SENTIMENT ANALYSIS FROM SOCIAL MEDIA IMAGE USING CNN

PRESENTED BY

MD. RASEL KHONDOKAR
STUDENT ID: 163-15-8383

AND

MD.MONJURUL ISLAM
STUDENT ID: 163-15-8487

The Thesis Report is Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering.

Supervised By
Syeda Tanjila Atik
Lecturer Department of CSE
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
DHAKA, BANGLADESH
September, 2019

APPROVAL

This Project/internship "**Sentiment analysis from social media image using CNN**", submitted by Md. Rasel Khondokar Student ID: 163-15-8383 and Md. Monjurul Islam Student ID: 163-15-8487 to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 13 September 2019.

BOARD OF EXAMINERS

DECLARATION

We hereby state that, under the guidance of Syeda Tanjila Atik, lecturer, CSE Department, Daffodil International University, we have accomplished this dissertation job named "Sentiment analysis from social media image using CNN" We also state that the work described in this study was produced by us as a consequence of our own original research and before the date of submission, neither the project nor any portion of this project was submitted elsewhere for award of any degree or diploma. We recognize other people's published and unpublished works from which we collected adequate understanding to derive the thesis.

Submitted By

[illegible]

Md. Rasel Khondokar

Student ID: 163-15-8383

Department of Computer Science and Engineering

Daffodil International University

[illegible]

Md.Monjurul Islam

Student ID: 163-15-8487

Department of Computer Science and Engineering

Daffodil International University

Supervised by

[illegible] 3/9/19

Syeda Tanjila Atik

Lecturer

Department of Computer Science and Engineering

Daffodil International University

ACKNOWLEDGEMENT

First, we convey our heartfelt gratitude to the all-powerful God for His blessed gifts that enable us to effectively finish the initial year thesis job. We are so thankful to our renowned manager and supervisor. Syeda Tanjila Atik, lecturer, CSE Department, Daffodil International University, for our deep indebtedness. Her profound understanding and passionate concern in Artificial Intelligence and Machine Learning assist us do this thesis. Her endless persistence, academic guidance, continuous motivation, continuous and vigorous oversight, constructive criticism, useful advice, studying many superior drafts and correcting them at all phases have created it possible for him to do so.

ABSTRACT

Images are used on social media to convey views, opinions, feelings, and emotions. We research the issue of knowing natural feelings from large-scale pictures of social media in this work. We need to understand public sentiment for Social marketing, to develop product quality, to improve customer service, social media monitoring and research purpose. In this work we have classified happy, sad and angry emotions of people by using their face image on social media. For the classification of pictures, we have used Convolutional Neural Networks (CNN). First, we modified an appropriate CNN architecture for the assessment of picture sensitivity. The findings indicate that in picture sentiment analysis, the suggested CNN can attain stronger efficiency than rival algorithms. By using this model we can classify image into three categories of sentiment named happy, sad and angry and we got an accuracy of 75.28%. This work can be further enhanced to be used in many other related areas of image classification.

TABLE OF CONTENTS

DECLARATION III

ACKNOWLEDGEMENT IV

ABSTRACT V

LIST OF FIGURES VIII

LIST OF TABLES IX

CHAPTER 1 : INTRODUCTION 1

1.1 Introduction 1

1.2 Motivation 1

1.3 Research Questions 2

CHAPTER 2 : BACKGROUND 3

2.1 Introduction 3

2.2 Related Research Works 9

2.3 Research Summary 10

2.4 Scope of the Problem 10

2.5 Challenges 11

CHAPTER 3 : RESEARCH METHODOLOGY 12

3.1 Convolutional Neural Networks 12

3.2 Experiment 13

3.3 Sequential Mode 13

3.4 Convolution layer 13

3.5 Max pooling 14

3.6 Flattening 14

3.7 Dense 14

3.8 Fully Connected Layer 14

3.9 Compiling the model 14

3.10 Image Augmentation for Deep Learning With Keras 15

CHAPTER 4 : EXPERIMENTAL RESULTS 16

4.1 Experimental Results 16

4.2 Summary 19

CHAPTER 5 : CONCLUSION AND FUTURE IMPLICATIONS 20

5.1 Conclusion 20

5.2 Real Life Applications 20

5.3 Future works 20

REFERENCES 21

APPENDIX A 22

Data Set Preparation and Work Flow in Rapid Miner 22

APPENDIX B 23

Python Code 23

LIST OF FIGURES

Figure 1. 1 : Example Photos For The Reaction Of People About An Event. 1
Figure 2. 1 : Real Image Vs Computer Readable Image. 3
Figure 2. 2 : Input Image. 4
Figure 2. 3: Different Types Of Filters To Detect Different Features 5
Figure 2. 4 : Filter/Kernel With Different Sizes 5
Figure 2. 5 : Filter/Kernel 6
Figure 2. 6 : Feature Mapping 6
Figure 2. 7 : Relu(Rectified Linear Unit):Introduces Non-Linearity 7
Figure 2. 8: Rectified Linear Unit Function. 7
Figure 2. 9: Left **:** Before Relu Applied. Right : After Relu Applied 8
Figure 2. 10 : Max Pooling On Relu Feature Map 8
Figure 2. 11 : Cnn Architecture. 9
Figure 3. 1 : Process In Cnn 12
Figure 3. 2 :Cnn Algorithm 12
Figure 4. 1: Our Cnn Model/Algorithm 16
Figure 4. 2 : Visualize Feature Maps Output 17
Figure 4. 3: Visualizing Accuracy Of Train And Test 18
Figure 4. 4 : Visualizing Loss Of Train And Test 18

LIST OF TABLES

Table 3. 1: Dataset Details	13
Table 4. 1: Accuracy And Loss Of Train And Test On Every Epoch	17
Table 4. 2 : Model Summary	19

CHAPTER 1 : INTRODUCTION

1.1 Introduction

From a social media picture, you can determine how individuals feel about your item. People exchange a ton of information on social media in the type of pictures Analyzing information such as this from social media pages and/or photo sharing platforms such as Facebook, Instagram, Flickr, Twitter, Tumblr, etc. can provide perspectives into people's overall feelings about such elections, movies, events, products, etc. It would be helpful to know the emotion that an picture portrays in order to forecast mental marks on them automatically-like joy, sorrow, wrath, dread, etc. We are analyzing a picture in this document that fell into three classifications- Happy, Angry, and Sad. We do this by fine-tuning three distinct convolutionary.

