MAJOR PROJECT REPORT

Live Class Monitoring System(Face Emotion Detection)

Submitted By:

Jasmeet Singh (20210102718)

Shri Prakash Yadav (00410107219)

Shivam Rai(00110107219)

Mentor

Prof Prakash Rao

Department of Computer Science Engineering,

Netaji Subhas University of Technology, East Campus

Geeta Colony, Delhi -110031

DECLARATION

We hereby declare that the work presented in this report entitled "**Live Class Monitoring System(Face Emotion Detection)**" in fulfillment of the requirement for the award of the degree Bachelor of Technology in Computer Science Engineering, submitted to Department of Computer Science Engineering, Netaji Subhas University of Technology, East Campus affiliated to Guru Gobind Singh Indraprastha University, New Delhi, is an authentic record of our own work carried out during our degree under the guidance of Prof . Prakash Rao.

SIGNATURE:

Jasmeet Singh

Shri Prakash Yadav

Shivam Rai

CERTIFICATE

This is to certify that the Project work for **"Live Class Monitoring System(Face Emotion Detection)"** submitted by Jasmeet Singh, Shri Prakash Yadav and Shivam Rai in fulfillment for the requirements of the award of Bachelor of Technology Degree in in Computer Science Engineering, submitted to Department of Computer Science Engineering, Netaji Subhas University of Technology, East Campus, New Delhi is an authentic work carried out by his/her under my supervision and guidance. To the best of my knowledge, the matter embodied in the project has not been submitted to any other University / Institute for the award of any degree

DATE: **Prof Prakash Rao**

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my supervisor **Prof Prakash Rao** for providing his invaluable guidance, comments and suggestions throughout the course of this project, without which this project may not have been completed before the deadline. We would also like to convey our deep regards to all the faculty members of the CSE department, who have bestowed their great effort and guidance at appropriate times, without which it would have been a herculean task to finish the project on time.

We would also like to thank our seniors and friends, for helping us with the project and allowing us to complete the task on time.

ABSTRACT

On the road to the digital world, human-computer interaction is becoming very important. Much research has been done in this area over the last decade. Facial expressions are an important feature of nonverbal communication and play an important role in human-computer interaction. This project introduces a facial expression recognition (FER) approach using a convolutional neural network (CNN). Created with CNN, this model can be used to detect facial expressions in real time. This system can be used to analyze emotions while a user is watching a movie trailer or video lecture other than that also for small children in the classroom attending class.

❖ CONTENTS

PROBLEM STATEMENT

- One of the many challenges is how to ensure quality learning for students.
- Digital platforms can overwhelm the physical classroom in terms of the quality of the content, but when it comes to understanding whether students can understand the content in a live classroom scenario, this is still an open challenge.
- In the physical classroom, during the lecture, the teacher can look at the face of the class to measure emotions and adjust the lecture accordingly, fast or slow.
- He can identify students who need special attention.
- The digital platform has limitations regarding physical monitoring, but it has the data and machine capabilities that make the work. It provides data in video, audio, and text formats that can be analyzed using deep learning algorithms.

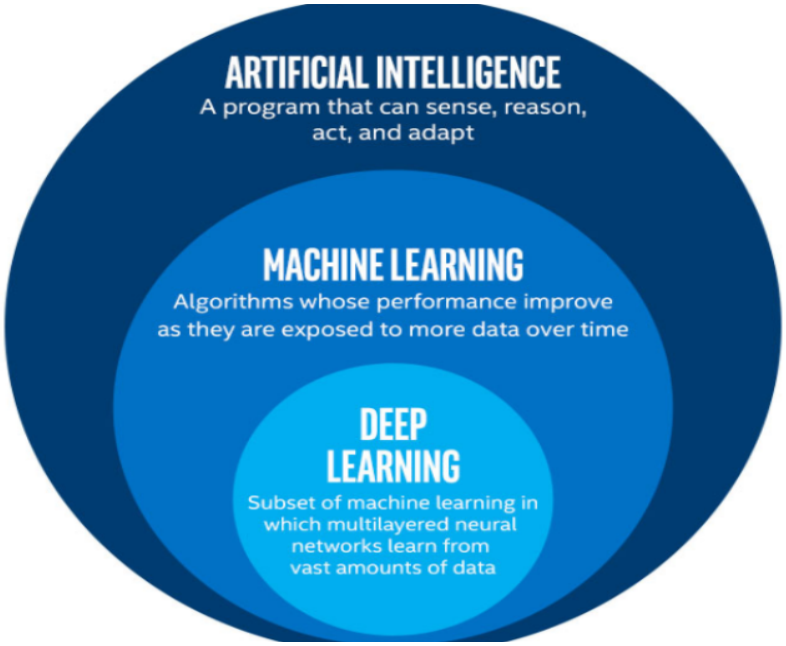
OBJECTIVE

- The objective of this project is to detect facial emotions in live feed using deep learning models.
- Fine tune the model for optimized results using different deep learning models and showcase advantages and disadvantages of each model that we implemented.
- Understand how deep learning and face emotion detection work efficiently together.
- To develop an interactive live face monitoring system to detect facial emotions.
- Model should be able to identify a student's emotions using minimum reference images.
- Model should work on the real-time webcam video feed.

Introduction to Deep Learning

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain—albeit far from matching its ability—allowing it to “learn” from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. Deep learning technology lies behind everyday products and services (such as digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies (such as self-driving cars).



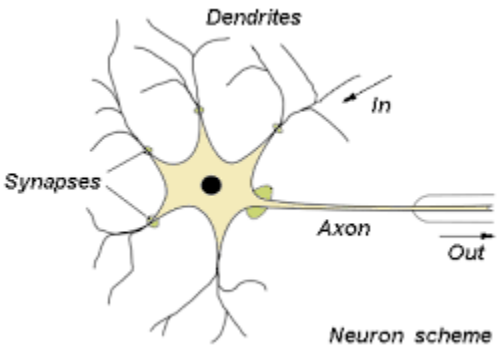
Neural Networks

Neural networks are robust and deep learning models capable of collecting large amounts of data in seconds. There are many different types of neural networks, and they help us in a variety of daily activities from recommending movies or music to helping us buy groceries online.

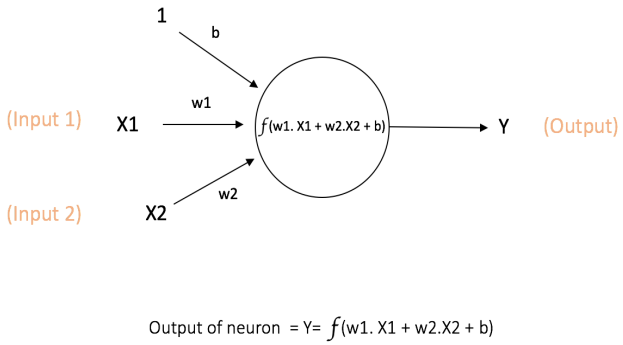
Similar to the way airplanes are stimulated by birds, neural networks (NNs) are stimulated by biological neural networks. Although the principles are the same, the process and structures can vary greatly. This is true of birds and aircraft as well as with biological sensory networks and in-depth sensory learning networks.

Neuron

The central part is called the cell body, where the nucleus resides. Various connections pass the stimulus to the cell body, called dendrites, and a few connections send the output to the other neurons called axons. The thickness of the dendrites and axons implies the power of the stimulus. Many neurons with various cell bodies are stacked up and form a biological neural network.



Biological Neuron



Neuron

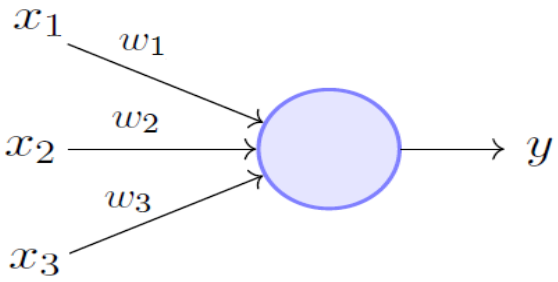
This same structure is visible in deep learning neural networks. The input is passed through an activation function (similar to the nucleus) with weighted edges (similar to dendrites). The generated output can be passed to another activation function. Many of these activation functions can be stacked up, and each of these is called a layer. Apart from the input layer and the output layer, there are many layers in the interiors of a neural network, and these are called hidden layers.

Types of Neural Networks

Perceptron:-

Perceptron is the simplest neural network structure. This model, which is also known as a single-layer neural network, contains only two layers:

- The Input Layer
- The Output Layer



There are no hidden layers here. Perceptron takes input and calculates the weighted input for each input node. This weighted input is passed through an activation function to generate the output.Due to the simple architecture, it cannot be used for complex tasks. This network is instead used for Logic Gates like AND, OR, or XOR.

- Applications of perceptron:-

Perceptrons are used in linear or binary model classification. They are also used in the formation of multilayer perceptrons, which we'll look at next.

