Dissertation on

“Detection of Cyber Bullying in Images”

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technologyin
Computer Science & Engineering

UE17CS490B – Capstone Project Phase - 2

Submitted by:

Sruthi Madineni	**PES1201700051**
Meghana Nayak	**PES1201701339**
Rachana H S	**PES1201701726**

Under the guidance of

Prof. Aruna S
Assistant Professor
PES University

January - May 2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013) 100ft
Ring Road, Bengaluru – 560 085, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

100ft Ring Road, Bengaluru – 560 085, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled

'Detection of Cyber Bullying in Images'

is a bonafide work carried out by

Sruthi Madineni	**PES1201700051**
Meghana Nayak	**PES1201701339**
Rachana H S	**PES1201701726**

in partial fulfilment for the completion of seventh semester Capstone Project Phase - 2 (UE17CS490B) in the Program of Study - Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period Jan. 2021 – May. 2021. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 8th semester academic requirements in respect of project work.

Signature	Signature	Signature
Prof. Aruna S	Dr. ShylajaS S	Dr. B K Keshavan
Assistant Professor	Chairperson	Dean of Faculty

External Viva

Name of the Examiners	**Signature with Date**
1.______________________	______________________
2.______________________	______________________

DECLARATION

We hereby declare that the Capstone Project Phase-2 entitled **“Detection of Cyber Bullying in Images”** has been carried out by us under the guidance of Prof. Aruna S, Assistant Professor and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester January – May 2021. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

PES1201700051	**Sruthi Madineni**	[illegible]
PES1201701339	**Meghana Nayak**	[illegible]
PES1201701726	**Rachana H S**	Rachana. H.S

ACKNOWLEDGEMENT

I would like to express my gratitude to Prof. Aruna S, Assistant Professor, Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE17CS490B - Capstone Project Phase - 2.

I am grateful to the project coordinators, Prof. Silviya Nancy J and Prof. Sunitha R, for organizing, managing, and helping with the entire process.

I take this opportunity to thank Dr. Shylaja S S, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department. I would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this project could not have been completed without the continual support and encouragement I have received from my family and my friends.

ABSTRACT

Cyber bullying is a form of bullying which is administered using some electronic means. It has become very common with the advance of Internet and usage of social media sites. This can happen by various means. Posting rumors, sexual remarks, threats, disclosing a person's information without his consent being few among them.

Most of the previous studies are done on texts using some text classification methods. Recently there have been works even on images. Lot of researchers are working on coming with a better classification algorithm on cyberbullying detection on images.

The purpose of this project is to build a design methodology to detect cyberbullying by using Convolution Neural Network.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Cyber bullying is a form of bullying which is administered using some electronic means. It has become very common with the advance of Internet and usage of social media sites. This can happen by various means. Posting rumors, sexual remarks, threats, disclosing a person's information without his consent being few among them. This not only affect the victim but it also affects the people close to him. This might lead to a person's low self-esteem, increase suicidal thoughts, being scared, frustrated, angry or depressed.

Detecting cyber bullying and preventing it from affecting the masses has been the concerns of many.There have been many works on the same using different modern computer techniques.

Nowadays Cyberbullying is a common phenomenon in many of the social media sites like Instagram, Twitter, Facebook and many other. According to the statistics for the year 2018-2020,60% of the parents reported that their kids were bullied. There are also other platforms like mails, messaging apps and certain websites where they target certain individuals or even a group of individuals. It has become very difficult to stop cyber bullying because parents and the relatives of the victims are most of the times not aware about the same.

Bullying content can be in the form of texts, images and videos. Most of the previous studies are done on texts using some text classification methods. Recently there have been works even on images. Lot of researchers are working on coming with a better classification algorithm on cyberbullying detection on images.

The purpose of this project is to build a design methodology to detect cyberbullying by using Convolution Neural Networks.

CHAPTER2

PROBLEM STATEMENT

Cyberbullying is utilizing electronic means to embarrass, threaten, harass and targeting another person. Cyberbullying could be in the form of rude or aggressive texts, abusive or inappropriate images, inappropriate videos etc. Cyberbullying focuses on things like race, person's gender, sexual orientation, religion which violates the law. Apart from violating the law it also causes victim to be in constant state of fear/upset, it can also lead them to fall into depression, anxiety and other mood related disorders.
According to recent survey it has been found that overall 36.5% percent of people feel that they havebeen cyberbullied in their lifetime and 17.4% have reported it has happened at some point in last 30 days. 87% of young people have seen cyberbullying occurring online.

As mentioned above cyberbullying can also happen in the form of text and it has become serious problem among teenagers and adolescents. Works on detection of cyberbullying in text which is also called as sentiment analysis has been done in the past and it is still ongoing to find maximum accuracy.

But our focus is on detecting cyberbullying in images, since less work has been done on this. Cyberbullying using images includes the usage of inappropriate and embarrassing images. Examples of cyberbullying using images include threatening to share inappropriate images of the victim in order to control or blackmail them, taking degrading pictures of the victim and sharing them without his/her permission etc.

Few works on cyberbullying in images have been done using the captions and comments underneath the image but this is not sufficient because sometimes the image may not contain any caption and it still could be bullying or abusive so, the caption or comments are not the only factors that determine the nature of the image. So, our goal is to focus solely on the content of the image irrespective of caption or comments

Models to Detect Cyberbullying content in images have been designed in the past using feature extractors like SIFT (Scale invariant feature transform) and HUE, color histograms with the combination of Bag of visual words and using deep learning algorithms like SVM (Support Vector Machines), LLM (Log Linear model). But this still yields low accuracy rate and our goal is to design a model which yields better accuracy and better feature extraction using deep neural networks (CNN).

CHAPTER 3
LITERATURE SURVEY

3.1 A Framework for Cyberbullying Detection in Social Network

3.1.1 Author and publication details

Authors: Krishna B Kansara and Narendra M Shekokar

Date: 5th Feb 2019

3.1.2 Summary

- This article talks about how Social Media could pose threats like cyberbullying by allowing users to post or share some kinds of abusive or offensive content and the proposed method will be able to detect if it is abusive or not based on the content and features of images.
- Abusive Image Detection:

  Step 1: Feature extraction from abusive or inappropriate images and using Local Binary Pattern algorithm. LBP works by extracting interesting points from an image called as local descriptors.

  Step 2: Using extracted features and forming a visual vocabulary or codebook using K-means algorithm and the obtained clusters or visual words together is called as bag of visual words. Bag of visual words has the features which represents the whole dataset of abusive images.

Step 3: Here SVM is used for classification of given input image as abusive or not. SVM algorithm builds a model that assigns new image into either class based on whether the image contains abusive or inappropriate contents or not. They have used Boolean system to express the likelihoodof the cyberbullying content by analyzing image features of the given image.

3.1.3 Results

The proposed model provides recall rate of 66% and since LBP is used as feature extractor here invariant to rotation and Increase in neighbors will lead to more response time and increase of computational complexity. Only pixel difference is used by LBP which is not sufficient.

3.2 LLM and BOVW Models for Adult Image Classification and Filtering

3.2.1 Author and publication details

Authors: Thomas Deselaers, Lexi Pimenidis and Hermann Ney

Date: Sept 2013

3.2.2 Summary

- This article talks about how inappropriate images could be detected by using PCA, GMM (Gaussian mixture model) and BOVW (Bag of Visual words).

3.2.1.1 Step 1: Image patches are extracted as local features around difference-of-Gaussian interest points and then they are scaled to a common size and then PCA to reduce dimensionality.

3.2.1.2 Step 2: Then they have created a bag of visual words using training algorithm for unsupervised training of GMM (Gaussian Mixture Model) and visual vocabulary that has been learned will be able to recognize frequent patterns in training images.

3.2.1.3 Step 3: Here classification to determine the class of images is represented by histograms. Using Log-linear models and SVM, the classification has been done. There are 5 classes here,

class 0: inoffensive images,

class 1: If people are lightly dressed, could be offensive in strict environment,

class 2: partially nude people, could be objected in schools,

class 3: completely nude people, objectionable in many environments class 4: pornographic images, offensive in almost every environment.

3.2.3 Results

Here there was more confusion between class 2 and class 3 and that affected the accuracy. Even though the color information was included it didn’t do much improvement in the accuracy and also since LLM is used here it is difficult to capture complete relationships using this model, algorithms which are more powerful and complex such as Neural Networks can easily outperform this algorithm.

3.3 Image Retrieval by Bag of Visual Words and Color Information

3.3.1 Author and publication details

Authors: N. Mansoori, M. Netaji, P. Razzaghi and S. Samavi

Date: May 2013

3.3.2 Summary

- This article talks about how content-based image classification using features of an image like color, texture and shape. Here they have used two feature extractors like SIFT (Scale Invariant Feature Transform) and HUE to get more accuracy.
- Feature Extraction: Here SIFT is used to extract the interesting points from an image which are called as key points and each key point has a descriptor.
- Feature Description: Here the extracted key points are described. Two descriptors are used here SIFT and HUE descriptors. SIFT detects local patches and HUE produces the hue descriptor which is invariant to both lighting geometry and specularities.
- Quantization: The descriptors obtained using SIFT and HUE are quantized to generate a codebook and this is done using K-Means algorithm, Quantization has a good impact on the final result.