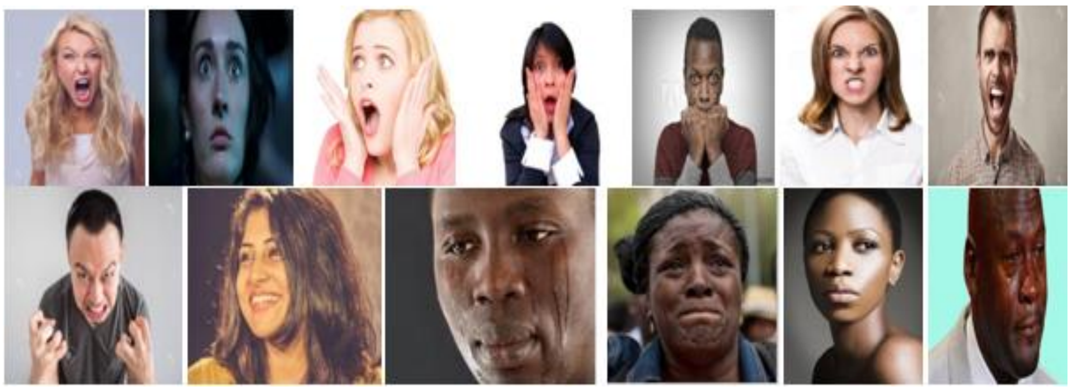


Figure 1. 1 : Example Photos for the reaction of people about an event.

1.2 Motivation

Analysis of facial expression was fundamentally a psychologist study subject. Recent progress in image processing and pattern recognition has significantly driven studies on instant identification of facial expression. There has been a great deal of effort in the past to recognize facial expression in still images. Many methods have been used for this intent. A very significant restriction to this approach is the reality that still pictures generally catch the expression's apex, i.e. the moment when the emotion indices are most visible. People rarely demonstrate their facial expression in their regular lives during ordinary communication with their colleagues. Previously some thesis done about sentiment analysis but most often they are not use deep learning. That’s why when we have to work with more data then these will not work efficiently.

1.3 Research Questions

As of the research project focuses on feature selection and classification of images.

Q Why sentiment analysis ?

Q Can the sentiment be classified with convincing level of accuracy ?

Q Which techniques of ML approaches would be good for sentiment ?

CHAPTER 2 : BACKGROUND

2.1 Introduction

A Convolutional Neural Network (CNN) is a Deep Learning algorithm that is capable of capturing an entry picture, assigning significance (learnable weights and biases) to different aspects in the picture and distinguishing them from each other. In a CNN, pre-processing is much smaller than other ranking methods. While filters are hand-crafted in basic techniques, CNN has the capacity to know these filters / characteristics with sufficient practice.

A CNN architecture is similar to that of the human brain connection model of Neurons and was motivated by the Visual Cortex's organization. Individual neurons react to stimuli.

What are Convolutional Neural Networks?

Convolutional neural networks ingest and store pictures as tensors, as opposed to neural networks, and tensors are matrices of figures with extra sizes.

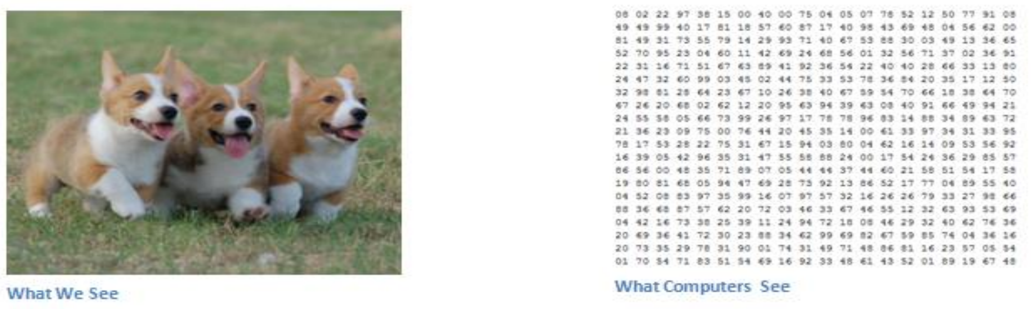


Figure 2. 1 : Real image vs computer readable image.

Let's claim we've got a JPG color picture and it's 480x 480 size. 480x 480x 3 (channels= RGB) will be the preferred range. Each of these figures has a significance of 0 to 255 that defines the strength of the pixels at that stage. These figures are the only outputs accessible to the machine, although they are irrelevant to us when we conduct image classification.

Convolutional Neural Networks have 2 main components.

Feature learning : You can see the stages of convolution, ReLU, pooling sheet here. Edges, shades, lines, curves are obtained in this teaching phase.

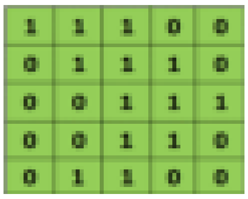
Classification :In this stage, you will see Fully Connected(FC) coating. They will attribute a probability that what the algorithm predicts is for the item on the picture.

Feature learning

Convolution :-

input image:

Each picture can be regarded as a matrix of pixel attributes, as mentioned above. Consider a 5 x 5 image whose pixel values are only 0 and 1 (note that the green matrix below is a special case where pixel values are only 0 and 1) for a gray image, pixel values ranging from 0 to 255:

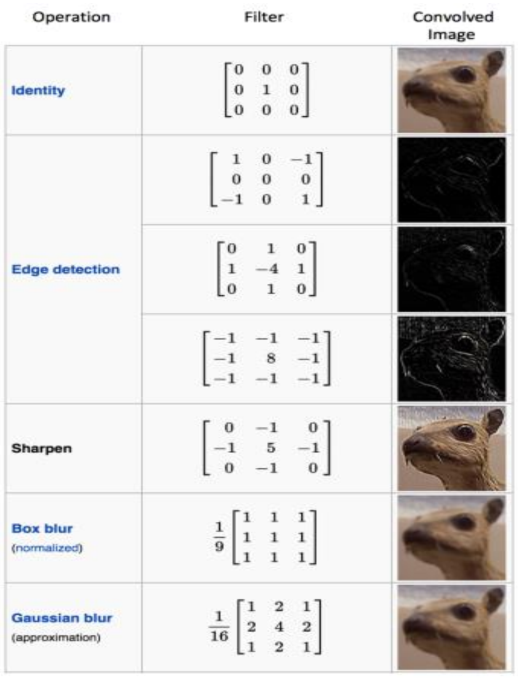


1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Figure 2. 2 : Input image.

Filter:

To get the Convolved coating, this entry picture is increased by a filter. These filters differ in forms and sizes to obtain various characteristics such as corners, angles, rows. Sometimes this filter was called a kernel, a feature detector.