Multi-layer Perceptron(MLP)

Multilayer perceptrons (MLPs), or feedforward neural networks, usually mean fully connected networks. In other words, each neuron in one layer is connected to all neurons in the adjacent layers. Hence, an MLP has higher processing power than a perceptron. However, the "fully-connectedness" of these networks makes them prone to overfitting data. Typical ways to reduce overfitting include early stopping, adding dropout layers, and adding regularization terms.

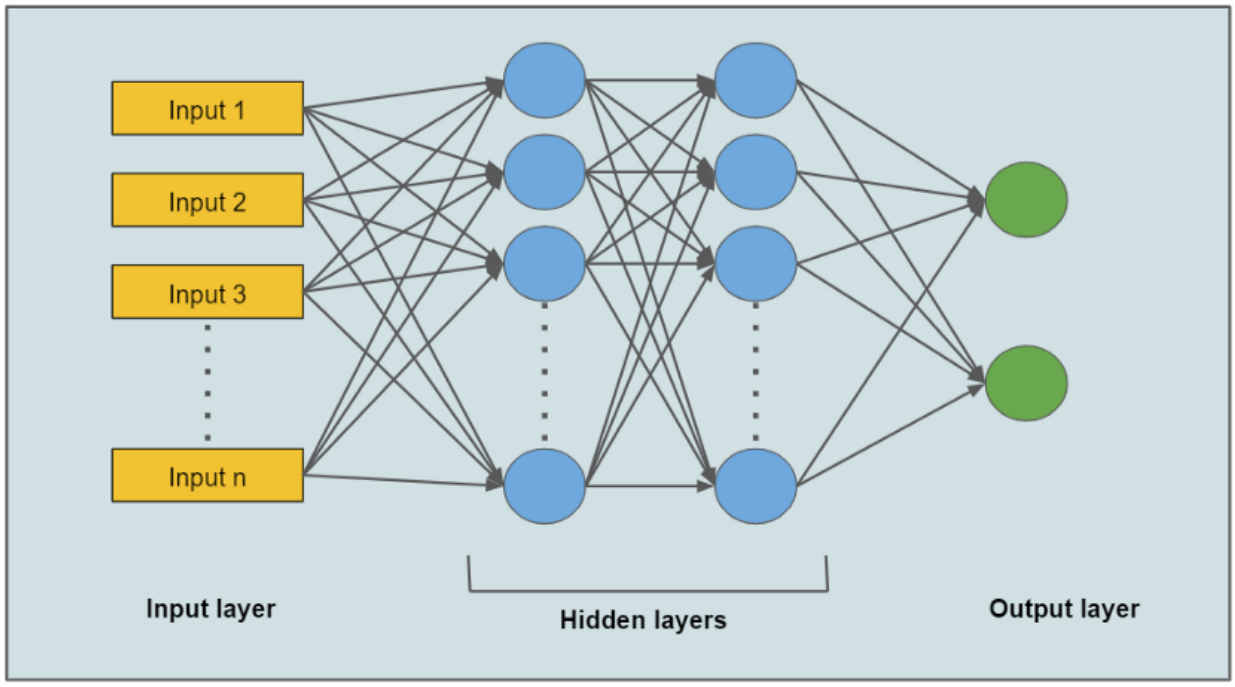


Figure 3: Architecture of Multi-layered perceptron (Image by author)

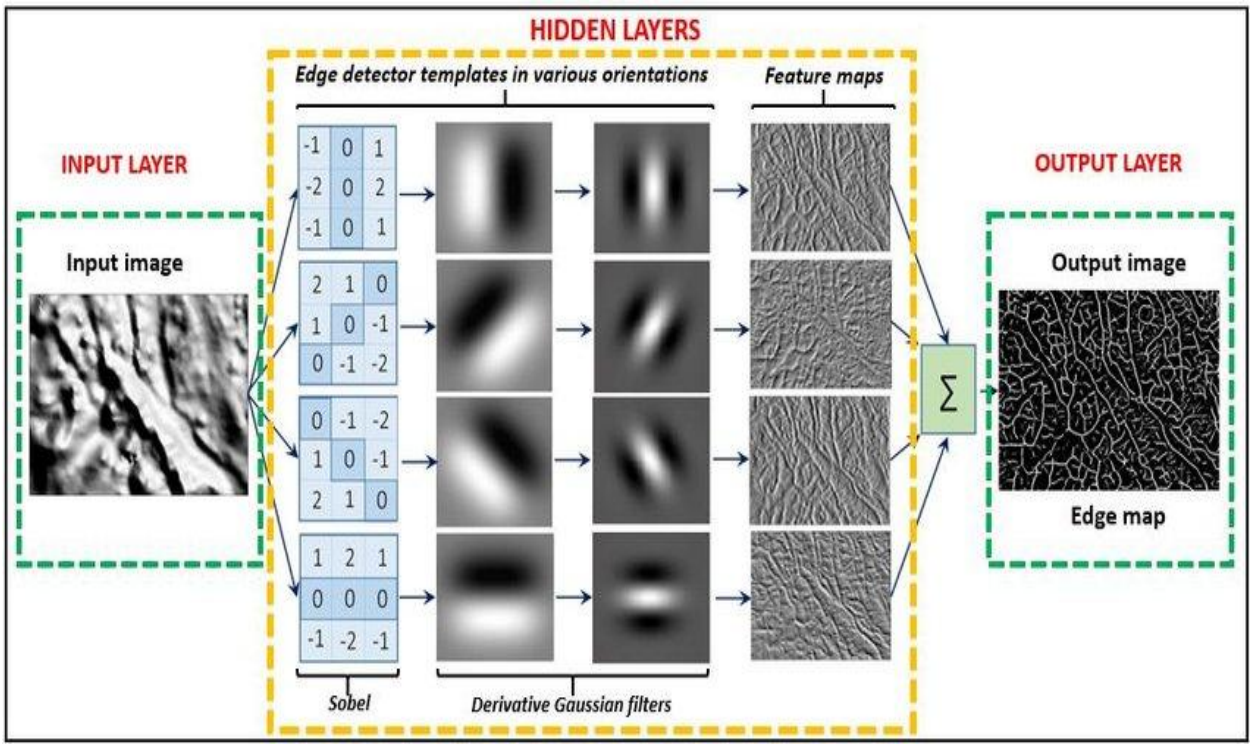
Applications of MLP

MLPs are widely used in a variety of areas. They're common in data compression for social networks, speech recognition and hand-written character recognition systems, computer vision applications, and data prediction systems.

Convolutional Neural Networks

Humans identify objects using neurons in the eyes which detect edges, shapes, depth, and motion. One of the most important types of neural networks in computer vision, convolutional neural networks (CNNs) are inspired by the visual cortex of the eyes and are used for visual tasks like object detection. The convolution layer of a CNN is what sets it apart from other neural networks. This layer performs dot product, which is component-wise multiplication followed by addition.

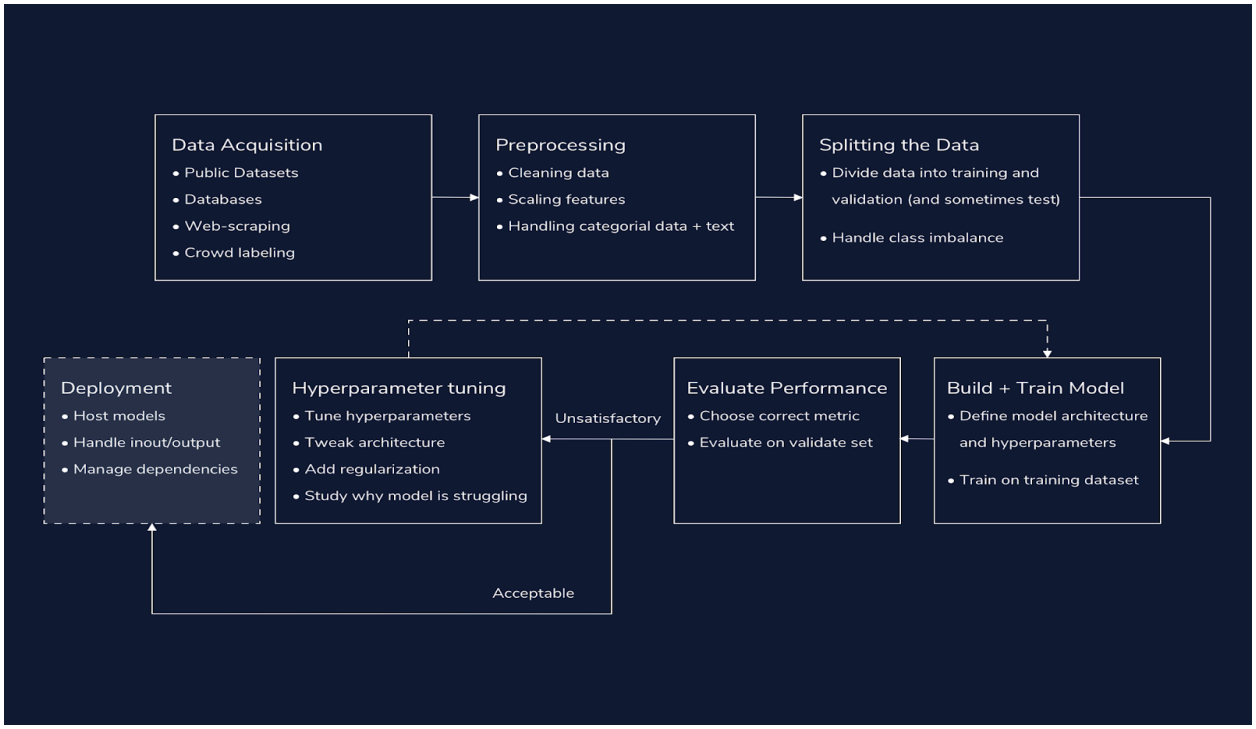
In the initial phases of a CNN, the filters are randomized and do not provide any useful results. Using a loss function, the filters are adjusted and over multiple iterations, the network gets better at achieving its task, such as detecting object edges, for example. Though they often require a large amount of training data, CNNs are widely applicable to a wide range of image and even language tasks.