- Training: Here for each image in the dataset, SIFT features are extracted and then these features are described by SIFT and HUE descriptors. Then codebook is obtained using k- means which needs the number of clusters. For each image according to the codebook BOW model are constructed.
- Testing Stage: Retrieving-here images are retrieved or matched according to the similarity values.

3.3.3 Results and conclusions

Using Both SIFT and HUE descriptors to classify an image into its class gave an average precision of 63.39%. The accuracy is less because during codebook generation it resulted in loss in quantization that can degrade the retrieval or matching performance.

3.4 Document Embedding Generation for Cyber-Aggressive Comment Detection using Supervised Machine Learning Approach

3.4.1 Author and publication details

Authors: Shylaja S S, Abhishek Narayanan, Abhijith Venugopal, Abhishek Prasad
Year:2019

3.4.2 Summary

- The main purpose of this paper is to detect cyber bullying comments on social media despite of challenging task in this paper is to detect the sender's tone through his text message. For this one need to obtain manually labelled data. From various sources comments can be collected and can be labelled based on the cyber aggressiveness. These collected comments from various sources will be grouped into datasets, and document embeddings will be generated which, then will be fed to the supervised machine learning algorithms for the training and prediction of test labels.

- Classification can be done by following 5 machine learning classification algorithms:
  1. Support Vector Machines (SVM)
  2. Logistic Regression
  3. Bernoulli Naive Bayes Algorithm
  4. Decision Trees
  5. Random Forest Classifier: Evaluation metrics

Various factors are considered as evaluation metrics such as Accuracy score, K-Fold Cross Validation, Confusion matrices, Precision, Recall, F-Beta-score, and Area under ROC Curve.

3.4.3 Results and conclusion:

Using all the algorithms for finding out the comments to be aggressive or non-aggressive, we found that doc2vec approach along with SVM classifier with rbf kernel has given an increased accuracy of 88.465%. Future work of this paper may include application of deep learning algorithms to the existing one for further optimization of the results.

3.5 Introduction to the special issue on deep learning for real-time information hiding and forensics

3.5.1 Authors and Publication details

Authors: Zhili Zhou, Ching-Nung Yang, Cheonshik Kim, Stelvio Cimato
Date: 28 January 2020

3.5.2 Summary

- Technology is evolving and growing rapidly. At the same time, the threat to multimedia data is also increasing due to various powerful multimedia processing tools. Due to which multimedia data can be forged and illegally copied. In order to prevent the above from unauthorized use, two approaches are proposed. They are, Information hiding digital forensics which complement each other. Now let's look into Image forensics, Image steganography and steganalysis.

- In Image forensics detection and classification of images can be done in 4 ways:

  1. Non-aligned double JPEG (NA-DJPEG) compression
  2. Real-time estimation for the parameters of Gaussian filtering via deep learning
  3. A real-time image forensics scheme based on multi-domain learning
  4. Image steganography and steganalysis

- Image steganography: The improvement of multimedia and deep learning technology carries new difficulties to steganography and steganalysis methods. To improve the correspondence dependability and proficiency for current ongoing vigorous steganography strategies, a connected code, made out of syndrome-trellis codes (STC) and cyclic redundancy check (CRC) codes, is proposed here.
- Image steganalysis: Steganalysis is the contrary technique of steganography. Fundamentally, we attempt to distinguish the presence of steganographic content in a computerized gadget and furthermore find the hidden message. Steganography is a technique that transmits secret data or messages in an appropriate multimedia carrier, e.g., image, audio, and video files.

3.6 Image analysis of cyberbullying using machine learning techniques

3.6.1 Authors and Publications details

Author: Hou Li

Year: 2016

3.6.2 Summary

- The intent of this paper is to find a prediction method to advise social media site users about what and what not to post in their accounts in order to avoid being bullied. Using ML algorithms, they train and test models on data crawled from Instagram.
- Classification: The image data is crawled from Instagram along with the metadata and other contextual information and also few of the related comments. They used Instagram API to achieve the same. According to a survey by Pew Research Center, Instagram is used by 52% teens and 40% claim it's the most used social media website.
- Labelling Instagram images-They had a two-question questionnaire for labelers. They were shown the photos and set of comments. Following that two questions were asked.

  a) Are the comments having any bullying content or not?

  b) Does the image contain any bullying content? If yes, is it because of the comments?

- The feature extraction techniques used in this paper include color histogram, edge detection coherence vector, scale invariant feature transform (SIFT), face features to describe the images and caption and user info were also considered as an auxiliary feature. All these features discussed above helped in detection of cyberbullying content when used along with the Machine Learning models.
- The supervised classification methods used are k-nearest neighbors, binary decision tree and SVM. Besides using different models, they have also analyzed the feature vectors and their — combinations to get a better classification model and high accuracy and it was observed that SVM gave the best result. They were able to get an overall accuracy of 68.84%.

3.6.3 Research gaps

- Users post photos with different purposes. Bullying might occur because of the user who posted the photo or the content of the images that the commenters have strong sentiment about. This will increase the difficulty of this classification problem.
- Here they have asked the labelers to label the bullied/non-bullied images by going through the given set of images and comments. This would be affected by the commenter's preferences. If the labelers label the photos based only on the subjective opinions, the classification would become more accurate.

3.7 Convolutional Neural Network for Inappropriate Image Classification

3.7.1 Authors and Publication details

Authors: Artha Agastya, Arief Setyanto, Dini Oktarina

Year: 2018

3.7.2 Summary

- CNN is one of the deep learning approaches that is used to classify the inappropriate images. The aim of this paper is to implement a CNN model to classify these images. These images are extracted from 800 videos and with a total of 16,727 images.
- This paper talks about how CNN is superior at classification of images.
- This paper talks about VCG-16 model which was developed at Oxford University. The model is trained using Image Net database, which include around 1.2 million training images, 50,000 validation images and 100,000 testing images. It has around 1000 prediction labels.
- This VCG-16 model is formed with 13 convolutional layers and 3 dense or fully connected layers. The convolutional layers use a 3x3 window and the pooling layer uses a 2x2 window. SoftMax gradient descent algorithm is applied in the prediction layer and ReLu is used as activation unit.
- The paper talks about modifications done to the existing VCG-16 model and how it improved the classification.

3.7.3 Results and Conclusion

- The testing accuracy reached to 93.8% from 91% on average. Compared to the existing method the proposed method has better precision with a 2% difference.
- To improve the accuracy, change the learning algorithm or fully connected layer's structure or even combine the VCG-16 with another variant of deep learning such as RNN and LSTM.

CHAPTER 4

PROJECT REQUIREMENTS SPECIFICATION

The high-level design for our project includes the following

1) Preprocessing of the bullied and non-bullied images.
2) Augmentation of the images
3) Training CNN model to do the classification
4) Building UI to show the above classification implemented.

Current System

Literature survey on the topic was carried out and it was observed that most of them are text classification. Various ML algorithms were used to classify the same. This kind of work on text classification is called sentiment analysis and the work is still ongoing to better the accuracy.

In recent years there have been many researchers who are interested in cyberbullying detection on images. Here few works have been done using available feature extraction techniques and classification is done using ML and DL algorithms.

Design Considerations

1. Design Goals

Our techniques use CNN for image classification. Currently there are few works done on image classification but with less precision. Using the dataset, we will be comparing our model with the existing models.

2. Architecture Choices

An alternate choice of feature extraction was considered but we feel our approach is better because

a) Selection of the type of feature extraction is very important and selected feature extractor may not work well with the all the images.

b) We need to do various trial and error method to get a good feature extractor to get a better accuracy. This might also involve combining different feature extraction technique and verifying the result.

3. Design Dependencies

- Tensorflow
- Keras
- Layers from tenserflow.keras
- Preprocessing.image from tensorflow. keras
- ImageDataGenerator from keras
- MobilenetV2 pre-trained network

CHAPTER 5

SYSTEM REQUIREMENTS SPECIFICATION

System Constraints Requires a high GPU memory since a lot of images have to be trained to have a better classification model. A high GPU memory affects the easy run on the local system and requires dependency on platforms like Google Colab.

Software Dependencies

- Technologies: Python, TensorFlow, scikit learn
- Tools: GPU by Google Colab, Anaconda Navigator
- Database: SQLite
- Google Drive, Google Colab
- Web Development: HTML, JavaScript, Django

Hardware Dependencies

- Working PC
- RAM size > 8GB
- Internet connection

CHAPTER 6

SYSTEM DESIGN

Why CNN??

- The main reason for using CNN is it doesn't require feature extraction.
- We need to extract the features manually in other models. When these two models were compared CNN gave a better accuracy because it covers both the local and global features and also different features are learned from the images.
- In the algorithms which uses feature extraction we need to select both local and global features and classifiers manually. In some cases, if one kind of feature worked well but, in another case, some other feature might work well.
- CNNs are useful for classification of images as dimensionality reduction suits well for the large number of parameters in an image.
- CNNs work well with large dataset.

