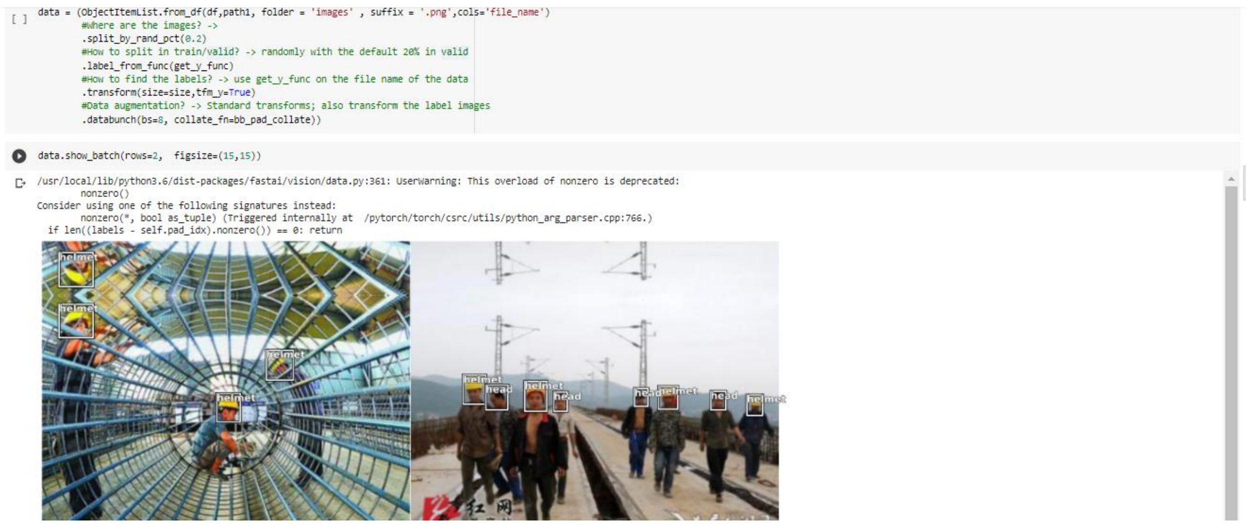


Figure 3.8 Image Acquisition

As shown in Figure 3.8, we can observe that the image is being displayed and it is having bounding boxes accurately noticing the person with the helmet. Here a random image is being visualized just to cross evaluate the results, later we have to get bounding boxes for each image available in the dataset.

STEP 4: SPLITTING OF DATASET AND CREATING DATABUNCHES

The concept of splitting of dataset is to train the model using some data and also to test the developed model using the remaining data so that we can find the accuracy of the developed model and also we can validate the developed model just to know that the developed model is capable of predicting the required results or not.



```
data = (ObjectItemList.from_df(df,path1, folder = 'images' , suffix = '.png',cols='file_name')
        #Where are the images? ->
        .split_by_rand_pct(0.2)
        #How to split in train/valid? -> randomly with the default 20% in valid
        .label_from_func(get_y_func)
        #How to find the labels? -> use get_y_func on the file name of the data
        .transform(size=size,tfm_y=True)
        #Data augmentation? -> Standard transforms; also transform the label images
        .databunch(bs=8, collate_fn=bb_pad_collate))
```

```
data.show_batch(rows=2,  figsize=(15,15))
```

```
/usr/local/lib/python3.6/dist-packages/fastai/vision/data.py:361: UserWarning: This overload of nonzero is deprecated:
        nonzero()
Consider using one of the following signatures instead:
        nonzero(*, bool as_tuple) (Triggered internally at  /pytorch/torch/csrc/utils/python_arg_parser.cpp:766.)
  if len((labels - self.pad_idx).nonzero()) == 0: return
```

Figure 3.9 Displaying data bunches

3.4 MODEL

3.4.1 Convolutional neural network

A convolutional neural network is a neural network architecture pioneered by LeCun .and later popularize by Krizhevsky. With the introduction of the Alex Net. A typical CNN consists of convolutional layers and pooling layers followed by a fully connected neural network. The convolutional layers and the pooling layers are supposed to learn how to extract relevant, locally distortion invariant, features from the input, and the fully connected neural network is supposed to learn how to classify these features. The features learned by the convolutional layers could be, e.g., edges of objects in an image.

3.4.2 RESNET-18

ResNet-18 is a convolutional neural network that is 18 layers deep. You can load a pretrained version of the network trained on more than a million images from the ImageNet database [1]. The pretrained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images. The network has an image input size of 224-by-224.

Was ResNet Successful?

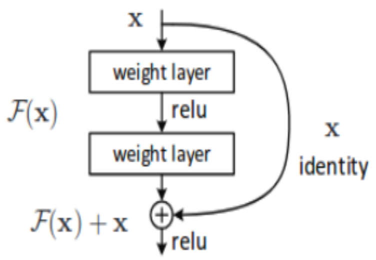
Won 1st place in the ILSVRC 2015 classification competition with top-5 error rate of 3.57% (An ensemble model).

Won the 1st place in ILSVRC and COCO 2015 competition in ImageNet Detection, ImageNet localization, Coco detection and Coco segmentation.

Replacing VGG-16 layers in Faster R-CNN with ResNet-101. They observed a relative-improvements of 28%.

Efficiently trained networks with 100 layers and 1000 layers also.

There are two kinds of residual connections:

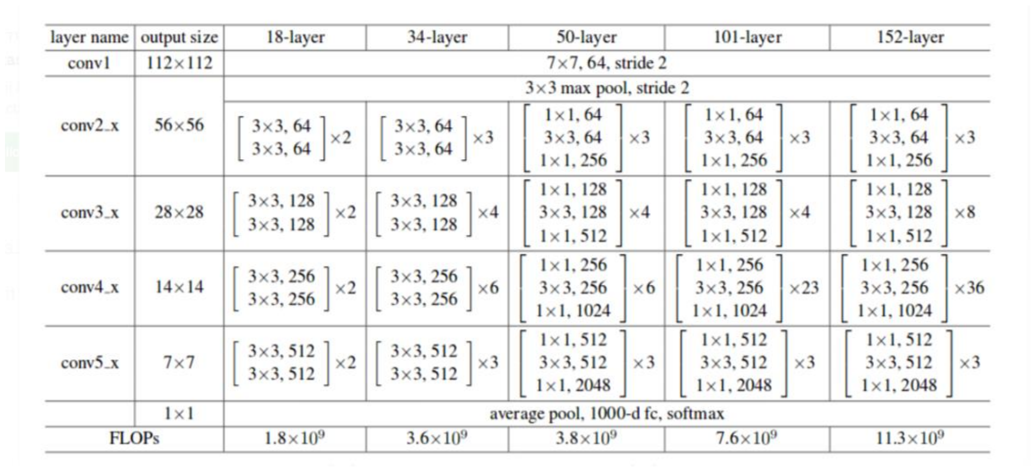


x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 3.10 Residual block

1. The identity shortcuts (x) can be directly used when the input and output are of the same dimensions.
2. When the dimensions change, A) The shortcut still performs identity mapping, with extra zero entries padded with the increased dimension. B) The projection shortcut is used to match the dimension (done by 1*1 conv).