Applications OF CNN

Because CNNs were inspired by the visual cortex, they are widely used for applications that involve the application of computer vision. These applications include facial recognition, face detection, object recognition, handwritten letter recognition, and the detection of tumors in medical diagnosis.

Workflow Of Our Model



We integrate the workflow of deep learning projects: how we build deep learning solutions to address real-world activities. The flow of in-deep learning activity has seven main components:

- Acquiring data
- Preprocessing
- Splitting and balancing the dataset
- Building and training the model
- Evaluation
- Hyperparameter tuning

Data Accumulation:-

In an in-deep learning study project, the most important concern is almost always data : "Can we get enough labeled data?" If we have more labeled data, our model could be better. Our ability to access data can make or break our solution. Not only is data acquisition often the most important part of an in-deep learning project, but it is also often the most difficult.

There are various sources where datas are available publicly

- Kaggle
- Web scraping/APIs
- Crowd-sourced Labeling

In our project we downloaded the FER 2013 dataset from Kaggle.

Pre-processing:-

Once we downloaded a dataset, we needed to process it in advance into useful features for our in-depth learning models. At the highest level, we have three main objectives when processing data for neural networks, we want to:

- Clean up our data
- Manage class and textual features
- Evaluate our true value using familiar or measuring techniques.

Data Augmentation:-

Data augmentation is a method which is used when our dataset is very small.It is used for image datasets. Our dataset has 35000 images which is very small number when neuran networks are concerned. In data

augmentation we perform shear, horizontal flip, vertical flip, etc to increase number of images.

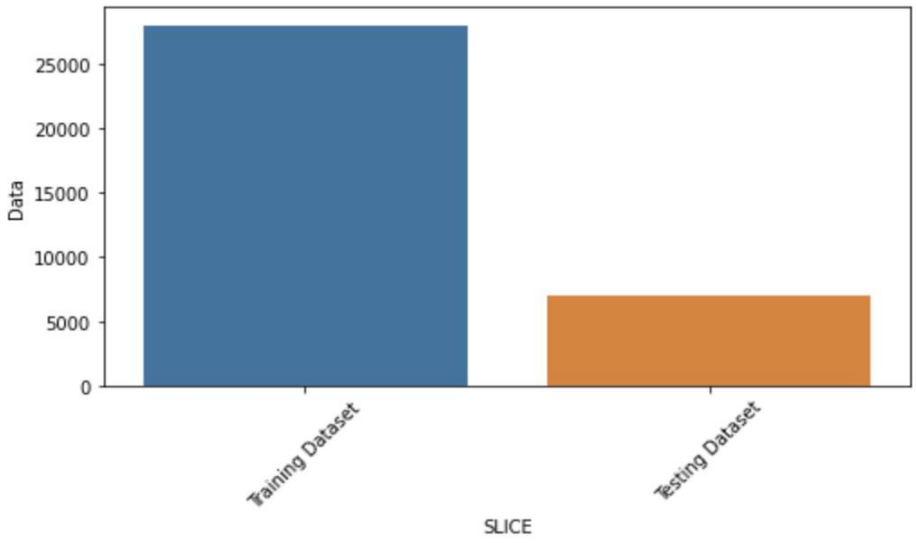
This is how data augmentation works.

Dividing and Measuring Data Set:-

Once we have processed our data, it is time to split our data. Generally, we divide our data into two data sets: training and validation. In some cases, we also create a third capture database, called a test set. When we do not do this, we often use the words “verification” and “test” interchangeably.

We train our model in the training database and test it in the validation database. When we define the third set of capture tests, we test our model in this data after we have finished selecting our model and adjusting our parameters. This third step helps us to avoid choosing a set of parameters that work only for the data we have selected for our verification set.

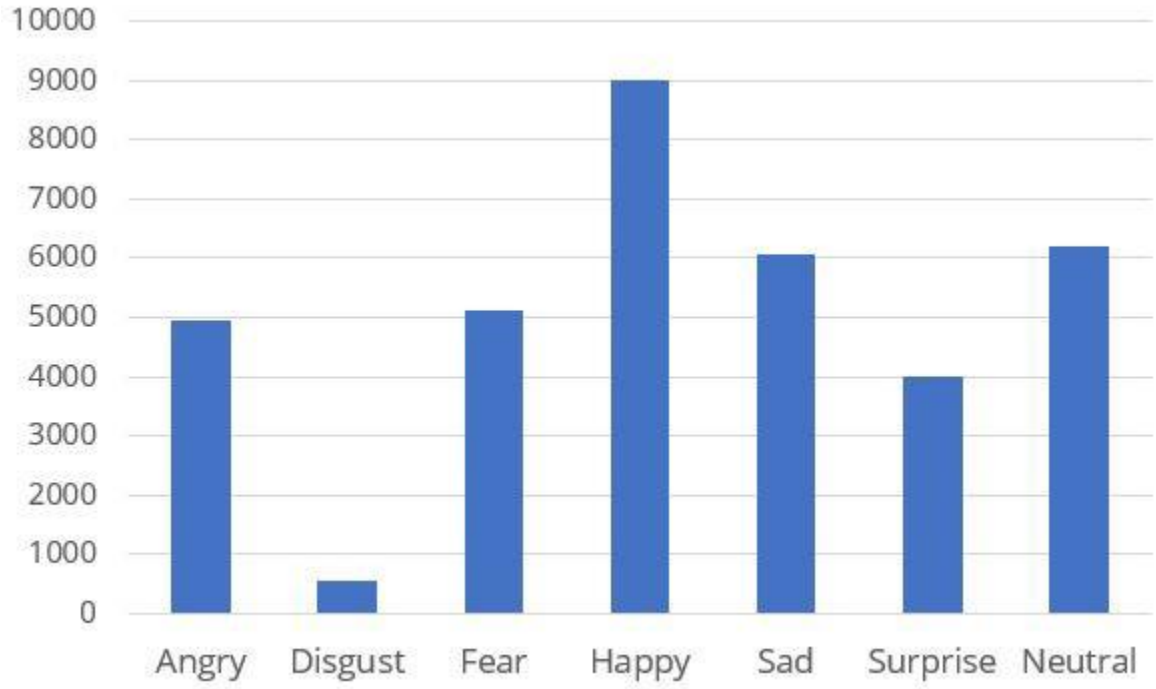
When we divide our data, there are two major considerations: the size of our division, and whether we will classify our data. After splitting our data, we need to address the inequalities in our training set.
In our project we have downloaded our dataset into two parts: training and validation dataset.
Training dataset has 28000 images and testing dataset has 7000 images.



Training Dataset Distribution

We must be very careful when dividing a very unequal data in order to be fragmented; it is very likely that many of our small class situations end up in training or in the certification set. In the first case, our verification metrics will not accurately capture our model's ability to distinguish a small group of people. In the second case, the model will exceed the chances of a multi-stage.

When we set the stratify parameter for a train_test_split activity to our list of labels, the task will cover a portion of each class, and ensure that this rate is consistent with our training and verification data.



Model Creation

Once we have segmented our data, it is time to choose our loss function, as well as our layers.

For each layer, we also need to select a reasonable number of hidden units. There is no perfect science for choosing the right size for each layer, or the number of layers - it all depends on your specific data and properties.

- It is a good practice to start with a few layers (2-6).
- Typically, we build each layer with between 32 and 512 hidden units.
- And we tend to reduce the size of the hidden layers as we go up - by model.
- We usually try SGD and Adam preparations first and we could also used different built in neural models like alex net, vgg , deep CNN etc.
- When you set the first reading level, the default default rate is 0.01.