It is very difficult to train a CNN model from scratch because it is very hard to get a large dataset to achieve a good model performance. Hence it is very common to use an existing pre-trained network which is trained on a very large dataset and use it in our classification model after tuning to our requirement. This is called Transfer Learning.

The important benefits of Transfer Learning are:

1. Decreases the training time.
2. Less data is required since it is already a trained model.
3. These models are developed by deep learning experts.

We have used MobileNetV2 model as a base model to construct the CNN model.

MobileNet Model

MobileNet is a model that is used for classification and detection. They are a class of small, low power and low-latency models. These are considered as a great deep learning models that can be used on mobile devices because of its small size. These models require less space and parameters when compared with other transfer learning models. Therefore, MobileNet is used in this project.

Design Description

We import all the modules mentioned above in the design dependencies section, and since we require different kinds of image to train our model, image data augmentation is performed. For augmentation we have used Keras ImageDataGenerator which lets us augment images in real-time while model is still training.

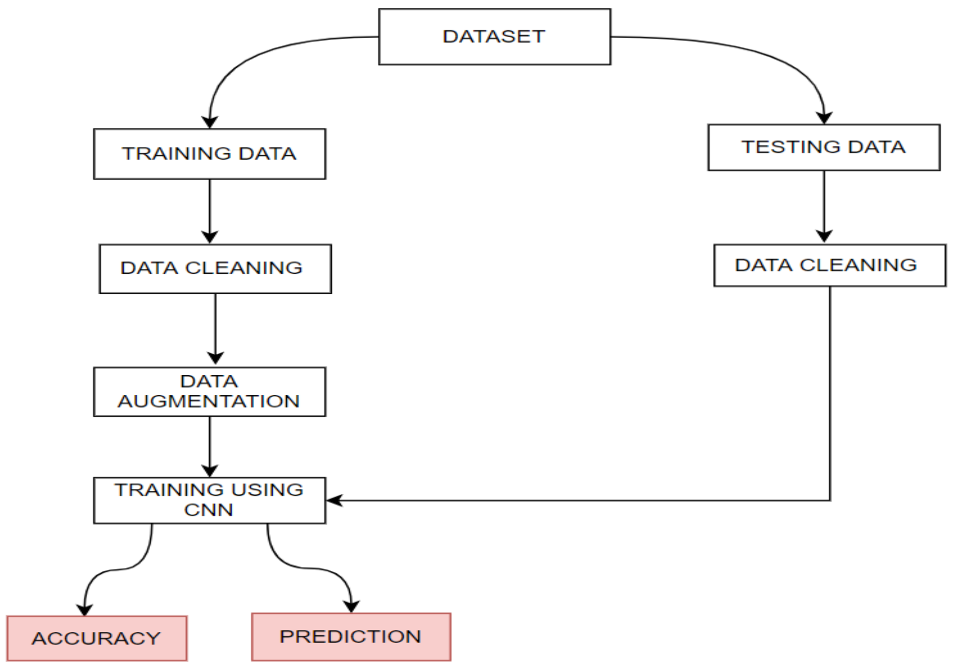


Fig 1. Flowchart of the CNN model

1.Augmentation

Image Augmentation is a technique which is used to increase the size of the training dataset artificially by creating different modified versions of the original images in the dataset. In Keras deep learning library augmentation is carried using ImageDataGenerator class. The images generated here are then used for training the model.

2.Classification Model

To design our model, we have used one of the most important library of deep learning model i.e., Tensorflow. TensorFlow is an open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. We have also used Keras, which is a high-level API that is built on top of TensorFlow. It is extremely user-friendly and easier. We have imported modules from tensorflow.keras.layers such as Flatten Conv2D, MaxPooling2D, Dropout, Dense and from tensorflow. keras preprocessing. image such as ImageDataGenerator, img_to_array, load_img. We have used pre-trained model called as MobileNetV2, which is a CNN architecture model for Image Classification. It takes very less computation power to run or apply transfer learning. It is also best suited for web browsers as browsers have limitation over computation, graphic processing and storage. By setting Batch size to 16 and by varying the epochs we are building a model with best accuracy to classify the image into Bullying or Non-Bullying.

3.User Interface

The diagram below shows the flow of how the CNN model built is used in our user interface.This is basically an application of where our model can be used to prevent cyberbullying.The user interface built is a basic interface similar to instagram where any user who has an account can upload photos to their gallery.Before uploading any images the basic requirement is to have an account.Once the user has an account before uploading any image our model discards the image if its bullying and doesn't allow the user to upload to his gallery.This is done with the intention of any user from getting hate online after they post some bullying content on their gallery.

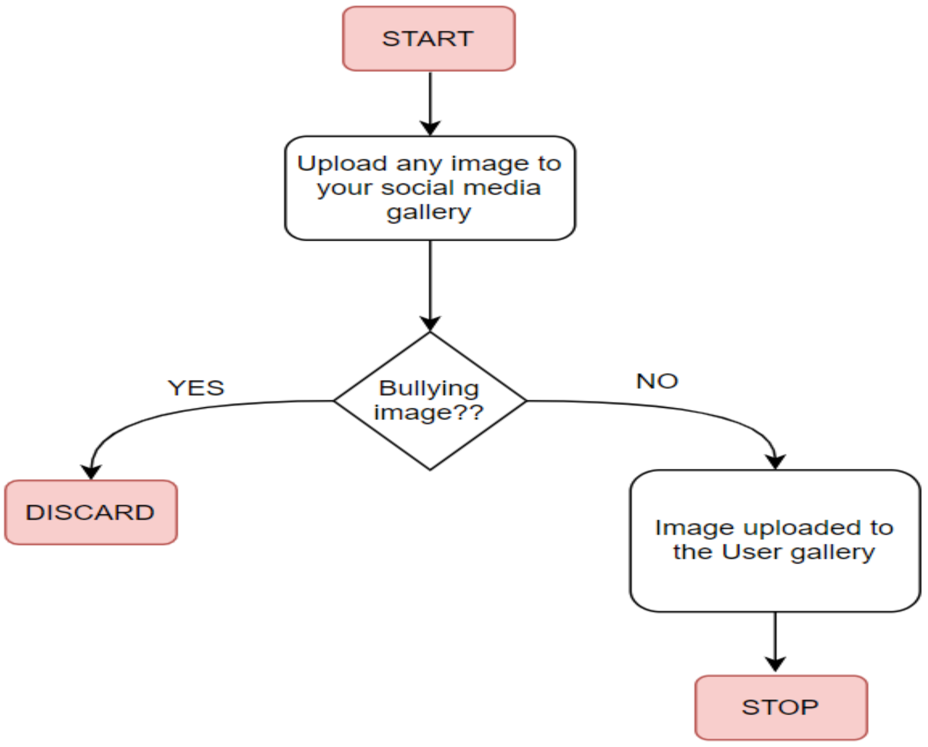

START
Upload any image to your social media gallery
Bullying image??
YES
NO
DISCARD
Image uploaded to the User gallery
STOP