Operation	Filter	Convolved Image
Identity	$\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}$	
Edge detection	$\begin{bmatrix}1&0&-1\\0&0&0\\-1&0&1\end{bmatrix}$	
	$\begin{bmatrix}0&1&0\\1&-4&1\\0&1&0\end{bmatrix}$	
	$\begin{bmatrix}-1&-1&-1\\-1&8&-1\\-1&-1&-1\end{bmatrix}$	
Sharpen	$\begin{bmatrix}0&-1&0\\-1&5&-1\\0&-1&0\end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix}$	

Figure 2. 3: Different types of filters to detect different features

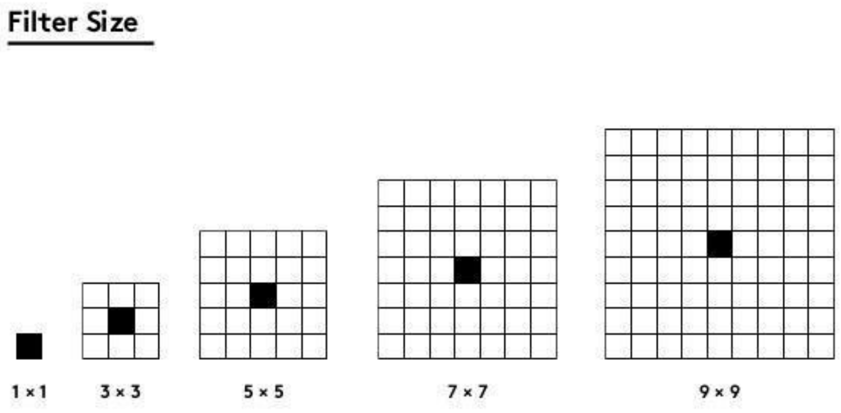
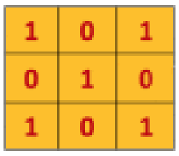


Figure 2. 4 : Filter/kernel with different sizes

lets consider 3*3 filter to extract some features from the input image which is in pixel values at input image above.

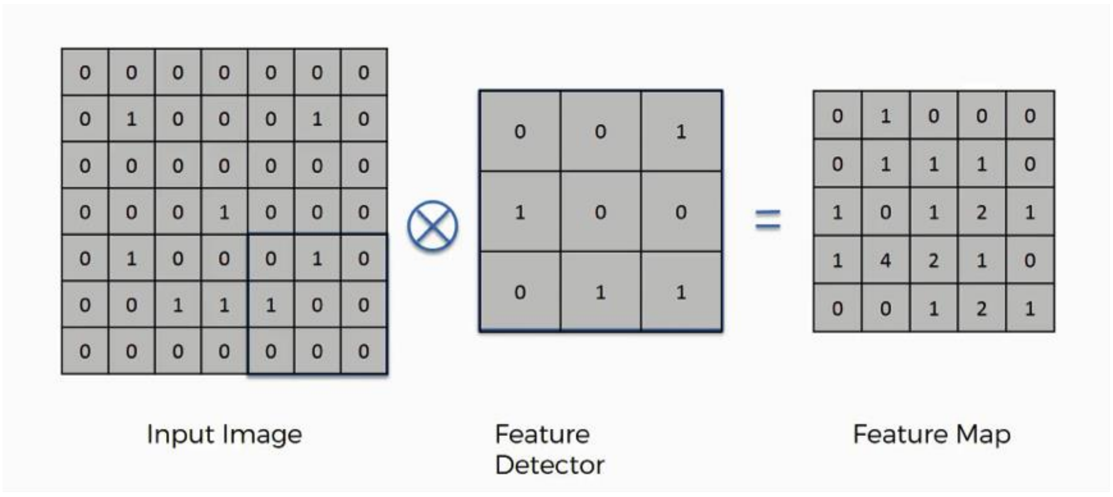


1	0	1
0	1	0
1	0	1

Figure 2. 5 : Filter/kernel

Convolved layer:

Each price in each column of the output picture is measured by the corresponding price and color of the filter. Sometimes this Conv Layer is called here as a Convolutional Feature, Feature Map, Filter Map.



0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

0	0	1
1	0	0
0	1	1

0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

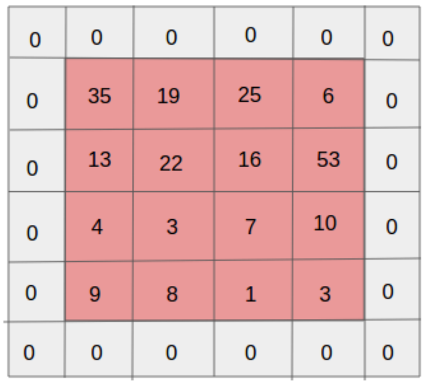
Figure 2. 6 : Feature mapping

Convolved feature / feature chart obtained by adding the filter / kernel to the original picture depending on the three parameters. We need to identify this before the convolution takes place.

Depth: In the picture above we regarded range = 1, but most of the outputs are range =3 which is 3 lines like Red, Green, Blue.

Stride: Stride is the amount of pixels we roll over the entry matrix with our filter matrix. When the step is 1 we migrate one pixel at a moment of the filters. When the step is 2, then as we move them around, the filters shoot 2 pixels at a moment. With a bigger step, narrower function charts will be produced.

Padding: It's like how many additional pixels you contribute to the picture. The padding would mostly be 1. Look at the box underneath. 4 * 4 panel filled picture 1.



0	0	0	0	0	0
0	35	19	25	6	0
0	13	22	16	53	0
0	4	3	7	10	0
0	9	8	1	3	0
0	0	0	0	0	0

Figure 2. 7 : ReLU(Rectified Linear Unit):introduces non-linearity

After each Convolution operation, an extra procedure called ReLU was used. The Relu procedure is non-linear.

Output = Max(zero, Input)

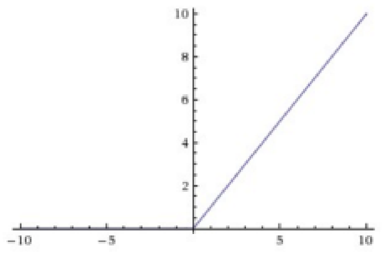


Figure 2. 8: Rectified Linear Unit function.

ReLU is an attribute based (implemented per sample) procedure and substitutes all adverse pixel numbers in the function chart with null. ReLU's purpose is to introduce non-linearity in our CNN, as most of the real-world data that we would like our CNN to learn would be non-linear (Convolution is a linear operation— element wise matrix multiplication and addition, so by introducing a non-linear function like ReLU we account for non-linearity).