Model Evaluation

Each time we train a model, we evaluate its effectiveness in our certification set. If we provide a confirmation set during training, Keras handles this automatically. Our performance on the verification set gives us an idea of how our model will work on new, invisible data.

When considering performance, it is important to choose the right metrics. In our project we have used accuracy as our evaluation metrics.

Tuning Hyperparameters:-

We will almost always need to multiply on our first hyperparameter. When we train and evaluate our model, we assess different learning values, group sizes, structures, and familiarity.

As we tune our parameters, we should look at our losses and metrics, and look for clues as to why our model is struggling:

Unstable reading means we may need to reduce the level of reading and / or increase our collection size.
The difference between training performance and test sets means that we overuse, and we have to reduce the size of our model, or increase familiarity (such as quitting).
Poor performance in both training and test set means we do not qualify, and we may need a larger model or a different level of learning.
A common practice is to start with a small model and extend our parameters until we see a split in training and validation, which means our data is extremely limited.

Importantly, because the weights of the neural network start at random, your scores will fluctuate, regardless of the hyperparameter. Another way to make accurate judgments is to use the same hyperparameter configuration multiple times, with different random seeds.

If our results are satisfactory, we are ready to use our model!

If we have made a test set, apart from our validation data, this is where we use it to test our model. The capture test set provides a final confirmation of the performance of our model on intangible data.

Libraries and Dependencies

1.Python 3

Python is a general-purpose i, object-oriented, and high-level programming language.s it is a **new version** of the language that is **incompatible** with the 2.x line of releases. The language is mostly the same as previous , but various details, especially how built-in objects like dictionaries and strings work, have changed considerably, and a lot of deprecated features have finally been removed.

2.Tensorflow 2.0

TensorFlow is open source platform for machine learning. It has a comprehensive, concise flexible ecosystem of tools, libraries and community resources that helps the developers to create machine learning and deep learning models by writing less amount of code

3.Streamlit

Streamlit is an open-source app framework for Machine Learning and Data Science teams. It provides the facility to developers to build attractive web apps within minutes without writing all the html Css and javascript

4.OpenCV

OpenCV provides a real-time optimized Computer Vision library, tools, and hardware, it is a library which is generally used in image processing and various object detection problem

These dependencies are generated through requirement.txt file below is the screenshot of the same:

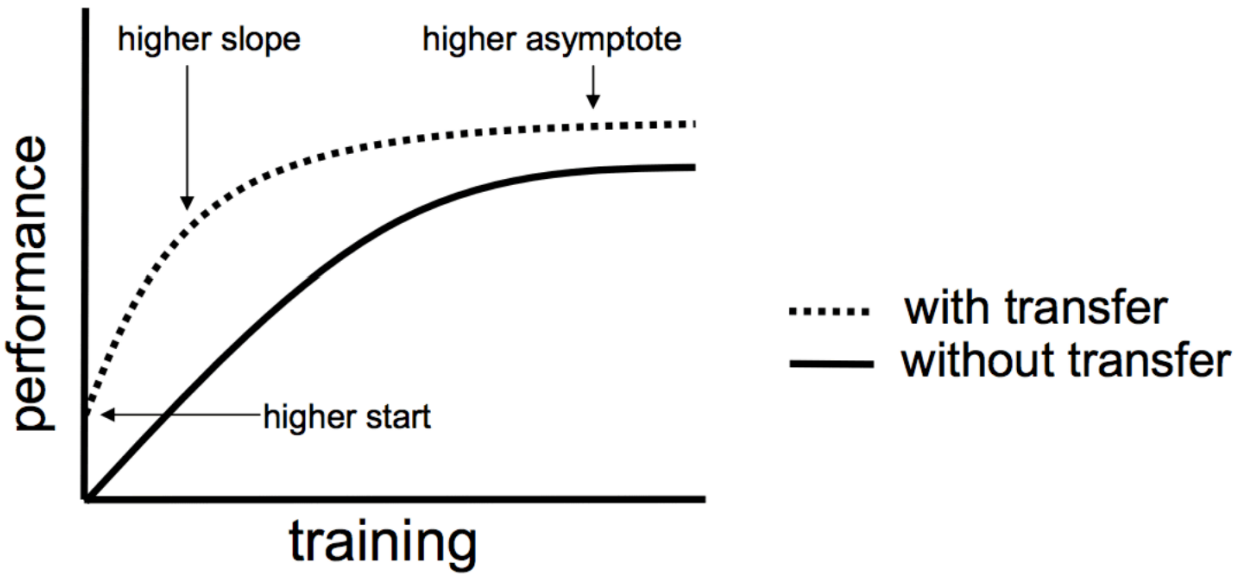
```
keras==2.9.0
numpy==1.21.6
opencv_python==4.5.5.64
streamlit==0.87.0
streamlit_webrtc==0.37.0
tensorflow==2.9.1

```

Nueral Network's Architecture

Transfer learning

In transfer learning we use a pre-trained model as a starting point for our new model.As we know that training a deep learning model is both computationally expensive and requires a large amount of data but sometimes we don't have much huge data to train our deep learning models or hardware resources. In that case we use a model which is trained on similar types of data by professionals of the concerned field. We use this model as a starting model for our model.



Transfer learning gives fast convergence but it should be used wisely as it gets easily overfit.
We could use transfer learning in Three ways which are given below:

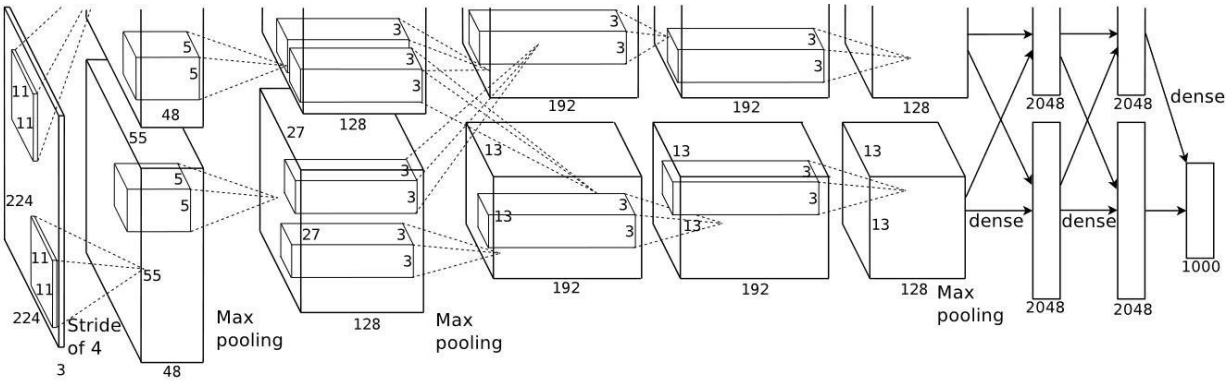
- Take output from the last layer of pretrained model as input feature for our next model and train a few fully connected layers on top of it.
- Add a few layers on top of pretrained models and train the whole network.Keep learning rate small.
- Train a softmax classifier on top of a pretrained model.

Different Pre-Trained Models Used In this project

In this project we have used four different pretrained models. These models are previously trained on large image datasets. Hence we have used them in our project. The models and their architecture and uses are mentioned below.

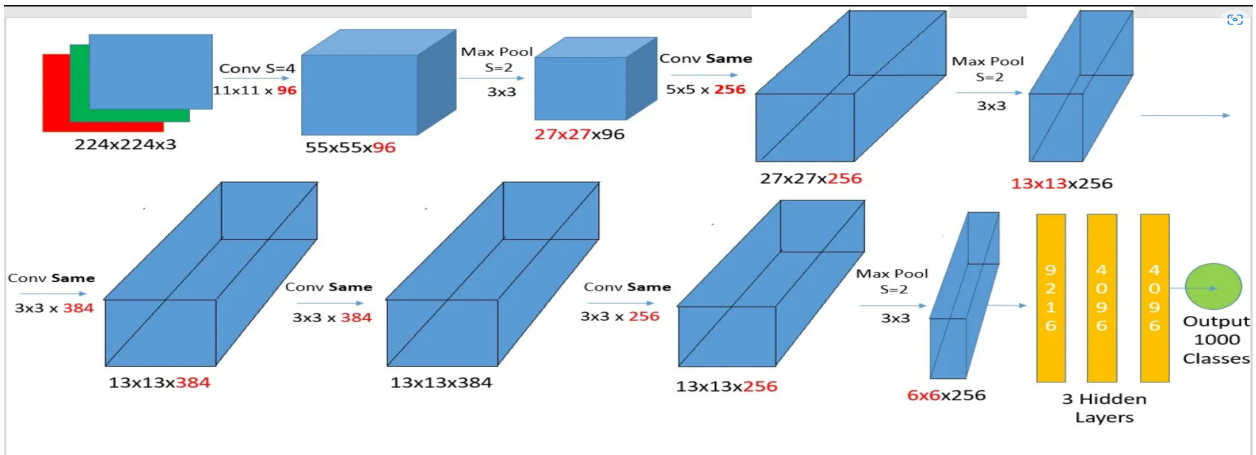
- AlexNet:- For image classification related deep learning problems alexnet is best. In 2012 alexnet won imagenet dataset classification problem. Imagenet is a dataset consisting of 1.2 million images belonging to 1000 categories. So if we are using transfer learning to train an image classification model we must use AlexNet.
  Let’s see the architecture of AlexNet.