Fig 2. Flowchart of the User Interface

CHAPTER 7

IMPLEMENTATION AND PSEUDOCODE

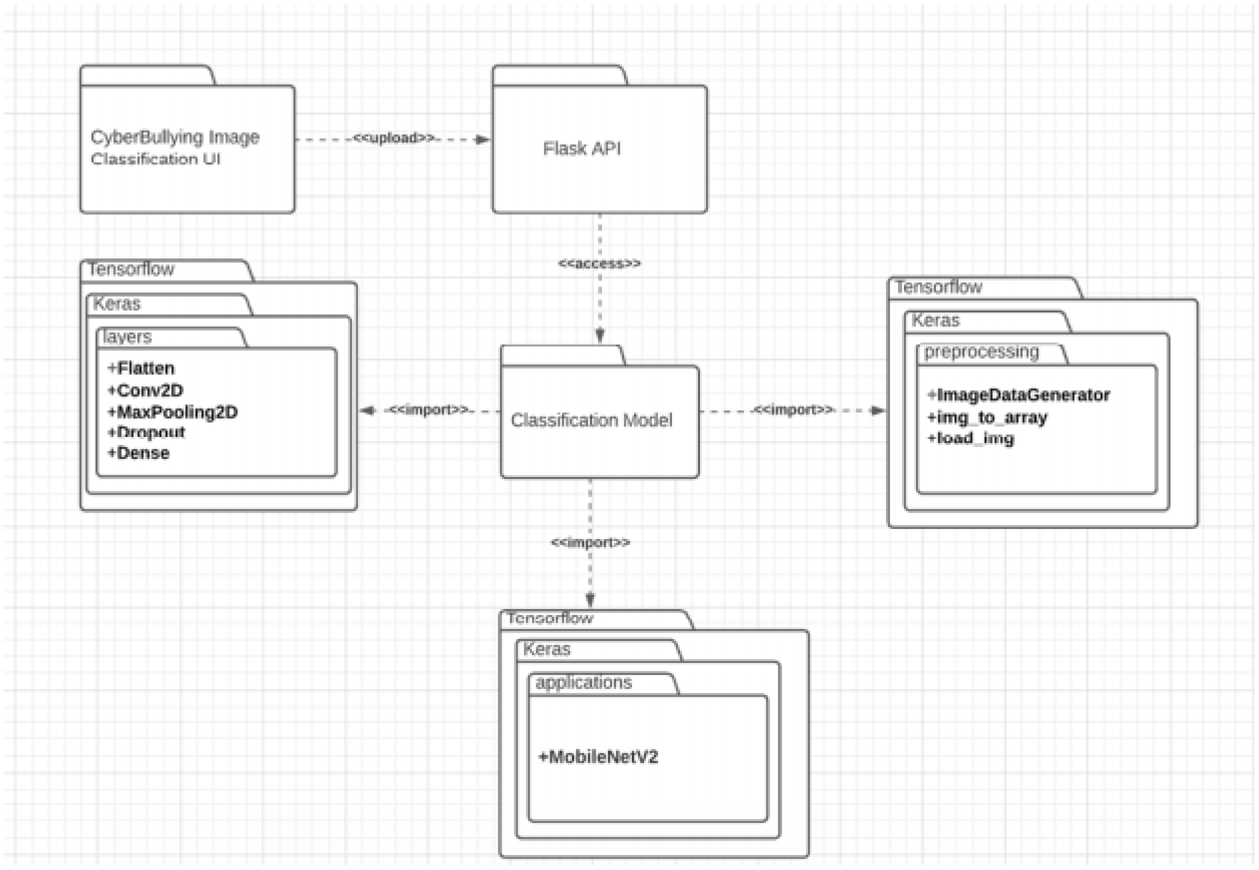


Fig 3. Package Diagram

1.Modules used to construct CNN model

The model implemented is a neural-network model which is designed to detect any cyber-bullying content present in the images. To design this model, the libraries used are keras and tensorflow.

1)**Keras** – Keras is a high-level neural network python library. It is made with focus of understanding deep learning techniques, such as creating layers for neural networks.

2)Tensorflow - TensorFlow is a software library or framework, designed by the Google team to implement machine learning and deep learning concepts in the easiest manner.

3)Keras layers API
Layers are the basic building blocks of neural networks in Keras. Many functions from keras. layers are used here such as:
Conv2D- This layer is used to create a layer or a kernel which has a height and a width.
Maxpooling2D- Maximum pooling, or max pooling, is a pooling operation that calculates the maximum, or largest, value in each patch of each feature map.

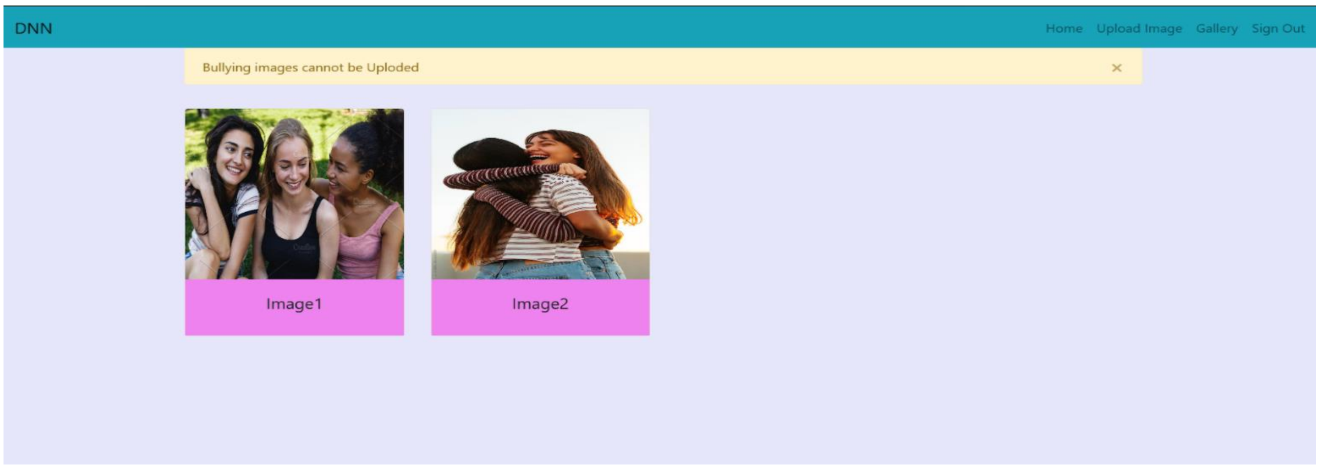
4) Keras ImageDataGenerator ()-This function lets us augment images in real-time while the model is still training. Data Augmentation is done using this in order to increase the size of the dataset and introduce variability in the dataset. This helps in increasing the efficiency model.

5) Img_to_array ()- Keras provides this function for converting a loaded image in PIL format into a NumPy array for use with deep learning models.

6)Load_img ()- Keras provides the **load_img** () function for loading an image from file as a PIL image object. In this format it is easy to open, edit, manipulate images.

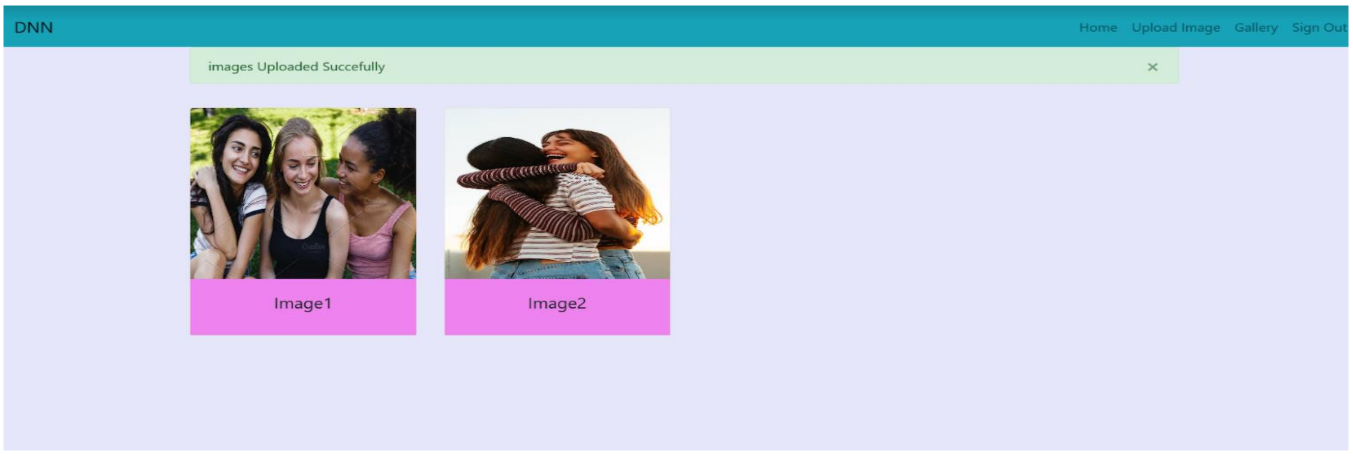
2. User Interface as a part of model application

As a part of user interface, we have created an application similar to Instagram where a user who has created an account can post images to their gallery. If the user tries to post any images which has any bullying content, then it prevents them from posting those images in their gallery and preventing them from getting bullied online. Fig 4 and Fig 5 shows a user trying to uploading a bullying image and non-bullying image respectively.