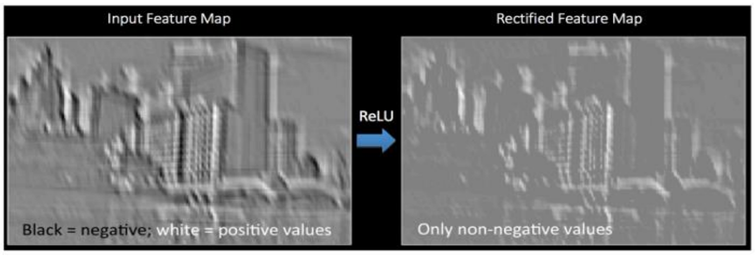


Figure 2. 9: Left **:** before relu applied. Right : after ReLU applied

Pooling layer:

The dimensionality of the convlayer or function chart is decreased in this stage, maintaining significant data. This temporal pooling is sometimes referred to as downsampling or sub-sampling. This can be Max pooling levels, Avg pooling, total pooling. Mostly we see the use of Max pooling.

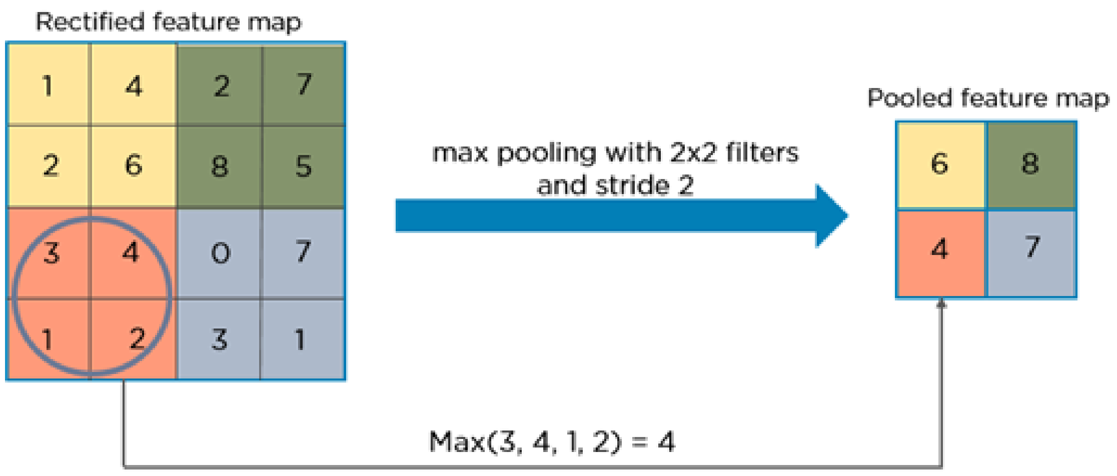


Figure 2. 10 : Max pooling on Relu feature map

Classification

Fully connected layer:

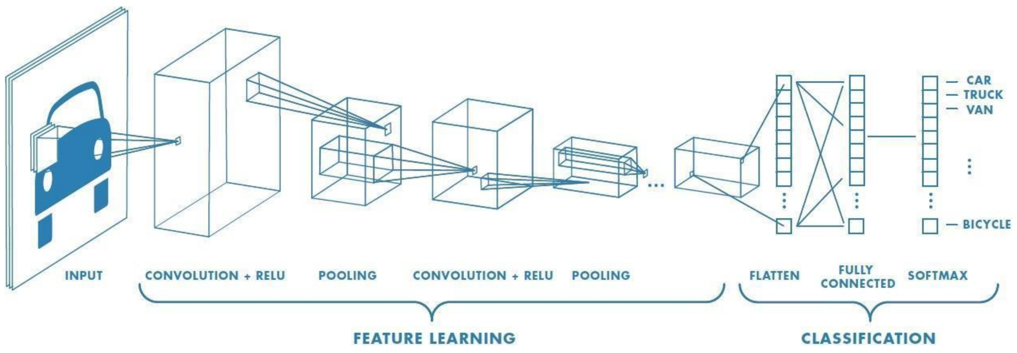


Figure 2. 11 : CNN Architecture.

The Fully Connected layer is a traditional multi-layer perceptron that utilizes the input layer's softmax activation function. The word "Fully Connected" means that the next layer connects each neuron in the prior cell to each neuron.

The yield from the convolutionary and pooling strata represents the original image's high-level characteristics. The Fully Connected layer's aim is to use these characteristics to classify the entry picture into different categories depending on the coaching dataset.

The amount of Fully Connected Layer's yield probabilities is 1. This is assured by using the Softmax in the Fully Connected Layer production cell as the activation function. The Softmax function requires an independent real-evaluated results vector and squashes it into a value vector between null and one that sums up to one.

In the picture above, using softmax to the fully linked coating provides categories like vehicle, truck, and bicycle the probability levels.

2.2 Related Research Works

Recently, a big amount of papers have emerged that discuss sentiment analysis and its relationship to image annotation, including:

Image Sentiment Analysis [1]. using a baseline sentiment algorithm to label Flickr images. conducted extensive experiments on manually labeled Twitter images.
Image sentiment analysis using deep convolutional neural networks with domain specific fine tuning [2].also use label Flickr images.
Visual Sentiment Analysis by Attending on Local Image Regions [3]. Use Flickr images. Use neural network to get the feature representation of dataset and classify the data by SVM classifiers. used VGG-ImageNet as the pretrained model. SentiBank: large-scale ontology and classifiers for detecting sentiment and emotions in visual content [4]. Visual sentiment topic model based microblog image sentiment analysis [5]. Firstly, obtain the visual sentiment features by using Visual Sentiment Ontology (VSO); then, build a Visual Sentiment Topic Model by using all images in the same topic; finally, choose better visual sentiment features according to the visual sentiment features distribution in a topic.

2.3 Research Summary

There has been a small amount of studies in this sector, according to previous literature research and study. The studies were quite effective in their own manner. This type of classification problem already done. The accuracy of the model is quite good for real data. The sentiment analysis from image is very useful thing. Can extract views, opinions, feelings, and emotions from images. It very useful for marketing, to develop quality of a product, improve customer service, social media monitoring and research purpose. In this work we extract features from pixel value of a face image. After taken features model will learn by train data and then predict for a new image which is happy, sad or angry. Then check it correct prediction or not.

2.4 Scope of the Problem

We used various machine learning algorithms and the model's data set involves training and testing. To find out which picture is in which class, we attempt to search the similarities and dissimilarities between the images.

Nowadays most of time we see image sentiment analysis taken from text. Sometimes from image but does not use machine learning approach. Sentiment analysis from image using deep learning algorithm CNN is very efficient.