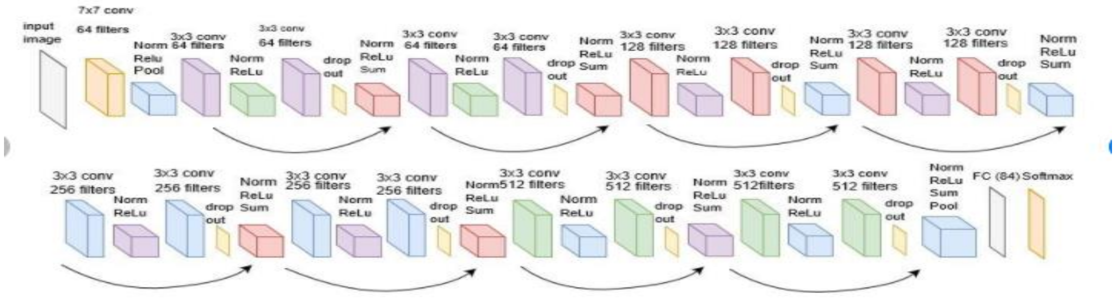
Each ResNet block is 2 layer deep in ResNet-18 network.



layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\left[\begin{array}{c}3\times3, 64\\3\times3, 64\end{array}\right]\times2$	$\left[\begin{array}{c}3\times3, 64\\3\times3, 64\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 64\\3\times3, 64\\1\times1, 256\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 64\\3\times3, 64\\1\times1, 256\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 64\\3\times3, 64\\1\times1, 256\end{array}\right]\times3$
conv3_x	28×28	$\left[\begin{array}{c}3\times3, 128\\3\times3, 128\end{array}\right]\times2$	$\left[\begin{array}{c}3\times3, 128\\3\times3, 128\end{array}\right]\times4$	$\left[\begin{array}{c}1\times1, 128\\3\times3, 128\\1\times1, 512\end{array}\right]\times4$	$\left[\begin{array}{c}1\times1, 128\\3\times3, 128\\1\times1, 512\end{array}\right]\times4$	$\left[\begin{array}{c}1\times1, 128\\3\times3, 128\\1\times1, 512\end{array}\right]\times8$
conv4_x	14×14	$\left[\begin{array}{c}3\times3, 256\\3\times3, 256\end{array}\right]\times2$	$\left[\begin{array}{c}3\times3, 256\\3\times3, 256\end{array}\right]\times6$	$\left[\begin{array}{c}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{array}\right]\times6$	$\left[\begin{array}{c}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{array}\right]\times23$	$\left[\begin{array}{c}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{array}\right]\times36$
conv5_x	7×7	$\left[\begin{array}{c}3\times3, 512\\3\times3, 512\end{array}\right]\times2$	$\left[\begin{array}{c}3\times3, 512\\3\times3, 512\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{array}\right]\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Table 3.2: ResNet networks Comparision

Example:



Proposed Modified ResNet-18 architecture for Bangla HCR. In the diagram, conv stands for Convolutional layer, Pool stands for MaxPool layer, batch norm stand for batch normalization, Relu stands for rectified linear unit activation layer, Sum stands for the addition in ResNet, and FC stand for fully connected hidden layers. In this architecture, we have eight ResNet modules which are modified by adding a dropout layer after the second convolutional layers.

Figure 3.11 Example of resnet-18 architecture

A residual neural network (ResNet) is an artificial neural network (ANN) of a kind that builds on constructs known from pyramidal cells in the cerebral cortex. Residual neural networks do this by utilizing skip connections, or shortcuts to jump over some layers. Typical ResNet models are implemented with double- or triple- layer skips that contain nonlinearities (ReLU) and batch normalization in between. An additional weight matrix may be used to learn the skip weights; these models are known as HighwayNets. Models with several parallel skips are referred to as DenseNets. In the context of residual neural networks, a non-residual network may be described as a plain network.

One motivation for skipping over layers is to avoid the problem of vanishing gradients, by reusing activations from a previous layer until the adjacent layer learns its weights. During training, the weights adapt to mute the upstream layer, and amplify the previously-skipped layer. In the simplest case, only the weights for the adjacent layer's connection are adapted, with no explicit weights for the upstream layer. This works best when a single nonlinear layer is stepped over, or when the intermediate layers are all linear. If not, then an explicit weight matrix should be learned for the skipped connection (a *HighwayNet* should be used).

Skipping effectively simplifies the network, using fewer layers in the initial training stages. This speeds learning by reducing the impact of vanishing gradients, as there are fewer layers to propagate through. The network then gradually restores the skipped layers as it learns the feature space. Towards the end of training, when all layers are expanded, it stays closer to the manifold and thus learns faster. A neural network without residual parts explores more of the feature space. This makes it more vulnerable to perturbations that cause it to leave the manifold, and necessitates extra training data to recover.

3.5 UML DIAGRAMS

Unified Modelling Language is a standard language for writing software blueprints. It can be used to visualize, specify, construct, and document the artifacts of a software-intensive system. Modelling is a proven and well-accepted engineering technique.

3.5.1 Data Flow Diagram

As shown in Fig 3.14, Data Flow Diagram is a graphical representation of the flow of the data through an information system modelling its process aspects. Firstly, read the dataset and preprocess the dataset. Next, Train learning model over the dataset and the test the model with the test dataset note down the accuracy given by that model iterate the above process in order to get the best model which gives maximum accuracy. It does not show information about process timing or whether processes will operate in sequence or in parallel, unlike a traditional structured flowchart which focuses on control flow.