Bullying images cannot be Uploded

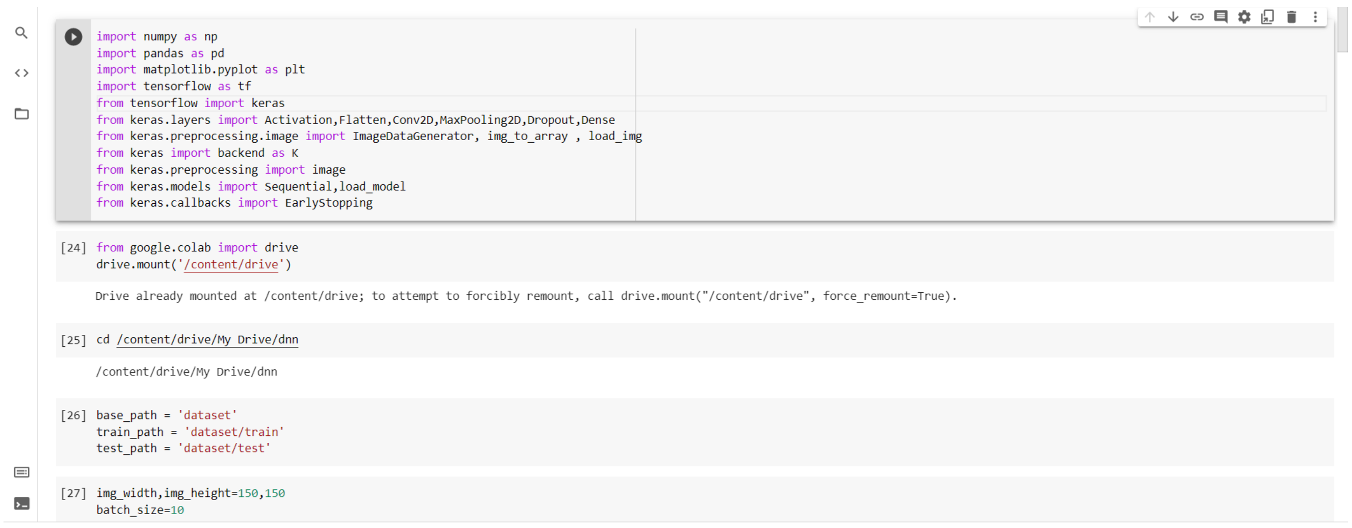
Fig 4. User Interface i



images Uploaded Succefully

Fig 5. User Interface ii

Implementation of cyber bullying detection model without Transfer Learning



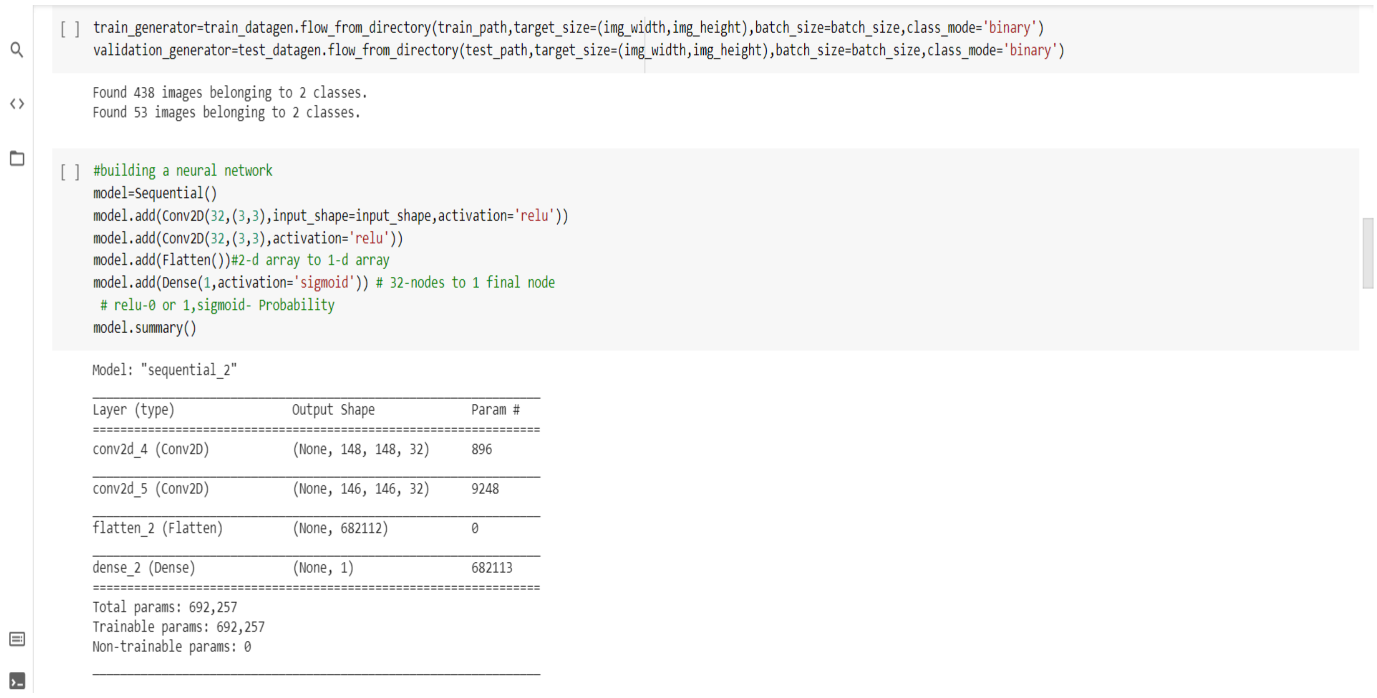
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow import keras
from keras.layers import Activation,Flatten,Conv2D,MaxPooling2D,Dropout,Dense
from keras.preprocessing.image import ImageDataGenerator, img_to_array , load_img
from keras import backend as K
from keras.preprocessing import image
from keras.models import Sequential,load_model
from keras.callbacks import EarlyStopping

[24] from google.colab import drive
     drive.mount('/content/drive')

     Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

[25] cd /content/drive/My Drive/dnn

     /content/drive/My Drive/dnn

[26] base_path = 'dataset'
     train_path = 'dataset/train'
     test_path = 'dataset/test'

[27] img_width,img_height=150,150
     batch_size=10
```

Fig 6. Libraries used

Implementation of the model has been done using google Colab because it provides RAM and its very efficient and fast for computational purposes like training a neural network model etc.

To build our model we have mainly used keras and TensorFlow. We have used many libraries from keras such as layers, backend, Image, Sequential and Early Stopping. In the above snapshot we have defined base-path, train-path, test-path from which images will flow while training the model. We have defined the image size as 150*150 and batch-size=10, where 10 images from dataset at once while training the model in a epoch before modifying the parameters in that particular epoch.

We train our model to improve its efficiency and should be able to generalize the image data. This is done by using the technique called as augmentation and its done using class provided by Keras called as ImageDataGenerator, which provides variation of a single image which is provided as input. For example, if a normal image is given as input variations such as rotated by 30 degree, horizontal-flip, vertical flip and many other such variations will be done by ImageDataGenerator where these are parameters that are given.

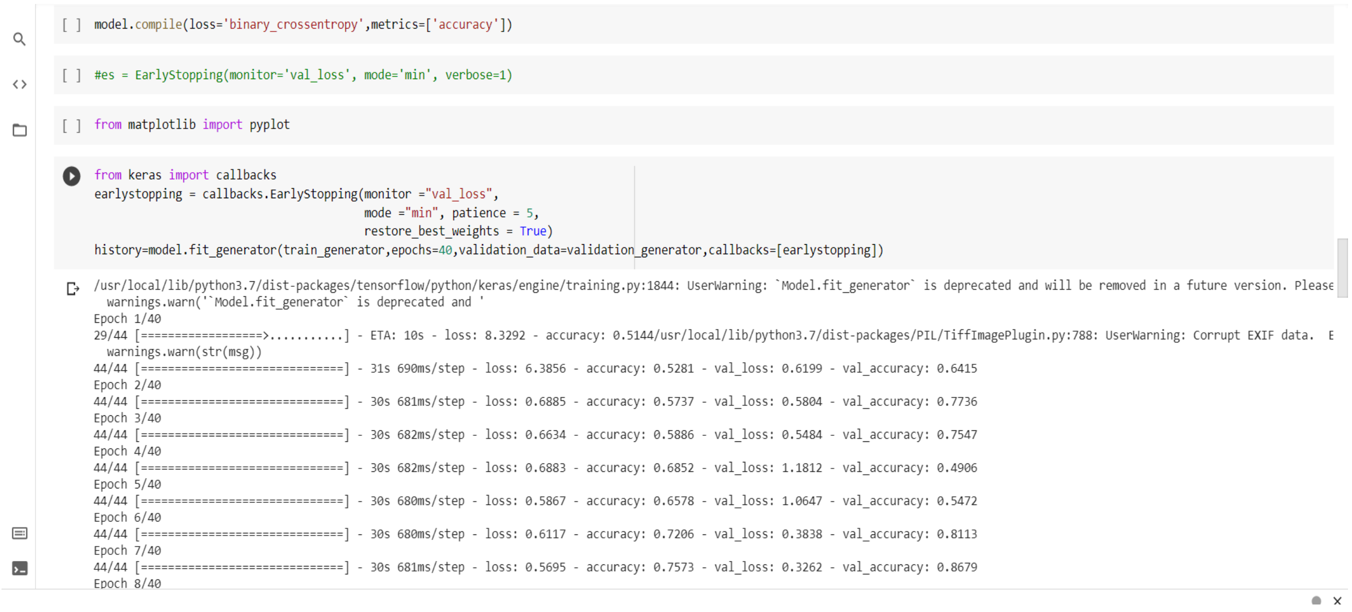


```
[ ] train_generator=train_datagen.flow_from_directory(train_path,target_size=(img_width,img_height),batch_size=batch_size,class_mode='binary')
    validation_generator=test_datagen.flow_from_directory(test_path,target_size=(img_width,img_height),batch_size=batch_size,class_mode='binary')

    Found 438 images belonging to 2 classes.
    Found 53 images belonging to 2 classes.