2.5 Challenges

We have been trying to gather images on facial expression about an event or product from the start for this thesis. But there are not enough pictures of a particular case or product. So we found it very hard to collect information. But we are continuing the process of information collection. It was not always suitable to use the picture collected for this purpose.

Our CPU takes more hours for train and test from image data. So we use TPU from google colaboratory. But there is problem in colaboratory, uploaded files will be deleted automatically after 12 hours.

It takes lots of time to re-run the code with a slight change. So that little changes are need more times.

CHAPTER 3 : RESEARCH METHODOLOGY

3.1 Convolutional Neural Networks

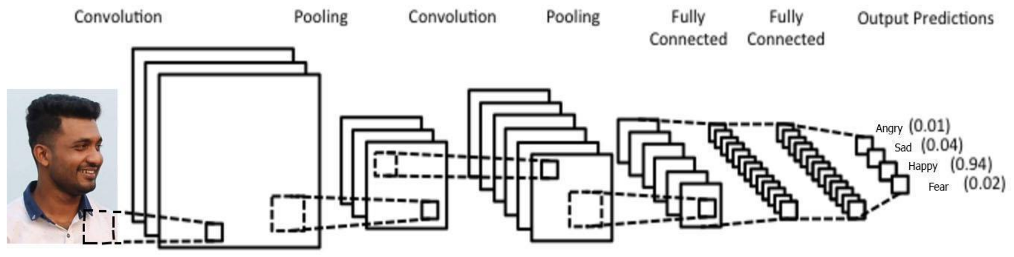
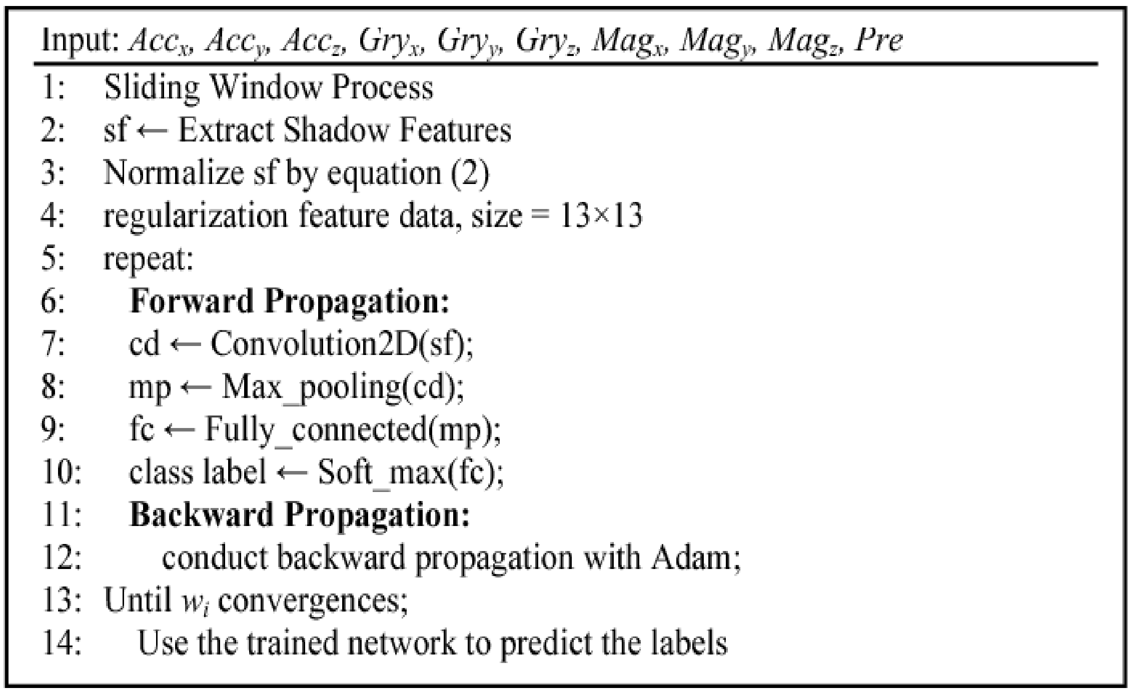


Figure 3. 1 : Process in CNN



```
Input: $Acc_x$, $Acc_y$, $Acc_z$, $Gry_x$, $Gry_y$, $Gry_z$, $Mag_x$, $Mag_y$, $Mag_z$, $Pre$
1:  Sliding Window Process
2:  sf ← Extract Shadow Features
3:  Normalize sf by equation (2)
4:  regularization feature data, size = 13×13
5:  repeat:
6:    Forward Propagation:
7:    cd ← Convolution2D(sf);
8:    mp ← Max_pooling(cd);
9:    fc ← Fully_connected(mp);
10:   class label ← Soft_max(fc);
11:   Backward Propagation:
12:     conduct backward propagation with Adam;
13: Until $w_i$ convergences;
14:   Use the trained network to predict the labels
```

Figure 3. 2 :CNN Algorithm

3.2 Experiment

We use Google, Facebook, Instagram to retrieve our information. It divided into instruction and testing. There are 2938 pictures available. We use CNN to forecast feelings. Convent operates by abstracting picture characteristics from detail to components of a greater stage. It is possible to describe an analogy with the manner people believe. Each of us knows how aircraft looks, but most likely we don't think about every little bit of aircraft composition when talking about aircraft. Similarly, the convent knows to acknowledge components of the greater standard in the picture, and this helps to identify fresh pictures when they appear comparable to those used for the practice.

Model is built using a subset of data:

Table 3. 1: Dataset details

Dataset	Image
Train	2700
Test	238
Total	2938

First model training attempt is done directly using available images from the dataset. Convent trains to identify sentiment using Keras and TensorFlow backend.

3.3 Sequential Mode

For most problems, the sequential API allows you to create layer-by-layer models. It is restricted in that it prevents you from creating designs that exchange levels or have various entrances or outputs. The Sequential Model API is good in most circumstances to develop deep learning designs.

3.4 Convolution layer

Our first layers are types of Conv2D. These are convolution levels that will cope with our two-dimensional matrices entry pictures. 32 The amount of nodes in each cell in the first phase. Depending on the dataset width, this amount can be adapted to be greater or smaller. The kernel magnitude for our convolution is the filter matrix length. A kernel volume of 3 implies we're going to have a matrix of 3x3 filters. For a refresher on this, see the start and the first picture. Activation is the layer's activation function. We will use the ReLU, or Rectified Linear Activation, to activate the feature. This activation function in neural networks has been shown to operate well. Also our first part requires the form of an entry. This is the form of each picture entry, 64, 64, 3 as seen previously, with the 1 being greyscale.