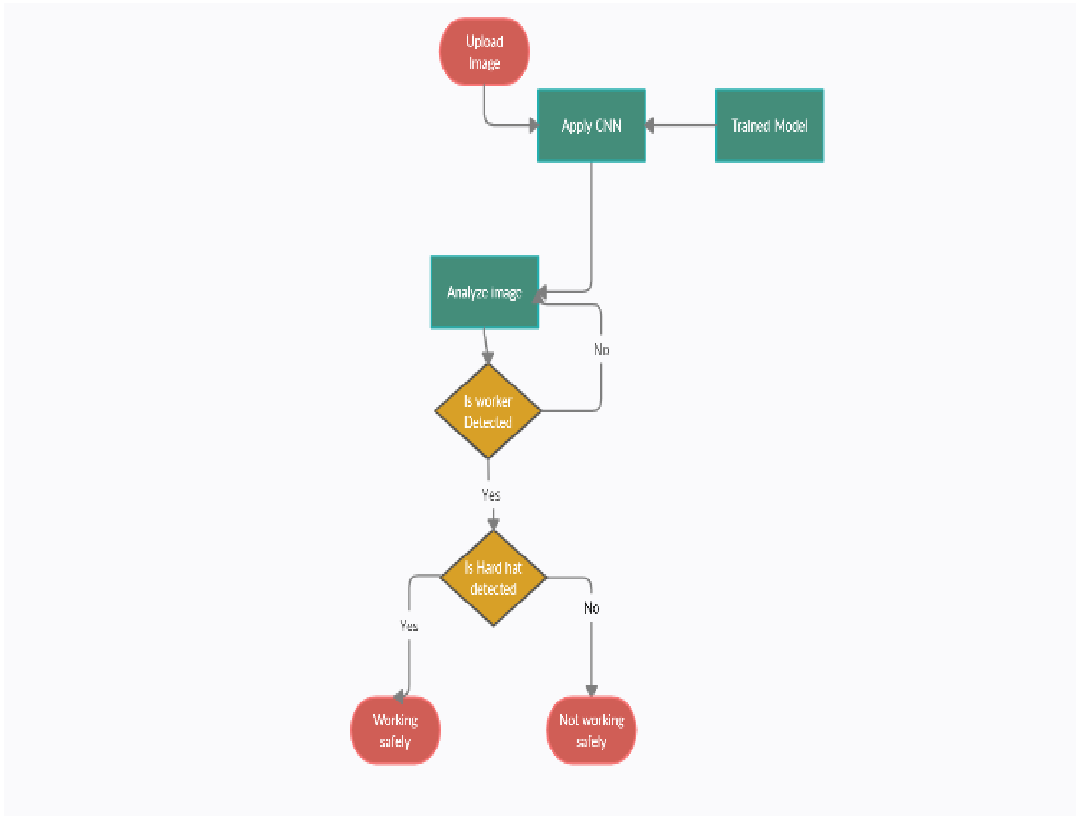


Figure 3.12: Data flow Diagram for hard hat detection

3.5.2 Use -Case Diagrams

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

As shown in Figure the role of the system administrator is to give input. The data set used should allow operations like cleaning the blurred image, updating, adding and deleting. The other actor is the user and system who deploys a model based on the maximum accuracy for an application. The administrator divides the data set into training and testing phases and modelling is performed iteratively in order to get the best model for deployment.

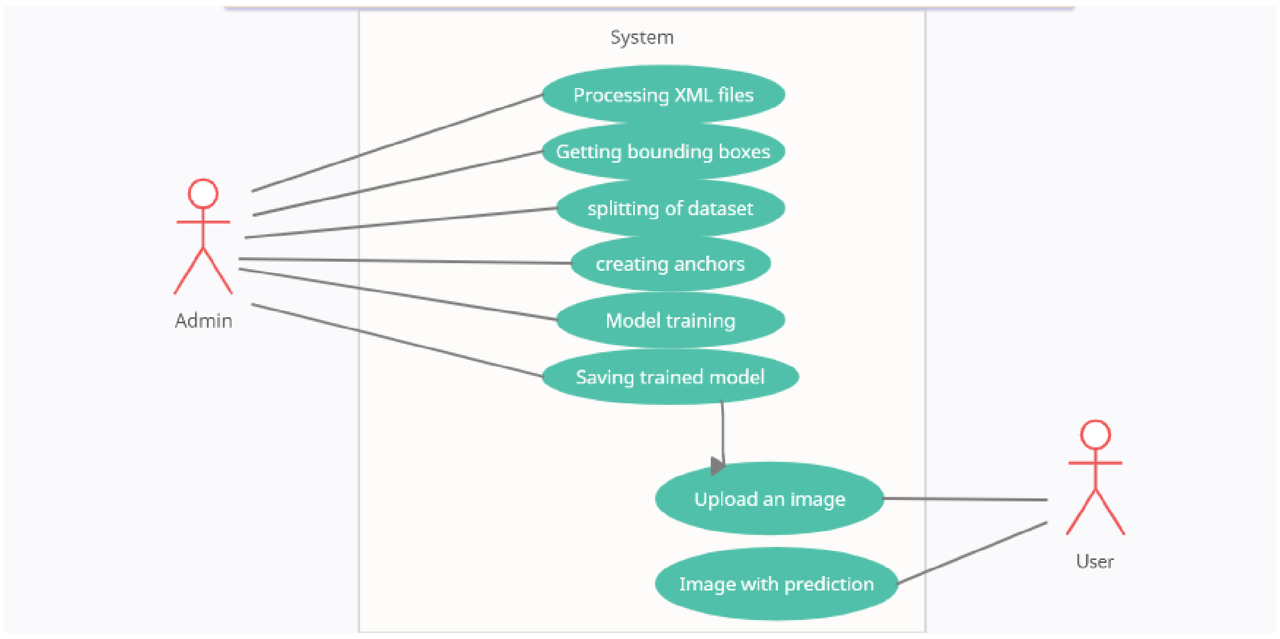


Figure 3.13: Use Case Diagram

3.5.3 Class Diagrams

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. As Shown in Figure Each class contain a set of attributes and functions.

As shown in Figure 3.16, there are four main classes to implement this algorithm. Out of which User class and Designer class would provide the access to the respective person.

Dataset class will collect the input and allows us to modify, add or delete data inputs. It takes the raw image as input it will be splitted into parts using grayscale. It also passes the input information to the modelling class that further processes the data implementing the algorithm over the data. The modelling class holds training and testing phases that will help the output prediction based on the given input.