AlexNet Architecture:-



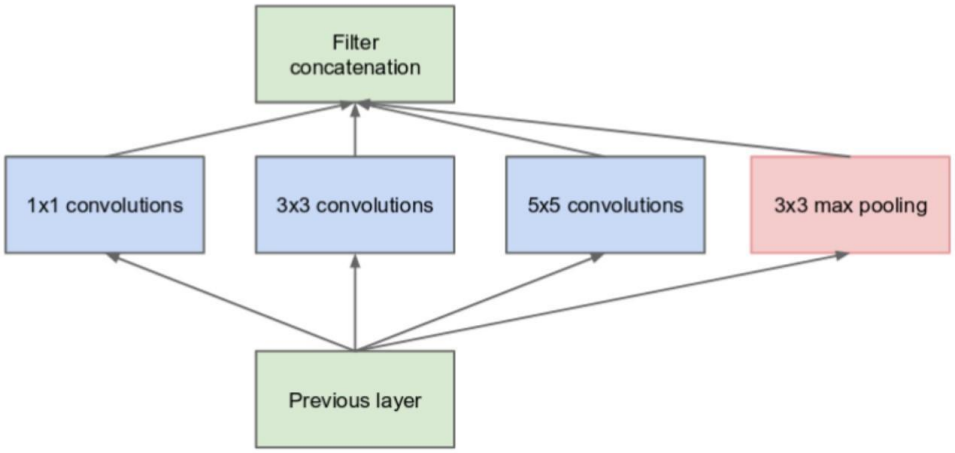
In AlexNet architecture

- There are 8 layers which are trainable.
- RGB images are input to AlexNet models.
- There are 5 layers which are combinations of convolutions and max-pooling layers.
- Then it has 3 fully connected layers.
- Relu activation is used in all layers of AlexNet.
- It used two Dropout layers.
- Softmax activation function is used.
- There are a total of 62.3 million parameters in alexnet.



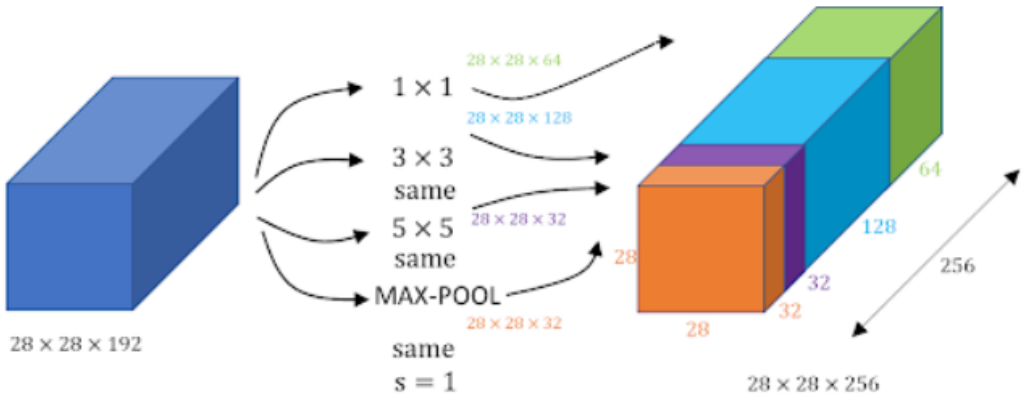
InceptionNet:-

The theory behind InceptionNet is to make the network wider instead of deeper. InceptionNet suggests using various sizes of CNN's at the same layer which also reduces the number of trainable parameters in our network.



In inception networks we use multiple inception modules in a network. Inception module is shown in the above diagram.GoogLeNet has 9 such inception modules stacked linearly. It is 22 layers deep (27, including the pooling layers). It uses global average pooling at the end of the last inception module.

Architecture of InceptionNet:-



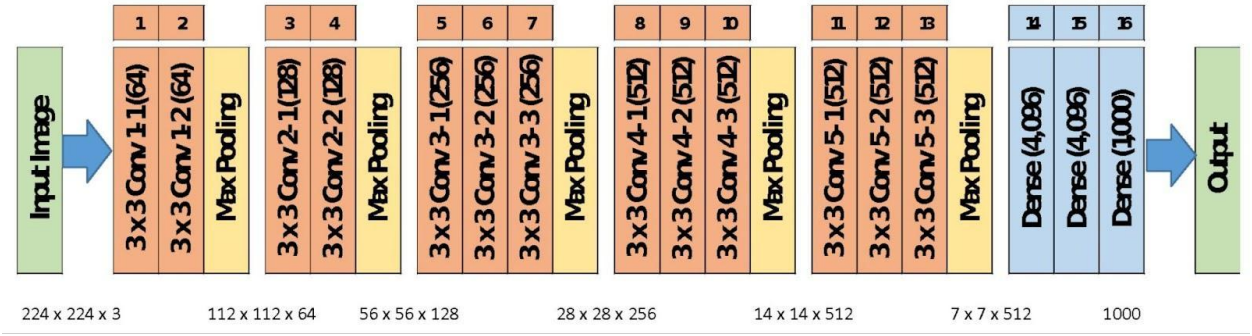
- Performance of InceptionNets is high.

- Easy to understand when breaked into modules.
- Coputance optimization.

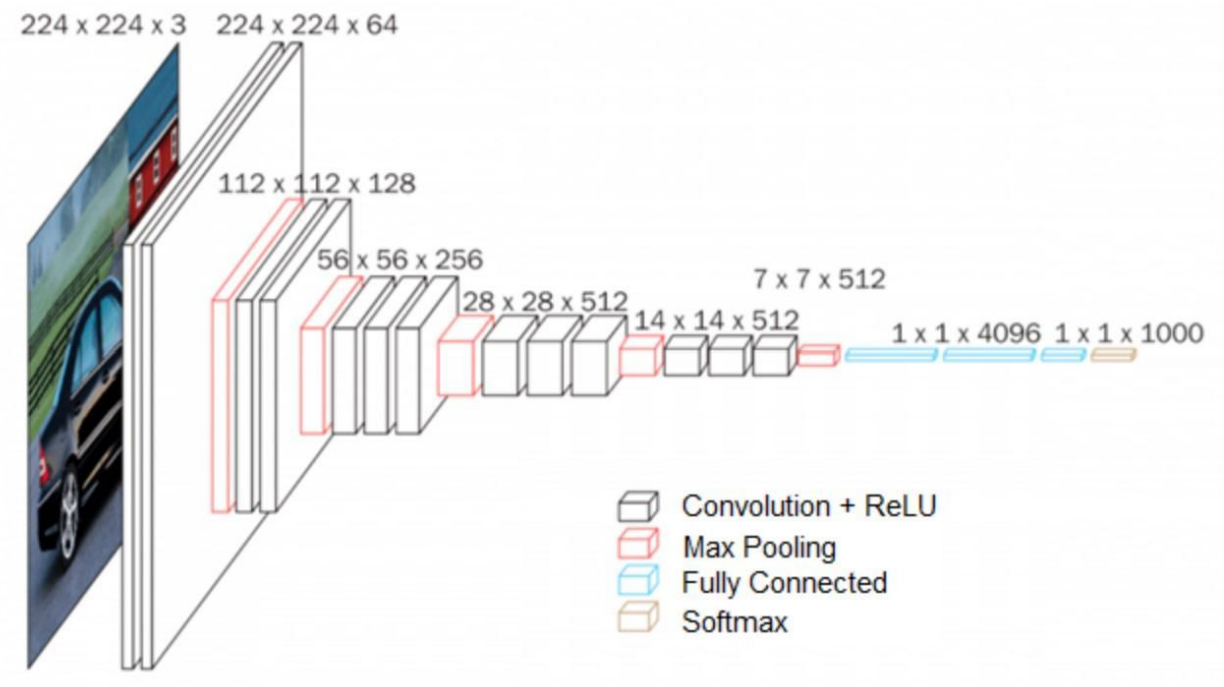
VGG16

VGG16 was proposed in 2013 by Karen Simonyan and Andrew Zisserman. The idea behind VGG16 is that instead of using multiple 3*3 , 5*5 and 7*7 convolutions why not use 3*3 convolutions only. This increases the depth of network but also with each 3*3 convolution we also use a non-linear activation layer which makes our model more discriminative.

There are two versions of VGG which are VGG16 and VGG32. In our project we have used VGG16.