3.5 Max pooling

Pooling parts decrease the output's temporal magnitude by combining kernel characteristics with a vector of those attributes. You bring the total out of each pot (core) as the fresh valuation for that panel, for example, in super pooling.

3.6 Flattening

Flatten serves as a connection between the convolution and dense layers.

3.7 Dense

Dense is the layer type we will use in for our output layer. Dense is a standard layer type that is used in many cases for neural networks.

3.8 Fully Connected Layer

The layer we call as FC layer, we flattened our matrix into a vector and feed it into a fully connected layer like a neural network. Fully connected layerŁŁThe final output layer is a normal fully-connected neural network layer, which gives the output.

3.9 Compiling the model

We need our template to be compiled. The compilation of the model requires three parameters: optimizer, loss, and metrics. We're going to use Adam as our optimizer. In many instances, Adam is usually a nice optimizer to be used. Adam optimizer adjusts the teaching level throughout the course. The training level determines how quickly the model's optimum weights are calculated.A lower learning rate may lead to more accurate weights (up to a certain point), but for our loss function, the time it takes to calculate the weights will be longer. This is the classification's most popular option. A reduced rating shows the model performs better. We will use the precision metric to see the precision rating on the validation panel when we teach the model to create situations even simpler to understand.

3.10 Image Augmentation for Deep Learning With Keras

Working with neural networks and deep learning designs requires data preparing. Data augmen-tation is also increasingly needed for more complicated duties of item identification. The picture augmentation API, like the remainder of Keras, is easy and strong. Keras offers the category ImageDataGenerator which specifies the setup for preparing and increasing picture information. This involves the following capacities:

1) Rescale 2) Shear 3).Zoom 4) Flip

CHAPTER 4 : Experimental Results

In this chapter, we will discuss about the results of the guided experiment.

4.1 Experimental Results

This pattern was to be assumed as the model abstracts the features from the image into more particular notions that can be used to make a classification. Although the ultimate picture does not show that the model has seen a head, we usually miss the capacity to understand these greater function charts.

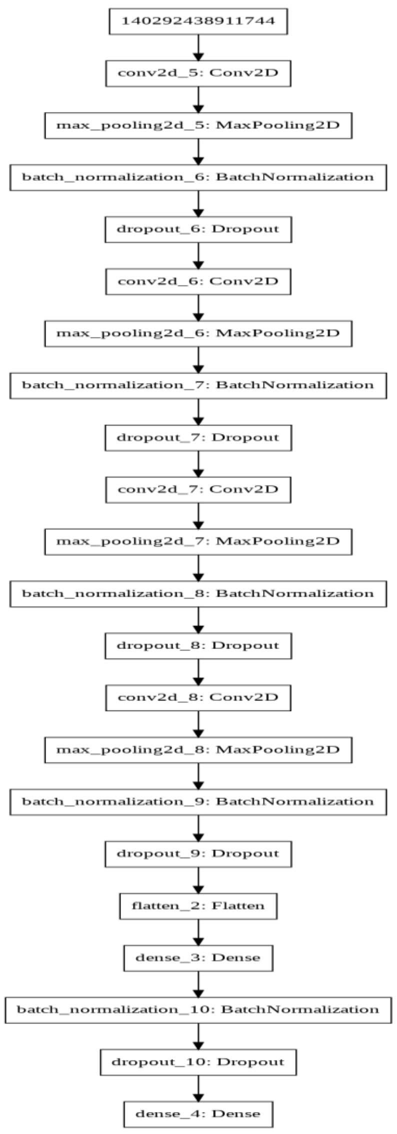

140292438911744
conv2d_5: Conv2D
max_pooling2d_5: MaxPooling2D
batch_normalization_6: BatchNormalization
dropout_6: Dropout
conv2d_6: Conv2D
max_pooling2d_6: MaxPooling2D
batch_normalization_7: BatchNormalization
dropout_7: Dropout
conv2d_7: Conv2D
max_pooling2d_7: MaxPooling2D
batch_normalization_8: BatchNormalization
dropout_8: Dropout
conv2d_8: Conv2D
max_pooling2d_8: MaxPooling2D
batch_normalization_9: BatchNormalization
dropout_9: Dropout
flatten_2: Flatten
dense_3: Dense
batch_normalization_10: BatchNormalization
dropout_10: Dropout
dense_4: Dense

Figure 4. 1: Our CNN model/algorithm

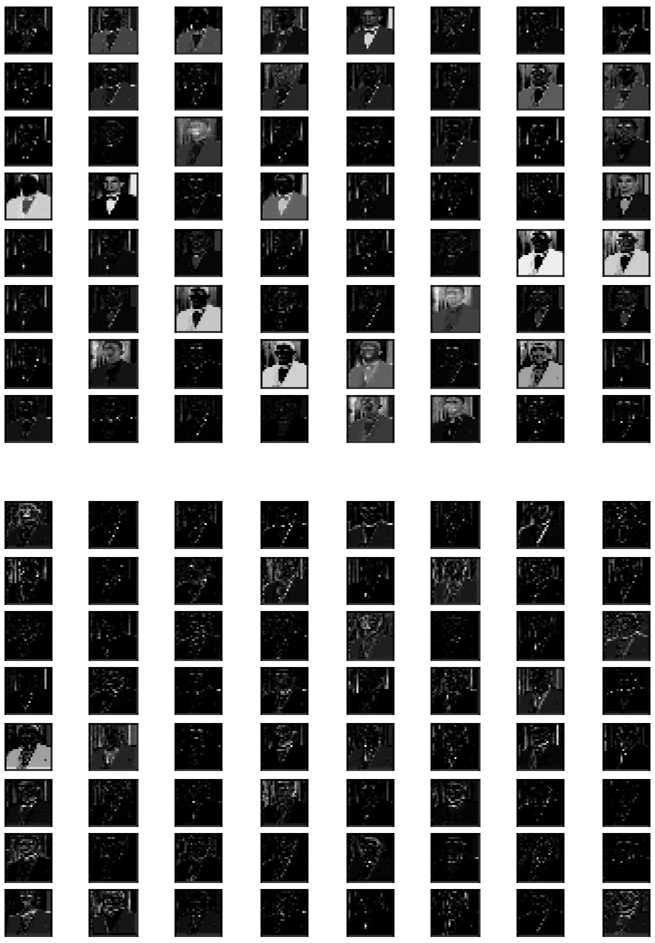


Figure 4. 2 : Visualize feature maps output

Accuracy and loss of each epoch

Final epoch accuracy : 0.752185

Table 4. 1: Accuracy and loss of train and test on every epoch

Epoch	Accuracy of test set	Accuracy of train set	Loss of test set	Loss of train set
1	0.458168	0.44865	0.998176	1.094363
2	0.478573	0.481389	0.953454	1.042238
3	0.555257	0.505294	0.925772	1.00928
4	0.562182	0.518528	0.921235	1.00117
5	0.512719	0.526998	0.933286	0.982559
.	.	.	.	.