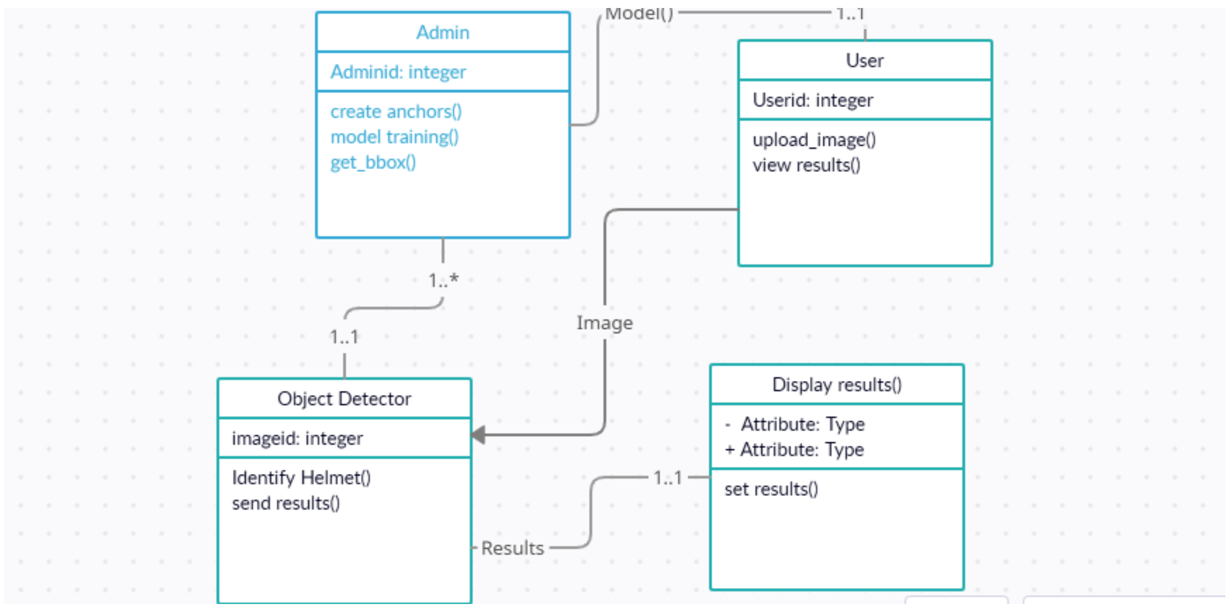


Figure 3.14: Class Diagram

3.5.4 Sequence Diagram

Sequence Diagram emphasizes the time ordering of messages. A sequence diagram shows a set of objects and the messages sent and received by those objects. The objects are typically named or anonymous instances of classes. Sequence diagrams are used to model the dynamic aspects of a system.

As shown in Figure 3.17 the user imports the required libraries and then imports the data set. If there are any noises in the data, clean the data using data cleaning operations such as cleaning the blurred parts and transform into clear image. Transform data into appropriate form for Modelling. Train an appropriate model over the data set using sklearn library used and deploy the best model based on the accuracy that a specific model provides. Finally, visualize the predictions using appropriate libraries such as matplotlib and send visualizations to the user.

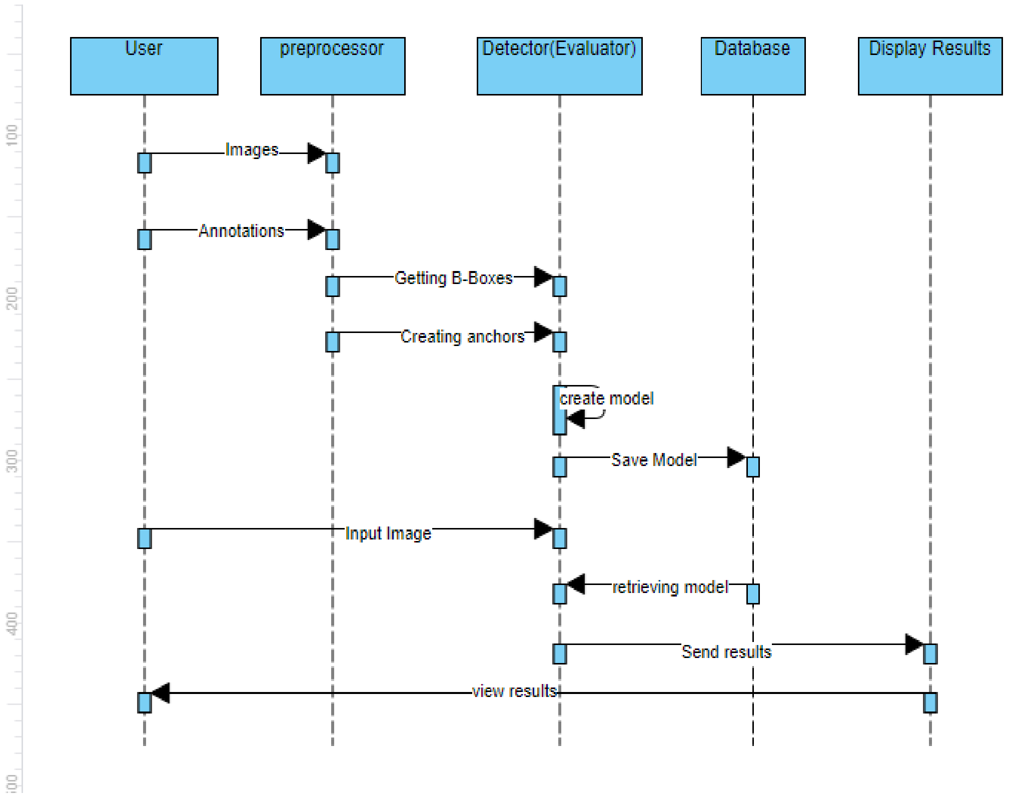


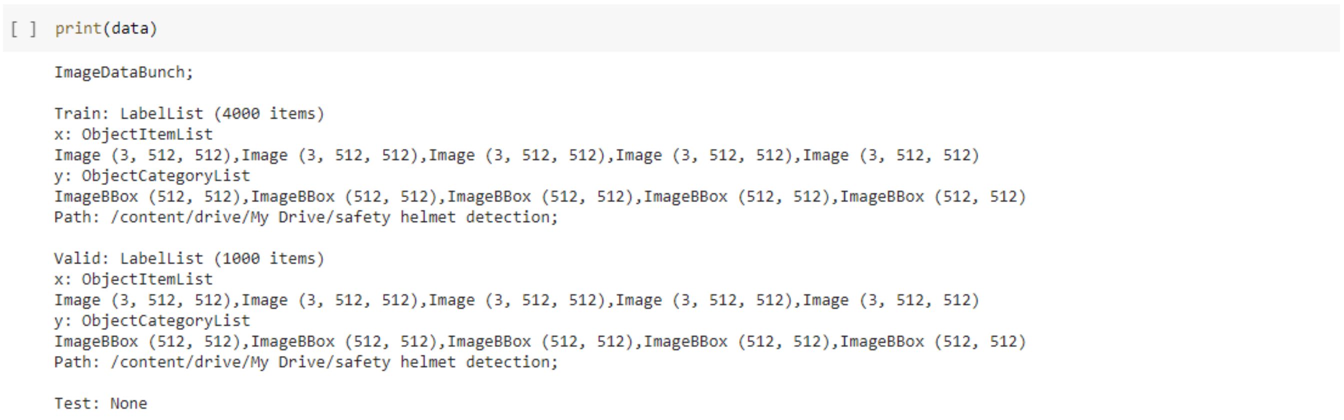
Figure 3.15: Sequence Diagram

4. TESTING AND RESULTS

4.1 TESTING

Testing is a very important module in the software development to verify, validate and provide quality and service for different components of software. It is used to minimize the risks by efficient use of resources in the development life cycle. This module can be employed at any point of the development process. It is efficient for the testing phase to be implemented at initial level to lower down the risks of defects and failures. Software is tested and implemented in various conditions and environments to examine different aspect of software. There exist various organizations for testing which is based on the type of software developed. This testing phase helps each and every module to work effectively and generate the optimum results.