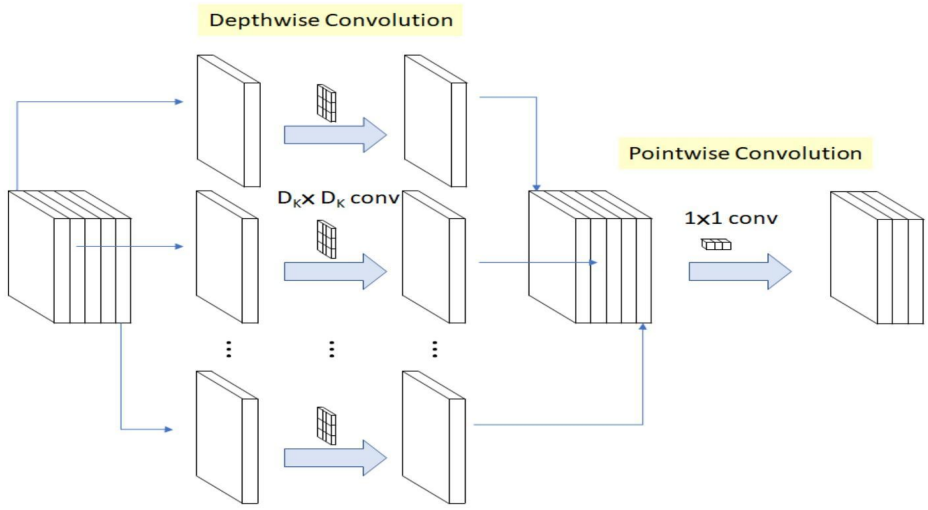
Architecture Of VGG16:-



- There are total 16 layers in VGG16.
- Maxpools along with convolution layers are used.
- ReLu activation is used.
- The classifier layer is softmax.

MobileNet:-

MobileNet was developed by tensorflow. MobileNet was the first mobile computer vision model. It was very lightweight. MobileNet uses convolutions which are depthwise separable.



Architecture of MobileNet:-

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1 Conv / s1	$3 \times 3 \times 512$ dw $1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$ $14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Whole Network Architecture for MobileNet

- It has ten 3*3 depthwise separable convolution layers.
- It has ten 1*1 convolution layers.
- It is very light.
- It is mobile thats why it is called MobileNet.

Dataset

Dataset Description

We have built a deep learning model which detects the real time emotions of students through a webcam so that teachers can understand if students are able to grasp the topic according to students' expressions or emotions and then deploy the model. The model is trained on the FER-2013 dataset . This dataset consists of 35887 grayscale, 48x48 sized face images with seven emotions - angry, disgusted, fearful, happy, neutral, sad and surprised. Here is the dataset

link:- https://www.kaggle.com/msambare/fer2013

The data set is split with a train-validation-test split of 75–12.5–12.5.

Let's examine the distribution of our data. The frequency structure indicates that the data set is distorted. Although there are more than 9000 happy faces, less than 1000 ‘disgusting’ faces. This tells us that we may not be able to detect the ‘disgusting’ face successfully with this data set.

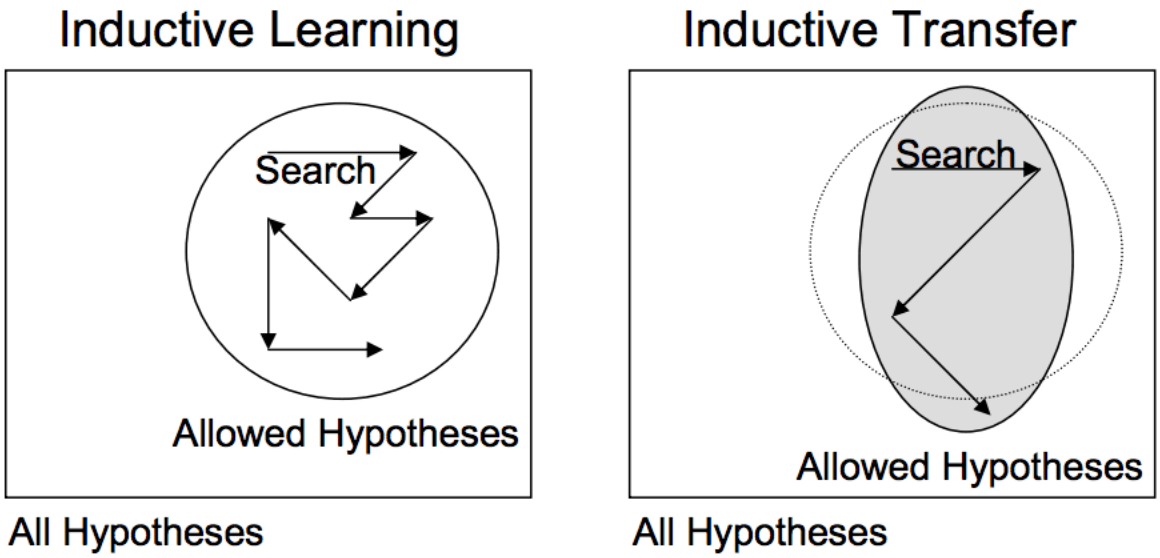
Training

1) Using Transfer Learning

Transfer learning is a machine learning method in which a work model is also used as the starting point for a second activity model.

It is a popular form of deep learning where previously trained models are used as a starting point for computer vision and natural language processing functions when given the large computer and time-consuming resources needed to develop neural network models in these problems and major overrides. to provide related issues.

Transfer learning and domain adaptation refer to the situation where what has been learned in one setting … is exploited to improve generalization in another setting This form of transfer learning used in deep learning is called inductive transfer. This is where the scope of possible models (model bias) is narrowed in a beneficial way by using a model fit on a different but related task.



How to use Transfer Learning?

We can use transfer to learn from your predictive modeling problems.

The two most common methods are as follows:

- Improve Model Method
- Pre-Trained Model Method

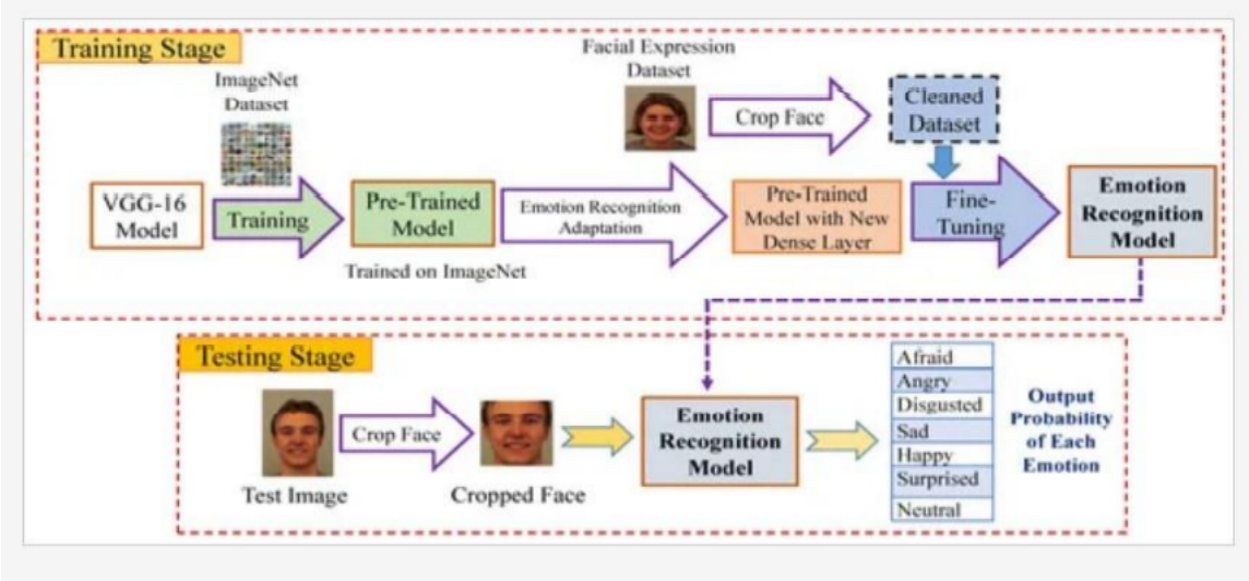
Improve Model Method

Select Source Function. You should select a problem-related predictive model for bulk data where there is a specific relationship to input data,