.	.	.	.	.
.	.	.	.	.
86	0.713843	0.809952	0.765709	0.466617
87	0.718768	0.81415	0.749822	0.470072
88	0.730073	0.811673	0.71797	0.472068
89	0.72329	0.809158	0.713125	0.470331
90	0.752185	0.816834	0.690561	0.463895

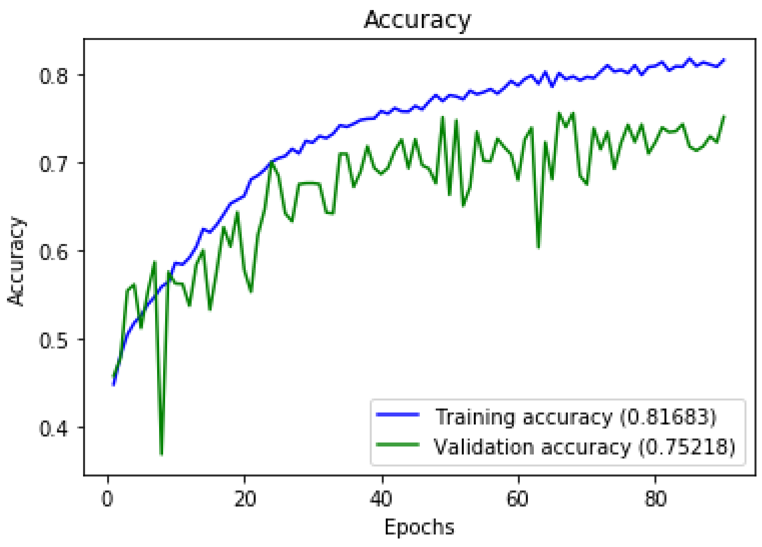


Figure 4. 3: Visualizing Accuracy of train and test

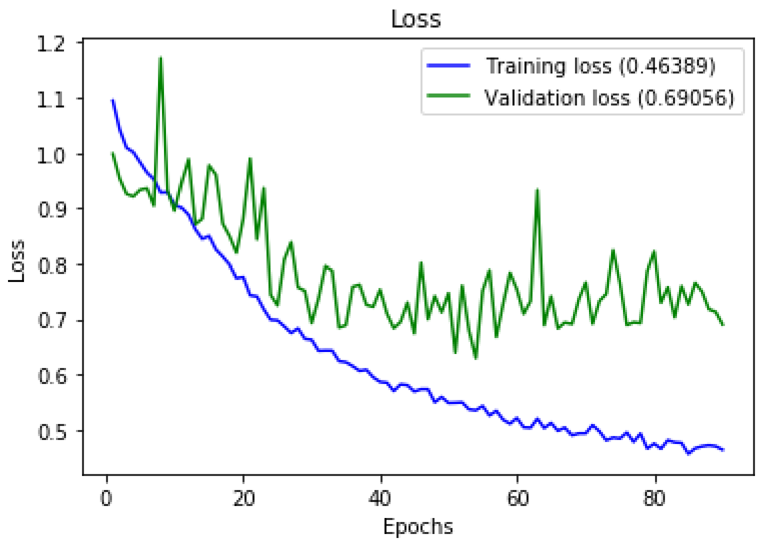


Figure 4. 4 : Visualizing loss of train and test

4.2 Summary

Table 4. 2 : Model summary

Layer (type)	**Output Shape**	**Param #**
conv2d_1 (Conv2D)	(None, 62, 62, 32)	896
batch_normalization_1	(Batch (None, 62, 62, 32)	128
dropout_1 (Dropout)	(None, 62, 62, 32)	0
max_pooling2d_1	(MaxPooling2 (None, 31, 31, 32)	0
conv2d_2 (Conv2D)	(None, 29, 29, 32)	9248
max_pooling2d_2	(MaxPooling2 (None, 14, 14, 32)	0
batch_normalization_2	(Batch (None, 14, 14, 32)	128
dropout_2 (Dropout)	(None, 14, 14, 32)	0
conv2d_3 (Conv2D)	(None, 12, 12, 32)	9248
max_pooling2d_3	(MaxPooling2 (None, 6, 6, 32)	0
batch_normalization_3	(Batch (None, 6, 6, 32)	128
dropout_3 (Dropout)	(None, 6, 6, 32)	0
conv2d_4 (Conv2D)	(None, 4, 4, 32)	9248
max_pooling2d_4	(MaxPooling2 (None, 2, 2, 32)	0
batch_normalization_4	(Batch (None, 2, 2, 32)	128
dropout_4 (Dropout)	(None, 2, 2, 32)	0
flatten_1 (Flatten)	(None, 128)	0
dense_1 (Dense)	(None, 128)	16512
batch_normalization_5	(Batch (None, 128)	512
dropout_5 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 3)	387
Total params: 46,563 Trainable params: 46,051 Non-trainable params: 512		

CHAPTER 5 : CONCLUSION AND FUTURE IMPLICATIONS

5.1 Conclusion

The work uses deep learning to classify emotions. Which efficiency is similar to other techniques that use handcrafted characteristics on the assignment of classifying emotions, and also some techniques that use deep learning to analyze feelings. The test findings indicate that more precise sensory characteristics in the assessment of sensory feelings will contribute to stronger performance. Using worldwide graphic characteristics, we use the fundamental and immediate fine tuning approach on CNN could contribute to similar CNN results. We expect that more research on sensory feeling assessment can be encouraged by our job on using local picture areas. In the future, we intend to integrate picture and sms with real-time forecast to build a more suitable and helpful model of sentiment analysis.

5.2 Real Life Applications

Analysis of sentiment is then used to determine communication efficiency and how well it was viewed. It can also be used to evaluate fulfillment with the client. Company A, for instance, may renew its item. The business can determine its clients by evaluating responses through feeling assessment. Medicine, e-learning, surveillance, marketing, entertainment and law. Measuring social media efficiency, government feeling for social media marketing, developing product quality, improving customer service, tracking social media and study purposes.

5.3 Future works

Our studies demonstrate that profound teaching Convolutionary Neural Networks (CNN) performs very well for anal-ysis of emotion, even on the fresh information straight gathered from Google. The next stage is to create an implementation in real time that will automatically forecast people's feelings about an case.

References

[1] Quanzeng You, Jiebo Luo, Hailin Jin, and Jianchao Yang, "Robust image sentiment analysis using progressively trained and domain transferred deep networks," *ACM*, January 2015.