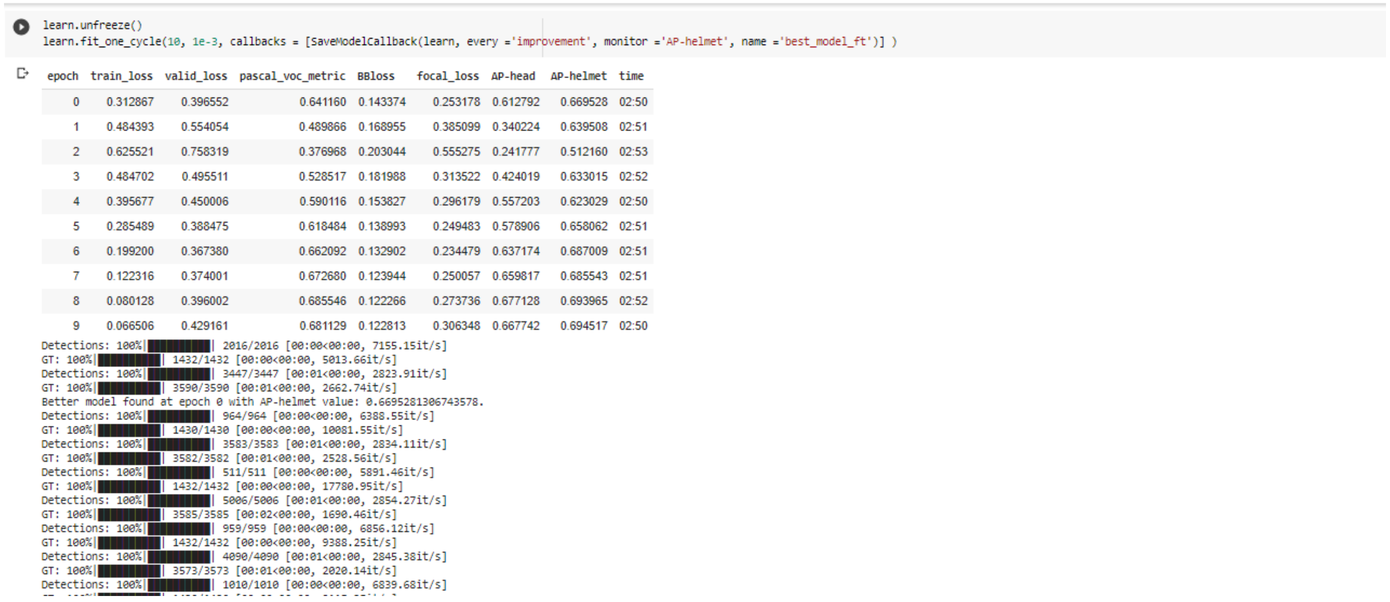
In this project, considering 4000 images for training purpose and remaining 1000 images for validation purpose. The number of images and size of the image and also the image bounding boxes considered were clearly shown in the below figure.



```
[ ]  print(data)

     ImageDataBunch;

     Train: LabelList (4000 items)
     x: ObjectItemList
     Image (3, 512, 512),Image (3, 512, 512),Image (3, 512, 512),Image (3, 512, 512),Image (3, 512, 512)
     y: ObjectCategoryList
     ImageBBox (512, 512),ImageBBox (512, 512),ImageBBox (512, 512),ImageBBox (512, 512),ImageBBox (512, 512)
     Path: /content/drive/My Drive/safety helmet detection;

     Valid: LabelList (1000 items)
     x: ObjectItemList
     Image (3, 512, 512),Image (3, 512, 512),Image (3, 512, 512),Image (3, 512, 512),Image (3, 512, 512)
     y: ObjectCategoryList
     ImageBBox (512, 512),ImageBBox (512, 512),ImageBBox (512, 512),ImageBBox (512, 512),ImageBBox (512, 512)
     Path: /content/drive/My Drive/safety helmet detection;

     Test: None
```

Figure 4.1 Displaying data label list

4.2 RESULT ANALYSIS:



```
learn.unfreeze()
learn.fit_one_cycle(10, 1e-3, callbacks = [SaveModelCallback(learn, every ='improvement', monitor ='AP-helmet', name ='best_model_ft')] )
```