[ ] #building a neural network
    model=Sequential()
    model.add(Conv2D(32,(3,3),input_shape=input_shape,activation='relu'))
    model.add(Conv2D(32,(3,3),activation='relu'))
    model.add(Flatten())#2-d array to 1-d array
    model.add(Dense(1,activation='sigmoid')) # 32-nodes to 1 final node
     # relu-0 or 1,sigmoid- Probability
    model.summary()

    Model: "sequential_2"
    _________________________________________________________________
    Layer (type)                 Output Shape              Param #
    =================================================================
    conv2d_4 (Conv2D)            (None, 148, 148, 32)      896
    _________________________________________________________________
    conv2d_5 (Conv2D)            (None, 146, 146, 32)      9248
    _________________________________________________________________
    flatten_2 (Flatten)          (None, 682112)            0
    _________________________________________________________________
    dense_2 (Dense)              (None, 1)                 682113
    =================================================================
    Total params: 692,257
    Trainable params: 692,257
    Non-trainable params: 0
    _________________________________________________________________
```

Fig 7. Neural Network Model

We have the passed the training images to train generator parameter and testing images as validation_generator. These will be used while training the model, images from the training will be passed in batches and variants of image will be produced and the model will be trained and while validating the model, variants of the testing images will be formed and they will be passed in batches which the model uses to produce the validation accuracy.

To build a model we have used sequential api to create a sequential model and the input layer takes in the images of input shape (150,150,3) where 3 stands for 3 channels called RGB. After the input layer we have introduced conv2D hidden layer where we have 32 neural nodes and 3*3 filter which slides across the input image and produces 32 feature maps of that particular image and the activation unit which has been used here is RELU. Once the feature maps of the input image are ready, they are passed to next layer which is a flattening where a 2-d data is converted into 1-d array. Once converted into array, it is passed to a dense layer where the final prediction takes place and the output will be in terms of probability, and the activation unit used here is Sigmoid, since it's a binary classification.



```
[ ] model.compile(loss='binary_crossentropy',metrics=['accuracy'])

[ ] #es = EarlyStopping(monitor='val_loss', mode='min', verbose=1)

[ ] from matplotlib import pyplot

from keras import callbacks
earlystopping = callbacks.EarlyStopping(monitor ="val_loss",
                                        mode ="min", patience = 5,
                                        restore_best_weights = True)
history=model.fit_generator(train_generator,epochs=40,validation_data=validation_generator,callbacks=[earlystopping])
```

```
/usr/local/lib/python3.7/dist-packages/tensorflow/python/keras/engine/training.py:1844: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please
  warnings.warn('`Model.fit_generator` is deprecated and '
Epoch 1/40
29/44 [=================>..........] - ETA: 10s - loss: 8.3292 - accuracy: 0.5144/usr/local/lib/python3.7/dist-packages/PIL/TiffImagePlugin.py:788: UserWarning: Corrupt EXIF data.  E
  warnings.warn(str(msg))
44/44 [==============================] - 31s 690ms/step - loss: 6.3856 - accuracy: 0.5281 - val_loss: 0.6199 - val_accuracy: 0.6415
Epoch 2/40
44/44 [==============================] - 30s 681ms/step - loss: 0.6885 - accuracy: 0.5737 - val_loss: 0.5804 - val_accuracy: 0.7736
Epoch 3/40
44/44 [==============================] - 30s 682ms/step - loss: 0.6634 - accuracy: 0.5886 - val_loss: 0.5484 - val_accuracy: 0.7547
Epoch 4/40
44/44 [==============================] - 30s 682ms/step - loss: 0.6883 - accuracy: 0.6852 - val_loss: 1.1812 - val_accuracy: 0.4906
Epoch 5/40
44/44 [==============================] - 30s 680ms/step - loss: 0.5867 - accuracy: 0.6578 - val_loss: 1.0647 - val_accuracy: 0.5472
Epoch 6/40
44/44 [==============================] - 30s 680ms/step - loss: 0.6117 - accuracy: 0.7206 - val_loss: 0.3838 - val_accuracy: 0.8113
Epoch 7/40
44/44 [==============================] - 30s 681ms/step - loss: 0.5695 - accuracy: 0.7573 - val_loss: 0.3262 - val_accuracy: 0.8679
Epoch 8/40
```

Fig 8. Training of the Model

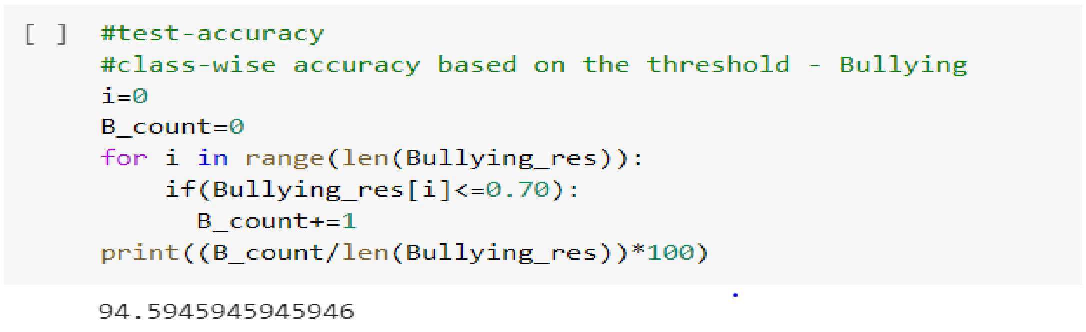
CHAPTER 8

RESULTS AND CONCLUSIONS

A) TRANSFER LEARNING MODEL

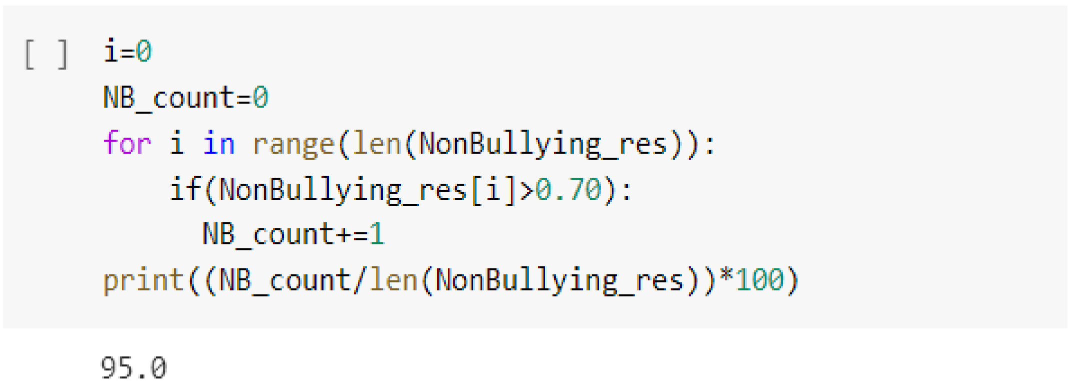
8.1 Class-wise accuracy and overall model accuracy

We have achieved an accuracy of 94.54 % for Bullying class and 95 % for non-bullying class with the CNN model. The overall accuracy for the model designed is 94.73 %.



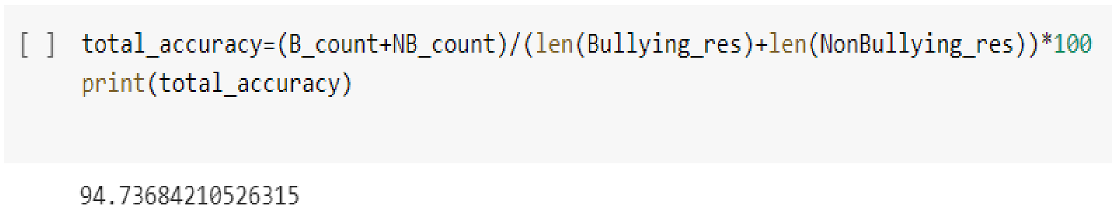
```
[ ]  #test-accuracy
     #class-wise accuracy based on the threshold - Bullying
     i=0
     B_count=0
     for i in range(len(Bullying_res)):
         if(Bullying_res[i]<=0.70):
           B_count+=1
     print((B_count/len(Bullying_res))*100)

     94.5945945945946
```

Fig 9. Test accuracy for bullying class



```
[ ]  i=0
     NB_count=0
     for i in range(len(NonBullying_res)):
         if(NonBullying_res[i]>0.70):
           NB_count+=1
     print((NB_count/len(NonBullying_res))*100)

     95.0
```

Fig 10. Test accuracy for non-bullying class

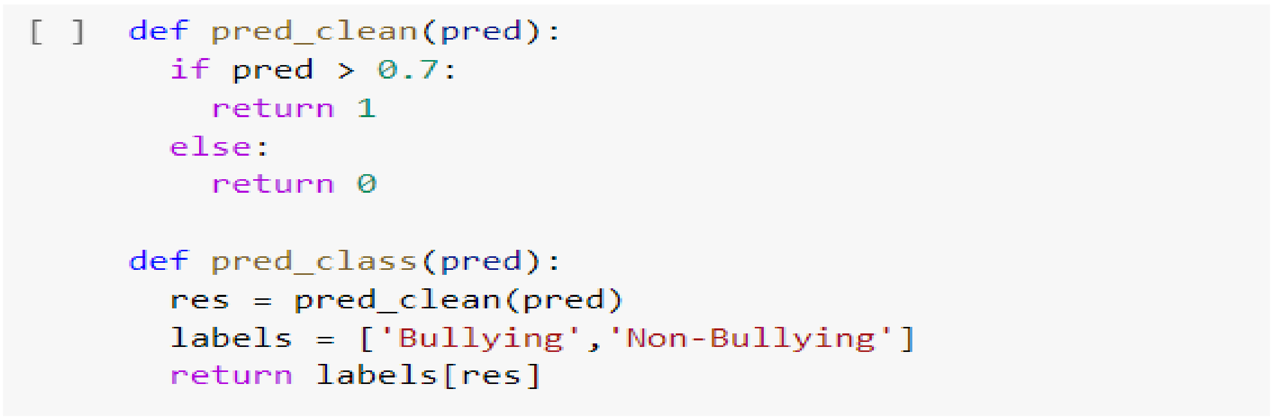