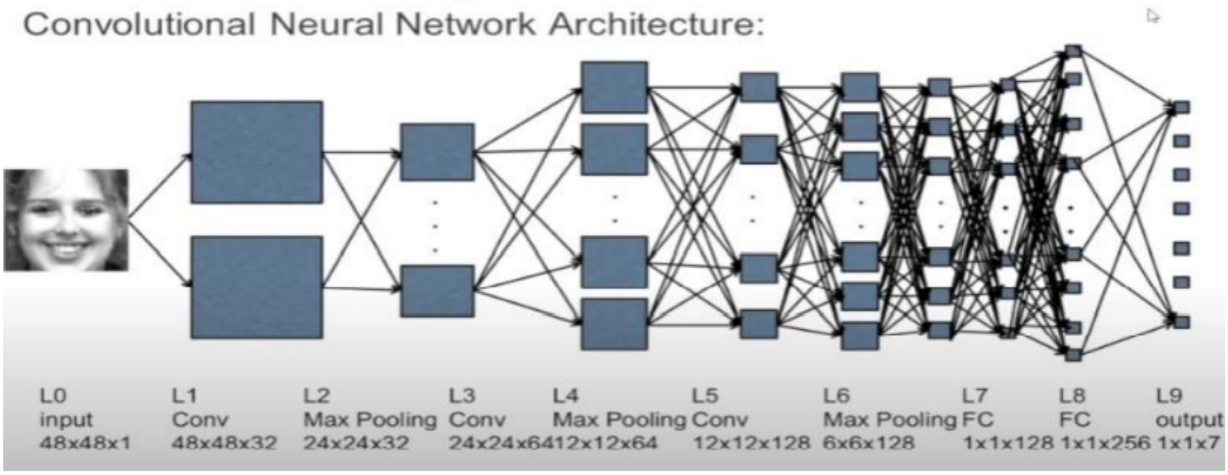
output data, and / or concepts learned during mapping from input to output.
Upgrade Source Model. Next, you should develop a competent model for this first job. The model should be better than the non-model model to ensure that certain feature reading is done.
Reuse Model. A model equivalent to the source function may be used as the first model of the second work model in which you are interested. This may involve the use of all or part of the model, depending on the model used.
Shuna Model. Optionally, the model may need to be modified or upgraded to the input-output pairing data available in order to perform the function in which it is interested.\

Pre-trained Model Approach

1. Select Source Model. A pre-trained source model is chosen from available models. Many research institutions release models on large and challenging datasets that may be included in the pool of candidate models from which to choose from.
2. Reuse Model. The model pre-trained model can then be used as the starting point for a model on the second task of interest. This may involve using all or parts of the model, depending on the modeling technique used.
3. Tune Model. Optionally, the model may need to be adapted or refined on the input-output pair data available for the task of interest.



2) Using CNN layers



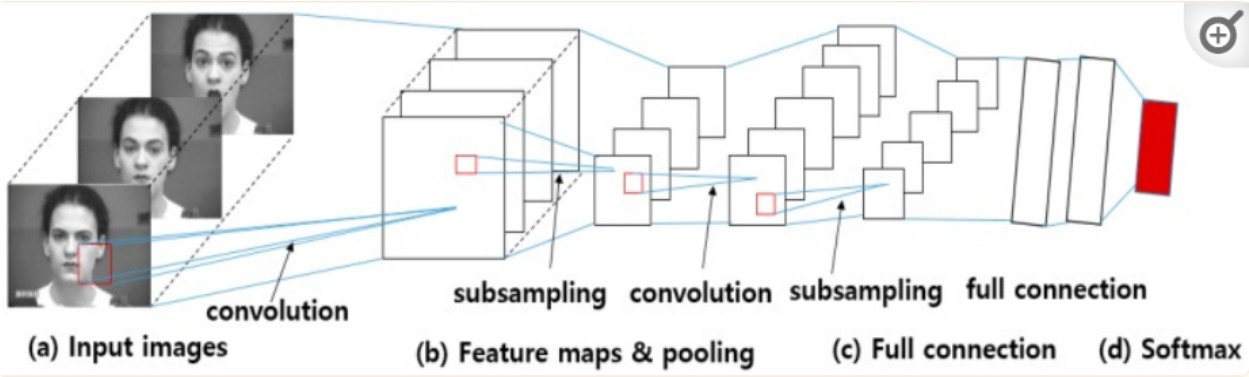
In our CNN model we have four convolution layers and two fully connected layers.We have used ‘relu’ activation function. Output model is deep so it could easily overfit, thats why we have used ‘dropout’ rate of 0.25. We have also used batch normalization to normalize inputs at later layers. Our optimisation function is ‘adam’ and loss function is categorical cross entropy.
In this model we have

- **Total params: 4,478,727**
- **Trainable params: 4,474,759**
- **Non-trainable params: 3,968**

Optimizing the model

We have trained five models on top of our data. Four models are pretrained models which we have retrained on top of our dataset and one is three layered convolution network which we have trained from scratch. The training results which we got during our model creation is shown below.

Model	Epochs	Train Accuracy	Validation Accuracy
AlexNet	20	99.28	62.81
AlexNet2	15	81.67	66.07
InceptionV3	50	43.47	49.90
MobilNet	10	67.02	38.50
VGG16	25	99.27	41.54
Deep CNN	20	80.34	68.28

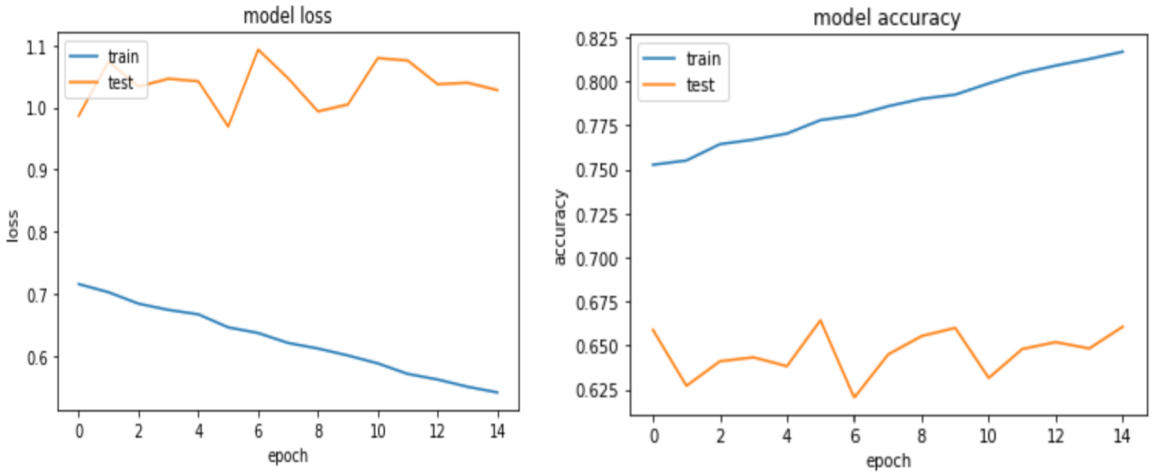


Procedure of CNN-based FER approaches: (**a**) The input images are convolved using filters in the convolution layers. (**b**) From the convolution results, feature maps are constructed and max-pooling (subsampling) layers lower the spatial resolution of the given feature maps. (**c**) CNNs apply fully connected neural-network layers behind the convolutional layers, and (**d**) a single face expression is recognized based on the output of softmax (face images are taken from CK+ dataset

Loss & Accuracy Plot

We got two better performing models one is AlexNet and one is Fully connected CNN. Below is the graph of their performance.

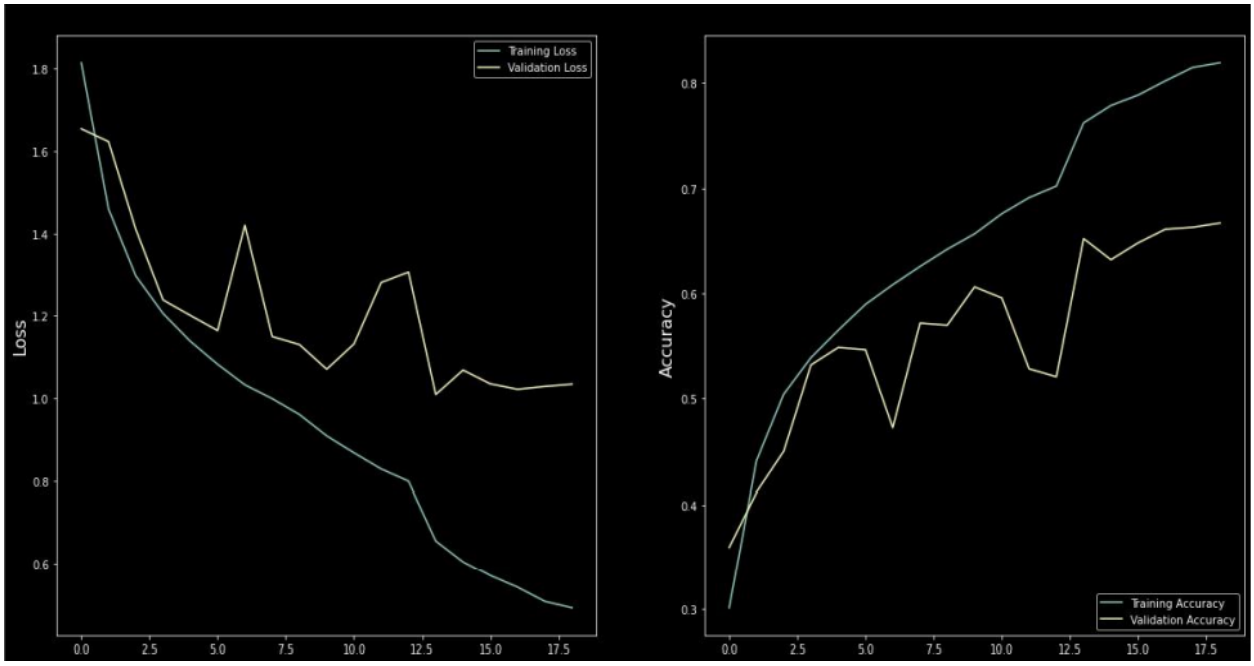
AlexNet



LOSS ACCURACY

Although the AlexNet has achieved accuracy of 66 percent after some hyperparameter tuning, it was only able to classify two emotions correctly.