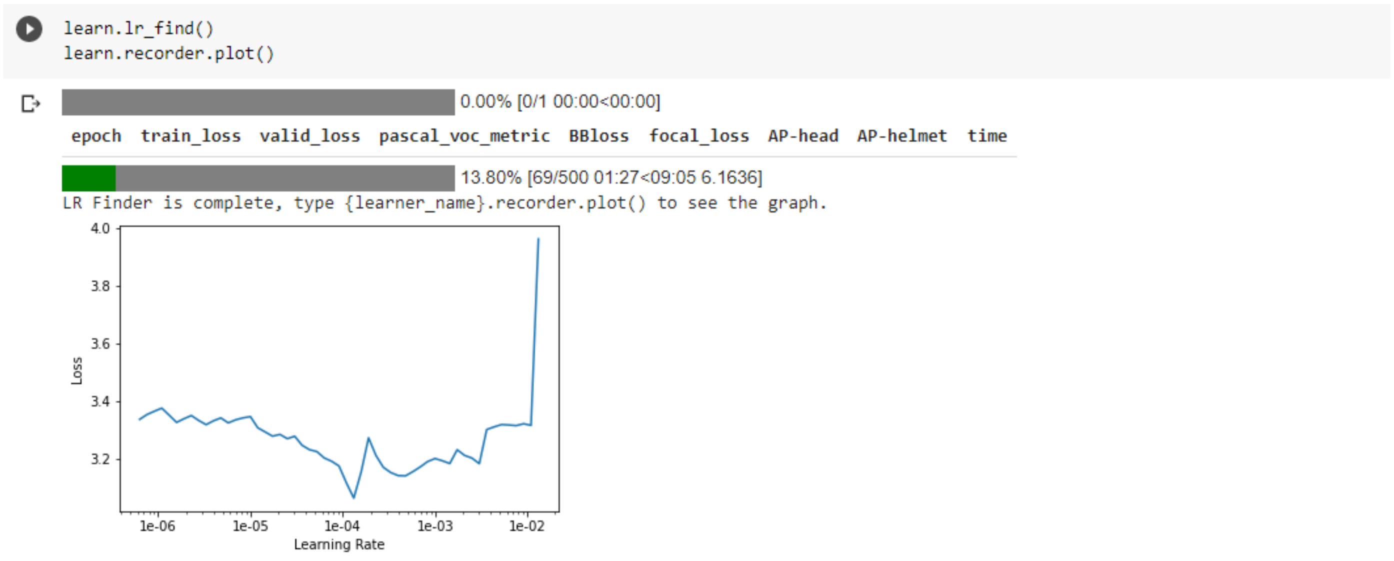
epoch	train_loss	valid_loss	pascal_voc_metric	BBloss	focal_loss	AP-head	AP-helmet	time
0	0.312867	0.396552	0.641160	0.143374	0.253178	0.612792	0.669528	02:50
1	0.484393	0.554054	0.489866	0.168955	0.385099	0.340224	0.639508	02:51
2	0.625521	0.758319	0.376968	0.203044	0.555275	0.241777	0.512160	02:53
3	0.484702	0.495511	0.528517	0.181988	0.313522	0.424019	0.633015	02:52
4	0.395677	0.450006	0.590116	0.153827	0.296179	0.557203	0.623029	02:50
5	0.285489	0.388475	0.618484	0.138993	0.249483	0.578906	0.658062	02:51
6	0.199200	0.367380	0.662092	0.132902	0.234479	0.637174	0.687009	02:51
7	0.122316	0.374001	0.672680	0.123944	0.250057	0.659817	0.685543	02:51
8	0.080128	0.396002	0.685546	0.122266	0.273736	0.677128	0.693965	02:52
9	0.066506	0.429161	0.681129	0.122813	0.306348	0.667742	0.694517	02:50

```
Detections: 100%|██████████| 2016/2016 [00:00<00:00, 7155.15it/s]
GT: 100%|██████████| 1432/1432 [00:00<00:00, 5013.66it/s]
Detections: 100%|██████████| 3447/3447 [00:01<00:00, 2823.91it/s]
GT: 100%|██████████| 3590/3590 [00:01<00:00, 2662.74it/s]
Better model found at epoch 0 with AP-helmet value: 0.6695281306743578.
Detections: 100%|██████████| 964/964 [00:00<00:00, 6388.55it/s]
GT: 100%|██████████| 1430/1430 [00:00<00:00, 10081.55it/s]
Detections: 100%|██████████| 3583/3583 [00:01<00:00, 2834.11it/s]
GT: 100%|██████████| 3582/3582 [00:01<00:00, 2528.56it/s]
Detections: 100%|██████████| 511/511 [00:00<00:00, 5891.46it/s]
GT: 100%|██████████| 1432/1432 [00:00<00:00, 17780.95it/s]
Detections: 100%|██████████| 5006/5006 [00:01<00:00, 2854.27it/s]
GT: 100%|██████████| 3585/3585 [00:02<00:00, 1690.46it/s]
Detections: 100%|██████████| 959/959 [00:00<00:00, 6856.12it/s]
GT: 100%|██████████| 1432/1432 [00:00<00:00, 9388.25it/s]
Detections: 100%|██████████| 4090/4090 [00:01<00:00, 2845.38it/s]
GT: 100%|██████████| 3573/3573 [00:01<00:00, 2020.14it/s]
Detections: 100%|██████████| 1010/1010 [00:00<00:00, 6839.68it/s]
```

Figure 4.2 Training the data

As this project is about detection of hard hats or safety helmets from the images collected from construction site, the main concentration is on the accuracy of the class helmet among the three classes namely, head, helmet, person. So based on the accuracy of the helmet from the image the best model has been selected from the 10 epochs. It is clear from the above figure that the accuracy of detecting helmet is more in the epoch 9 compared to all the 10 epochs.

As the number of epochs increases, the tendency in increasing the accuracy also increases. Basically I have trained it up to 13 epochs with 3epochs initially and then freezing the model to visualize and analyze the results obtained and then continued the training process for the next 10 epochs. Keeping the time factor in mind we can increase or decrease the number of epochs accordingly.



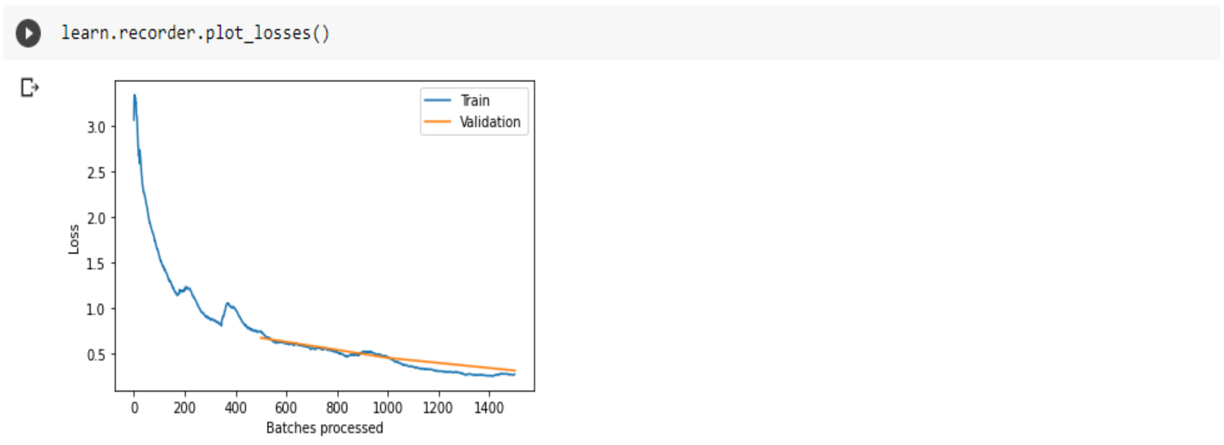
```
learn.lr_find()
learn.recorder.plot()
```

```
0.00% [0/1 00:00<00:00]
epoch  train_loss  valid_loss  pascal_voc_metric  BBloss  focal_loss  AP-head  AP-helmet  time
13.80% [69/500 01:27<09:05 6.1636]
LR Finder is complete, type {learner_name}.recorder.plot() to see the graph.
```

Figure 4.3 Learning rate Vs Loss

After training the model for 3 epochs, just to crosscheck whether we are going in a right path or not, it is advisable to visualize and analyze what the model is learning. So the above figure shows the plot graph plotted between the learning rate and the loss, where learning rate is considered as X axis and the loss is marked on the Y axis.