```
[ ]  total_accuracy=(B_count+NB_count)/(len(Bullying_res)+len(NonBullying_res))*100
     print(total_accuracy)

     94.73684210526315
```

Fig 11. Total accuracy of the CNN model

8.2 Prediction

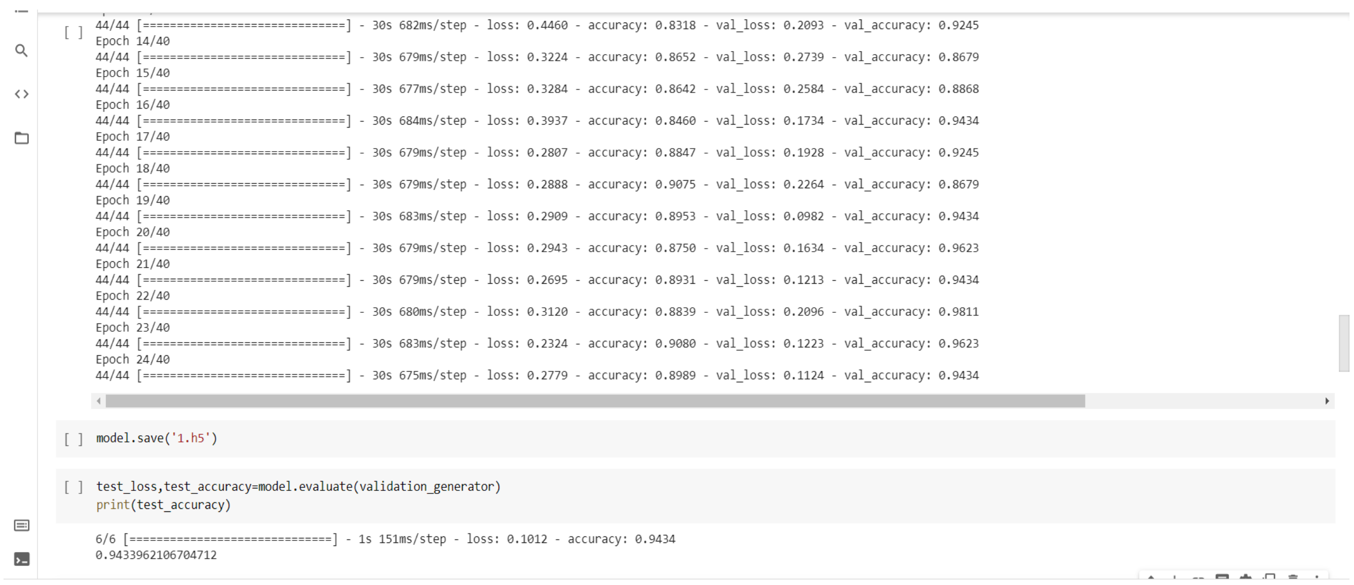
The model predicts to which class the images belong to in terms of probabilities. It assigns each class a range of probabilities. After training the model, probabilities are assigned to each class and according to our interest we set our threshold to 0.7 after analyzing the probabilities given out by the model for both classes. If the probability of the image is greater than 0.7 its Non-Bullying else it's Bullying.



```
[ ]  def pred_clean(pred):
       if pred > 0.7:
         return 1
       else:
         return 0

     def pred_class(pred):
       res = pred_clean(pred)
       labels = ['Bullying','Non-Bullying']
       return labels[res]
```

Fig 12. Prediction function

B) MODEL WITHOUT USING TRANSFER LEARNING



```
44/44 [==============================] - 30s 682ms/step - loss: 0.4460 - accuracy: 0.8318 - val_loss: 0.2093 - val_accuracy: 0.9245
Epoch 14/40
44/44 [==============================] - 30s 679ms/step - loss: 0.3224 - accuracy: 0.8652 - val_loss: 0.2739 - val_accuracy: 0.8679
Epoch 15/40
44/44 [==============================] - 30s 677ms/step - loss: 0.3284 - accuracy: 0.8642 - val_loss: 0.2584 - val_accuracy: 0.8868
Epoch 16/40
44/44 [==============================] - 30s 684ms/step - loss: 0.3937 - accuracy: 0.8460 - val_loss: 0.1734 - val_accuracy: 0.9434
Epoch 17/40
44/44 [==============================] - 30s 679ms/step - loss: 0.2807 - accuracy: 0.8847 - val_loss: 0.1928 - val_accuracy: 0.9245
Epoch 18/40
44/44 [==============================] - 30s 679ms/step - loss: 0.2888 - accuracy: 0.9075 - val_loss: 0.2264 - val_accuracy: 0.8679
Epoch 19/40
44/44 [==============================] - 30s 683ms/step - loss: 0.2909 - accuracy: 0.8953 - val_loss: 0.0982 - val_accuracy: 0.9434
Epoch 20/40
44/44 [==============================] - 30s 679ms/step - loss: 0.2943 - accuracy: 0.8750 - val_loss: 0.1634 - val_accuracy: 0.9623
Epoch 21/40
44/44 [==============================] - 30s 679ms/step - loss: 0.2695 - accuracy: 0.8931 - val_loss: 0.1213 - val_accuracy: 0.9434
Epoch 22/40
44/44 [==============================] - 30s 680ms/step - loss: 0.3120 - accuracy: 0.8839 - val_loss: 0.2096 - val_accuracy: 0.9811
Epoch 23/40
44/44 [==============================] - 30s 683ms/step - loss: 0.2324 - accuracy: 0.9080 - val_loss: 0.1223 - val_accuracy: 0.9623
Epoch 24/40
44/44 [==============================] - 30s 675ms/step - loss: 0.2779 - accuracy: 0.8989 - val_loss: 0.1124 - val_accuracy: 0.9434
```

```
[ ] model.save('1.h5')
```

```
[ ] test_loss,test_accuracy=model.evaluate(validation_generator)
    print(test_accuracy)