[2] Stuti Jindal and Sanjay Singh, "Image sentiment analysis using deep convolutional neural networks with domain specific fine tuning," *IEEE*, June 2016.

[3] V Gajarla and A Gupta , "Emotion detection and sentiment analysis of images," *gatech*, 2015.

[4] D Borth , T Chen , R Ji , and SF Chang , "SentiBank: large-scale ontology and classifiers for detecting sentiment and emotions in visual content," *ACM*, October 2013.

[5] Donglin Cao , Rongrong ji , and Dazhen LinSh , "Visual sentiment topic model based microblog image sentiment analysis," *Springer*, 2016.

[6] Quanzeng You, Jiebo Luo, Hailin Jin, and Jianchao Yang, "Robust Image Sentiment Analysis Using Progressively Trained and Domain".

Appendix A

Data Set Preparation and Work Flow in Rapid Miner

Data collected from google, facebook, instagram, twitter posts.

Appendix B

Python Code

```
# start CNN
import tensorflow as tf
# pip install --upgrade keras
# Importing keras libraries and packages
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense, Dropout, BatchNormalization
from keras.callbacks import Callback
from keras import backend as K

# Initialising the CNN
classifier = Sequential()

# Step 1 - convolution
classifier.add(Convolution2D(32, (3, 3), input_shape = (64, 64, 3), activation = 'relu'))
# Step 2 - Pooling
classifier.add(MaxPooling2D(pool_size=(2, 2)))


classifier.add(BatchNormalization())
classifier.add(Dropout(0.25))


#  Adding a 2nd convolutional layer for better accuracy
classifier.add(Convolution2D(32, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))
classifier.add(BatchNormalization())
classifier.add(Dropout(0.25))

#  Adding a 3rd convolutional layer for better accuracy
classifier.add(Convolution2D(32, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))
classifier.add(BatchNormalization())
classifier.add(Dropout(0.25))

#  Adding a 4th convolutional layer for better accuracy
classifier.add(Convolution2D(32, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))
classifier.add(BatchNormalization())
classifier.add(Dropout(0.25))


# Step 3 - Flattening
classifier.add(Flatten())

# Step 4 - Full connection
classifier.add(Dense(128, activation='relu'))
classifier.add(BatchNormalization())
classifier.add(Dropout(0.5))
classifier.add(Dense(3, activation='sigmoid'))

# Compiling the CNN
classifier.compile(optimizer='adam',                    loss='categorical_crossentropy',
metrics=['accuracy'] )

# Fitting image to CNN
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(
        rescale=1./255,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True)

test_datagen = ImageDataGenerator(rescale=1./255)

train_set = train_datagen.flow_from_directory(
        'emotion (copy)/training',
        target_size=(64, 64),
        batch_size=32,
        class_mode='categorical')

test_set = test_datagen.flow_from_directory(
        'emotion (copy)/test',
        target_size=(64, 64),
        batch_size=32,
        class_mode='categorical')
no_of_epochs = 2
steps_in_each_epoch = 238
```

```
history = classifier.fit_generator(
        train_set,
        steps_per_epoch=steps_in_each_epoch,
        epochs= no_of_epochs,
        validation_data=test_set,
        validation_steps= steps_in_each_epoch)



print ("Accuracy and loss in each epoch")

import pandas as pd
acc_loss = pd.DataFrame({
     'Epoch' : [ i for i in range(1, 91)],
    'Accuracy of test set': history.history['val_acc'],
    'Accuracy of train set': history.history['acc'],
    'Loss of test set': history.history['val_loss'],
    'Loss of train set': history.history['loss'],

})
print(acc_loss)



def plot_history(history):
    loss_list = [s for s in history.history.keys() if 'loss' in s and 'val' not in s]
    val_loss_list = [s for s in history.history.keys() if 'loss' in s and 'val' in s]
    acc_list = [s for s in history.history.keys() if 'acc' in s and 'val' not in s]
    val_acc_list = [s for s in history.history.keys() if 'acc' in s and 'val' in s]

    if len(loss_list) == 0:
        print('Loss is missing in history')
        return

    ## As loss always exists
    epochs = range(1,len(history.history[loss_list[0]]) + 1)

    ## Loss
    plt.figure(1)
    for l in loss_list:
        plt.plot(epochs, history.history[l], 'b', label='Training loss (' + str(str(format(history.history[l][-1],'.5f'))+')'))
    for l in val_loss_list:
        plt.plot(epochs, history.history[l], 'g', label='Validation loss (' + str(str(format(history.history[l][-1],'.5f'))+')'))

    plt.title('Loss')
    plt.xlabel('Epochs')
    plt.ylabel('Loss')
    plt.legend()

    ## Accuracy
    plt.figure(2)
    for l in acc_list:
        plt.plot(epochs, history.history[l], 'b', label='Training accuracy (' + str(format(history.history[l][-1],'.5f'))+')')
    for l in val_acc_list:
        plt.plot(epochs, history.history[l], 'g', label='Validation accuracy (' + str(format(history.history[l][-1],'.5f'))+')')

    plt.title('Accuracy')
    plt.xlabel('Epochs')
    plt.ylabel('Accuracy')
    plt.legend()
    plt.show()



classifier.summary()

# visualize feature maps output from each block in the vgg model
from keras.applications.vgg16 import VGG16
from keras.applications.vgg16 import preprocess_input
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
from keras.models import Model
from matplotlib import pyplot
from numpy import expand_dims
# load the model
model = VGG16()
# redefine model to output right after the first hidden layer
```

```
ixs = [2, 5, 9, 13, 17]
outputs = [model.layers[i].output for i in ixs]
model = Model(inputs=model.inputs, outputs=outputs)
# load the image with the required shape
img = load_img('emotion (copy)/test/happy/images (100).jpg', target_size=(224, 224))
# convert the image to an array
img = img_to_array(img)
# expand dimensions so that it represents a single 'sample'
img = expand_dims(img, axis=0)
# prepare the image (e.g. scale pixel values for the vgg)
img = preprocess_input(img)
# get feature map for first hidden layer
feature_maps = model.predict(img)
# plot the output from each block
square = 8
for fmap in feature_maps:
	# plot all 64 maps in an 8x8 squares
	ix = 1
	for _ in range(square):
		for _ in range(square):
			# specify subplot and turn of axis
			ax = pyplot.subplot(square, square, ix)
			ax.set_xticks([])
			ax.set_yticks([])
			# plot filter channel in grayscale
			pyplot.imshow(fmap[0, :, :, ix-1], cmap='gray')
			ix += 1
	# show the figure
	pyplot.show()
```