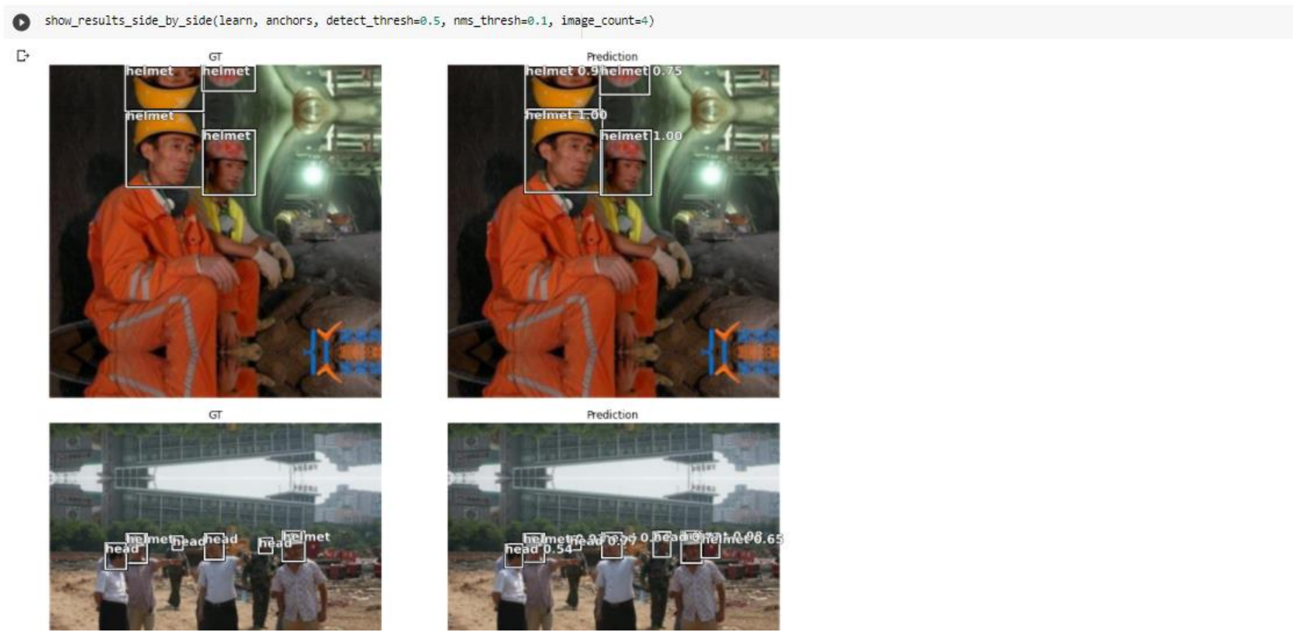
After completion of training the model with sufficient number of epochs and identified the best model among the various models by relying on the accuracy. We can visualize the loss occurred among the various batches in train and validation sets respectively which is visualized in the below figure.



```
learn.recorder.plot_losses()
```

Figure 4.4 Batches Processed Vs Loss

Displaying the predicted values and ground truth values side by side as shown in the below figure considering the threshold value ranges from 0.1 and 0.5.



```
show_results_side_by_side(learn, anchors, detect_thresh=0.5, nms_thresh=0.1, image_count=4)
```

Figure 4.5 Output: Ground truth and predicted

5. CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

The proposed method for detecting the wearing of hard hats by the workers at construction site is based on the convolutional neural networks. The model uses the RESNET-18 algorithm to detect the helmets. Dataset of 5000 images and 5000 annotations corresponding to the respective images containing various helmets is taken in which it is further classified into three classes namely head, person, helmet to train and test the model. The pytorch framework is chosen to train the model. The experiment results demonstrate that the method can be used to detect the safety helmets worn by the construction workers at the construction site. The presented method offers an alternative solution to detect the safety helmets and improve the safety management of the construction workers at the construction site.

The development of this program will help to save time, cost, and the life of workers at the site; however, at this stage, it will not completely reduce construction injuries and fatalities. Other factors, such as workers' behaviour, types of workers, types of hazardous work, and the level of skilled and trained workers, all influence the amount of construction injuries and fatalities [25, 26]. Therefore, it is necessary that safety engineers at construction sites conduct in-depth analyses, including risk analysis, to reduce the amount of construction accidents. The tool developed from this research will help safety engineers to achieve zero-fatality in construction sites.

5.2 Future Scope

An extension of the program is to detect whether the workers are wearing their vests; a third extension involves adding a program to detect a worker based on the outline of a worker and typical leg movements while walking. Additionally, the use of database management for all the workers such that the program can record all the workers' histories regarding safety-rule violations is being explored. Further study will continue until the program is able to detect effectively all the PPEs (vests, boots, globes, etc.). It will be easier to detect vests when using this program than to detect boots and gloves because the cameras presently cannot correctly capture the shape of the boots and gloves because of their small size compared to vests. In future, we can include some IOT based implementation to this model and add some sort of alarms that honks when a worker with no helmet is found in and around the construction site.

BIBILOGRAPHY

[1] K. Shrestha, P. P. Sherestha, D. Bajracharya and E. A. Yfantis, Hard-Hat Detection for Construction Safety Visualization, J. of Const. Eng. 2015 1-8.

[2] Health and Safety Executive, Health and safety in construction sector in Great Britain, 2014/15. Available at: http://www.hse.gov.uk/statistics/industry/construction/construction.pdf

[3] Occupational Safety and Health Administration, Module 13—Personal protective equipment. Available at: https://www.osha.gov/dte/

4] Y. S. Kim, J. H. Lee, H. S. Yoo, J. B. Lee and U. S. Jung, A performance evaluation of a Stewart platform based Hume concrete pipe manipulator, Auto. in Const. 18 (2009) 665-676.

[5] T. Yoshida and S. Chae, Application of RFID technology to prevention of collision accident with heavy equipment, Auto. in Const. 19 (2010) 368-374.

[6] L. Ding, H. L. Yu, C. Zhou, X. Wu and M. H. Yu, Safety risk identification system for metro construction on the basis of construction drawings, Auto. in Const. 27 (2012) 120-137.

[7] M. J. Skibniewski, Information technology applications in construction safety assurance, J. of Civ. Eng. and Manag. 20 (2014) 788-794.

[8] M. Memarzadeh, M. Golparvar-Fard and J. C. Niebles, Automated 2D detection of construction equipment and workers from site video streams using histograms of oriented gradients and colors, Auto. in Const. 32 (2013) 24-37.

[9] S. Du, M. Shehata and W. Badawy, Hardhat Detection in Video Sequences based on Face Features, Motion and Color Information, 3rd Int. Conf. on Comp. Res. and Dev. 2011.

[10] G. Gualdi, A. Prati, and R. Cucchiara, Contextual Information and covariance descriptors for people surveillance: an application for safety of construction workers, EURASIP J. on Im. and Vid. Proc. 9 (2011) 1-16.

[11] D. Bajracharya. Real time pattern recognition in digital video with application to safety in construction sites. University of Nevada, Las vegas. 2013.

[12] M.W. Park, N. Elsafty and Z. Zhu, Hardhat-Wearing Detection for Enhancing On-Site Safety of Construction Workers, J. of Const. Eng. and Manag. 141 (2015).

[13] A. H. M. Rubaiyat, T. T. Toma, M. Kalantari-Khandani, S. A.Rahman, L. Chen, Y. Ye, C.S. Pan. Automatic detection of helmet uses for construction safety. 2016 IEEE/WIC/ACM Int.l Conf. on Web Intel. Work. Nebraska. 2016. 135-142.