6/6 [==============================] - 1s 151ms/step - loss: 0.1012 - accuracy: 0.9434
0.9433962106704712
```

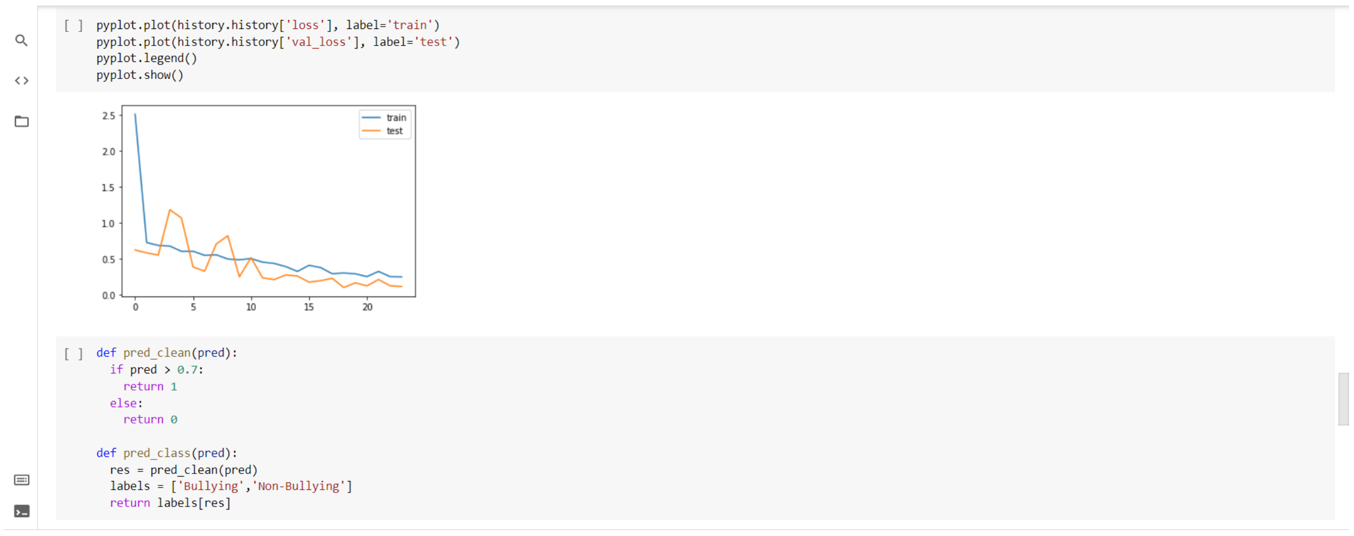
Fig 13. Model accuracy

To monitor loss and accuracy we have used binary_crossentropy and metrics accuracy respectively. In order to avoid overfitting, the model with increased number of epochs, we used callback method from earlystopping. EarlyStopping is a technique which monitors validation loss and for each epoch it checks the validation loss, as loss decreases the accuracy should increase keeping this in consideration, if the validation loss keeps on decreasing after many epochs which means the model is overfitting and it's considering irrelevant features from the image to classify into either classes and the loss keeps increasing because of this.

Here we have given epochs – 40 and we have used earlystopping and the earlystopping function has stopped at epoch 24 because the loss kept on increasing and at 24th epoch we have got 94.34 percent validation accuracy.

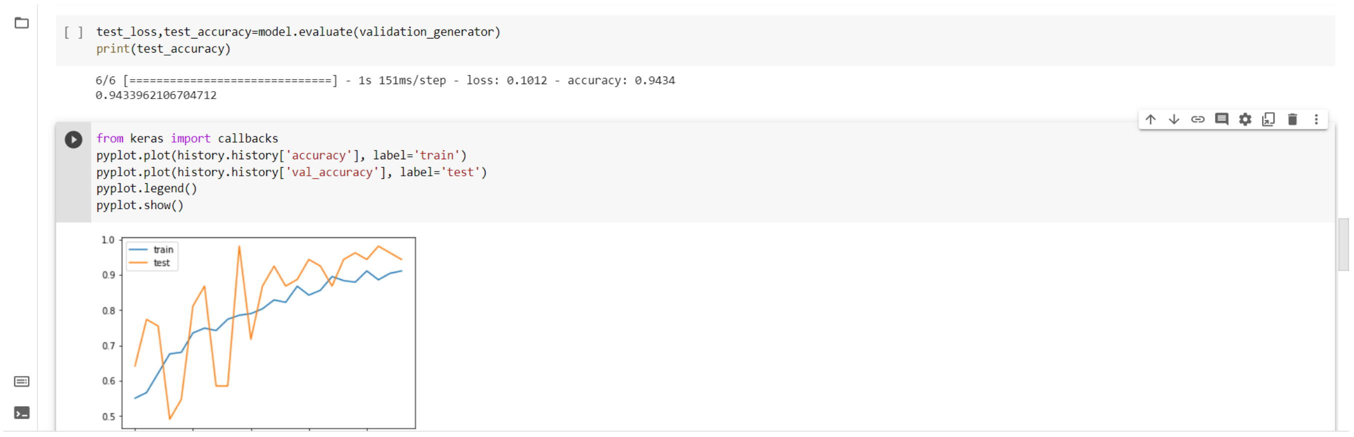
After saving the model, we have passed the testing dataset to the model and the obtained accuracy is 94.34 percent accuracy.

Below two graphs show us with the increase in number of epochs the validation accuracy and training accuracy kept increasing with very few variations until 24th epoch.
Second graph suggests that with the increase number of epochs the validation loss kept decreasing with little variations until 24th epoch.
To summarize we can see that with decrease in loss, the accuracy is increasing.


```
[ ] pyplot.plot(history.history['loss'], label='train')
    pyplot.plot(history.history['val_loss'], label='test')
    pyplot.legend()
    pyplot.show()
```

```
[ ] def pred_clean(pred):
      if pred > 0.7:
        return 1
      else:
        return 0

    def pred_class(pred):
      res = pred_clean(pred)
      labels = ['Bullying','Non-Bullying']
      return labels[res]
```

Fig 14. loss vs validation loss plot

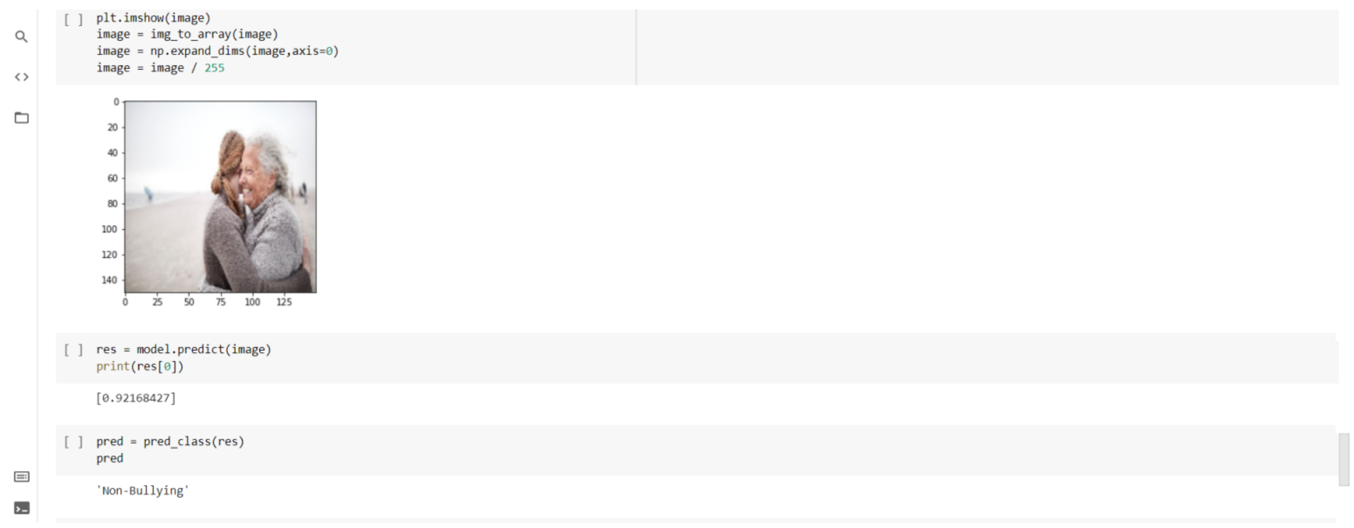


```
test_loss,test_accuracy=model.evaluate(validation_generator)
print(test_accuracy)
```

```
6/6 [==============================] - 1s 151ms/step - loss: 0.1012 - accuracy: 0.9434
0.9433962106704712
```

```
from keras import callbacks
pyplot.plot(history.history['accuracy'], label='train')
pyplot.plot(history.history['val_accuracy'], label='test')
pyplot.legend()
pyplot.show()
```

Fig 15. Accuracy vs validation accuracy plot



```
plt.imshow(image)
image = img_to_array(image)
image = np.expand_dims(image,axis=0)
image = image / 255
```

```
res = model.predict(image)
print(res[0])
```

```
[0.92168427]
```

```
pred = pred_class(res)
pred
```

```
'Non-Bullying'
```

Fig 16. Prediction

Once the model is trained, we have a given a non-bullying image to our model and our model has assigned a range of probabilities to both classes and for this non-bullying image, it has assigned a probability of 0.92 and according to pred which we have defined its above 0.7, so its non-bullying.

REFERENCE /BIBLIOGRAPHY

[1] Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Emiliano, De Cristofaro, Gianluca Stringhini, Athena Vakali, Nicolas Kourtellis- "Detecting Cyberbullying and Cyberaggression in Social Media"-

[2] Squicciarini,Rajtmajer,Caragea,HaotiZhong,HaoLi,Christopher Griffin,DavidMiller -"Content-Driven Detection of Cyberbullying on the Instagram Social Network"-

[3] Lu Cheng,Jundong Li,Yasin N,Silva Deborah,L. Hall Huan Liu-"XBully: CyberbullyingDetection within a Multi-Modal Context"-2019

[4] Zhili Zhou · Ching-Nung Yang · Cheonshik Kim · Stelvio Cimato- "Introduction to the special issue on deep learningfor real-time information hidingand forensics" - 28 January 2020

[5] Hao Li-"Image analysis of cyberbullyingusing machine learningtechniques"-2016

[6] Belhassen Bayar and Matthew C. Stamm-"Design Principles of Convolutional Neural Networks for MultimediaForensics"-2017

[7] Shylaja S S, Abhishek Narayanan, Abhijith Venugopal, Abhishek Prasad- "Document Embedding Generation for Cyber-Aggressive Comment Detection using Supervised Machine LearningApproach"-2019

[8] Krishna B. Kansara and Narendra M. Shekokar- "A Framework for Cyberbullying Detection inSocial Network"-2019

[9] Thomas Deselaers and Lexi Pimenidis and Hermann Ney- "LLM and BOVW Models for AdultImage Classification andFiltering"-2013

[10] N. Mansoori, M. Nejati, P. Razzaghi and S. Samavi- "Image Retrieval by Bag of Visual Words andColor Information"-2013

APPENDIX A

DEFINITIONS, ACRONYMS AND ABBREVATION

Abbreviations:

SIFT - Scale Invariant Feature Transform, the scale-invariant feature transform is a feature detectionalgorithm in computer vision to detect and describe local features in images.

BOVW- Bag of Visual Words, the bag-of-words model sometimes called bag-of-visual-wordsmodelcan be applied to image classification, by treating image features as words.

SVM- Support Vector Machine, a machine learning model.

CNN- Convolutional Neural Networks, a deep learning algorithm.