[14] K. Mikolajczyk and T. Tuytelaars, Local Image Features, in Encyclo-pedia of Biometrics, S. Li and A. Jain, Eds. Springer US, 2009, pp. 939-943.

[15] R. Brunelli, Template Matching Techniques in Computer Vision: Theory and Practice. Wiley 2009.

[16] J. Yu, J. Amores, N. Sebe, Q. Tian, A New Study on Distance Metrics as Similarity Measurement, IEEE Int. Conf. on Mult. and Expo, 2006

APPENDIX

```
# Installing required packages

!pip install object-detection-fastai

# Importing required libraries

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import os
import sys
from tqdm.notebook import tqdm
from xml.etree.ElementTree import parse
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.image as immg
from fastai.vision import *
from fastai import *
from fastai.callbacks import *
from sklearn.model_selection import StratifiedKFold,KFold

from object_detection_fastai.helper.object_detection_helper import *
from object_detection_fastai.loss.RetinaNetFocalLoss import RetinaNetFocalLoss
from object_detection_fastai.models.RetinaNet import RetinaNet
from object_detection_fastai.callbacks.callbacks import BBLossMetrics, BBMetrics, PascalVOCMetri
c


# Setting path to the dataset

path = '/content/drive/My Drive/safety helmet detection/annotations'
path_img = '/content/drive/My Drive/safety helmet detection/images'


# appending all annotated files from dataset

file_xml = []
for file in os.listdir(path):
    if '.xml' in file:
        file_xml.append(file)


# Processing XML files for BBox's and Labels
```

```
hat = []
labels = []
object_xmin=[]
object_ymin=[]
object_xmax=[]
object_ymax=[]
for i in tqdm(range(len(file_xml))):
    name = file_xml[i]
    hat.append(name[:-4])
    objects = parse(os.path.join(path,file_xml[i])).findall('object')
    object_xmin.append([int(x.find("bndbox").findtext("xmin")) for x in objects])
    object_ymin.append([int(x.find("bndbox").findtext("ymin")) for x in objects])
    object_xmax.append([int(x.find("bndbox").findtext("xmax")) for x in objects])
    object_ymax.append([int(x.find("bndbox").findtext("ymax")) for x in objects])
    labels.append([x.findtext('name') for x in objects])


df = pd.DataFrame({'file_name':hat,'xmin':object_xmin,'ymin':object_ymin,
                                   'xmax':object_xmax,'ymax':object_ymax,'labels':labels})


path1 = Path('/content/drive/My Drive/safety helmet detection')


# Function to get BBOXs

def image_lbl(df):
    hat2bbox = {}
    for i in tqdm(range(df.shape[0])):
        bbox = []
        lbl =[]
        title = []
        a = df.iloc[i][1:-1].values
        l = df.iloc[i][-1]
        for j in range(len(l)):
            bbx = [x[j] for x in a]
            if l[j]!='person':
                bbx = [bbx[1],bbx[0],bbx[3],bbx[2]]
                lbl.append(bbx)
                title.append(l[j])
        bbox.append(lbl)
        bbox.append(title)
        hat2bbox[df.iloc[i][0]+'.png'] = bbox
    return hat2bbox
```

```
hat2bbox = image_lbl(df)


# Function to see a sample image

def show_sam(n):
    name = df.iloc[n][0] + '.png'
    fig,ax = plt.subplots(figsize=(8,8))
    ax.imshow(immg.imread(os.path.join(path_img,name)))
    B = hat2bbox[name]
    for l,bbox in zip(B[1],B[0]):
        bbox = [bbox[1],bbox[0],bbox[3],bbox[2]]
        bbox[2] = abs(bbox[0]-bbox[2])
        bbox[3] = abs(bbox[1]-bbox[3])
        draw_rect(ax,bbox,text=l)
    plt.axis('off')


show_sam(random.randint(2,4800))


# Function to get BBOXs for each image

get_y_func = lambda o: hat2bbox[Path(o).name]


tfms = get_transforms()
size = 512


# Forming databunches

data = (ObjectItemList.from_df(df,path1, folder = 'images' , suffix = '.png',cols='file_name')
        #Where are the images? ->
        .split_by_rand_pct(0.2)
        #How to split in train/valid? -> randomly with the default 20% in valid
        .label_from_func(get_y_func)
        #How to find the labels? -> use get_y_func on the file name of the data
        .transform(size=size,tfm_y=True)
        #Data augmentation? -> Standard transforms; also transform the label images
        .databunch(bs=8, collate_fn=bb_pad_collate))


data.show_batch(rows=2,  figsize=(15,15))
```

```
len(data.train_ds),len(data.valid_ds),data.classes


print(data)


anchors = create_anchors(sizes=[(32,32)], ratios=[1], scales=[0.3, 0.6, 1.2, 2, 2.8, 3.4,])


fig,ax = plt.subplots(figsize=(8,8))
ax.imshow(image2np(data.valid_ds[0][0].data))

for i, bbox in enumerate(anchors[:6]):
    bb = bbox.numpy()
    x = (bb[0] + 1) * size / 2
    y = (bb[1] + 1) * size / 2
    w = bb[2] * size / 2
    h = bb[3] * size / 2

    rect = [x,y,w,h]
    draw_rect(ax,rect)


print(anchors)


n_classes = data.train_ds.c

crit = RetinaNetFocalLoss(anchors)

encoder = create_body(models.resnet18, True, -2)
model = RetinaNet(encoder, n_classes=data.train_ds.c, n_anchors=6, sizes=[32], chs=32, final_bias=-
4., n_conv=3)


voc = PascalVOCMetric(anchors, size, [i for i in data.train_ds.y.classes[1:]])
learn = Learner(data, model,
                loss_func=crit,
                callback_fns=[BBMetrics],
                metrics=[voc],
               model_dir='/content/drive/My Drive/safety helmet detection')


learn.split([model.encoder[6], model.c5top5])
learn.freeze_to(-2)
learn.lr_find()
learn.recorder.plot()


gc.collect()


# Training for 3 epochs

learn.fit_one_cycle(3, 1e-
3 , callbacks = [ SaveModelCallback(learn, every ='improvement', monitor = 'AP-
helmet', name = 'Best_model' ) ] )


learn.load('Best_model');
learn.export('/content/drive/My Drive/safety helmet detection/SafetyHelmet.pkl');


learn.recorder.plot_losses()


# Training the model for remaining epochs

learn.unfreeze()
learn.fit_one_cycle(10, 1e-
3, callbacks = [SaveModelCallback(learn, every ='improvement', monitor ='AP-
helmet', name ='Best_model_ft')] )


learn.load('Best_model_ft');
learn.export('/content/drive/My Drive/safety helmet detection/SafetyHelmet_ft.pkl');


# Displaying the results

show_results_side_by_side(learn, anchors, detect_thresh=0.5, nms_thresh=0.1, image_count=4)
```