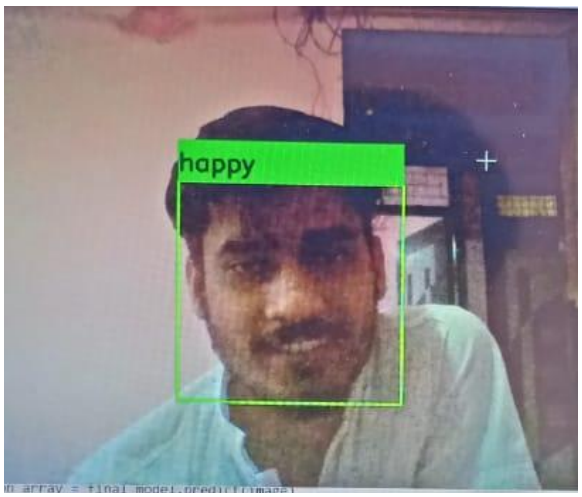
Convolution Neural Network (CNN)

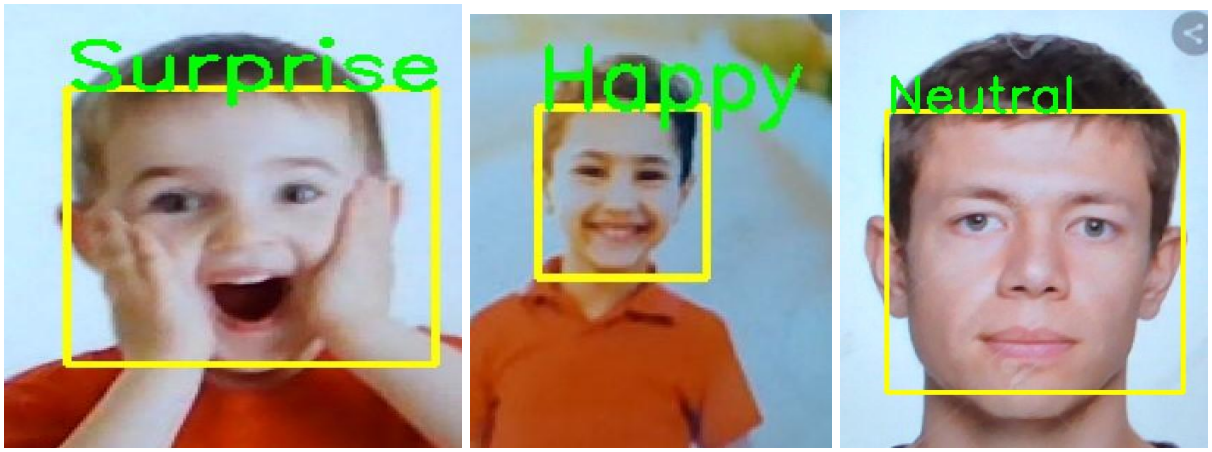


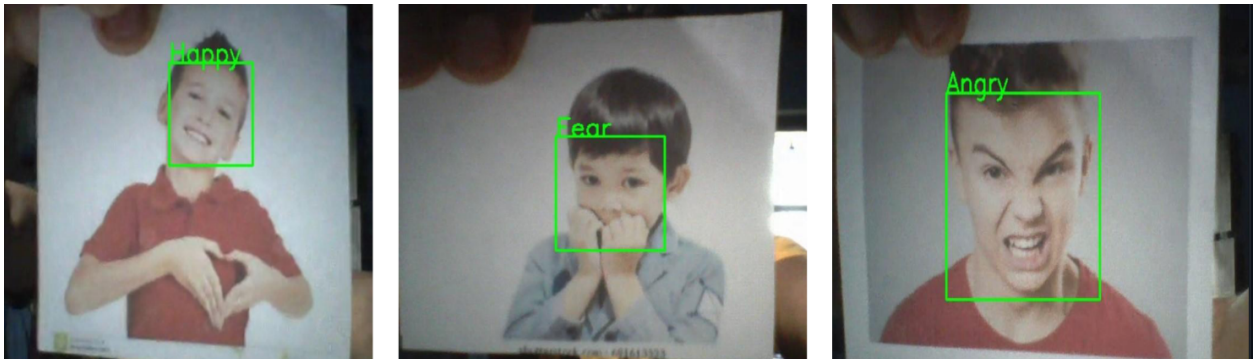
LOSS **ACCURACY**

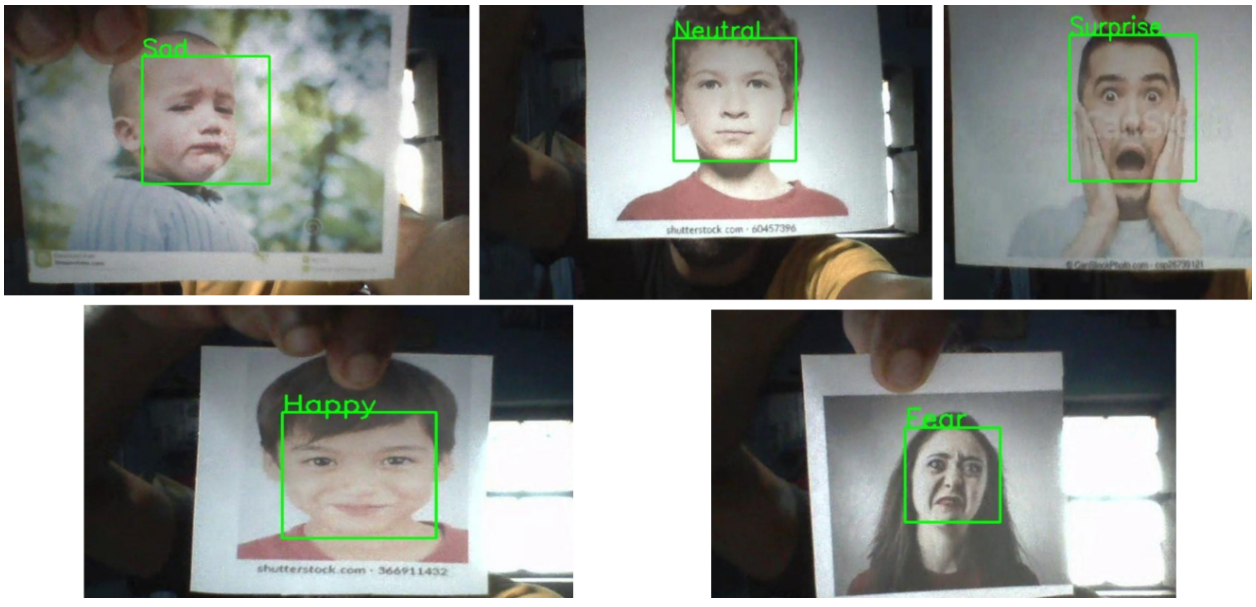
Cnn was performing better than AlexNet. It was able to classify six emotions correctly. However it was not able to classify disguised emotions. The reason behind this could be data. As our dataset is highly imbalanced. We had only 1200 disguised images and out of 35000 images.

OUTPUT




Surprise
Happy
Neutral


Happy
Fear
Angry


Sad
Neutral
Surprise
Happy
Fear

Challenges

We have face various challenges in order to do this project:

- The dataset is not equally distributed some type of images are more in numbers while some are less ,that create the first challenge for us
- We tried various Neural models like VGG , Alex Net etc , to get the best model for this project and we also tried our own constructive model .
- We also faced issue while tunning the hypermeter of the various neural models.
- We also faced issue to host the our model on streamlit and cloud platform

Conclusion

We were able to solve the problem live face emotion detection. The best model we could train was deep CNN. We were able to identify six emotions correctly which are Happy, Sad, Fear, Surprise, Anger and Neutral. We were not able to identify disgust emotion. The possible reason for which could be dataset skewness. Out of 35000 images we had 1200 disgust images only. This was a beautiful journey. We learned and enjoyed a lot building this project.

References

- https://www.appliedaicourse.com/
- https://medium.com/analytics-vidhya/building-a-real-time-emotion-detector-towards-machine-with-e-q-c20b17f89220
- B. Ko, “A Brief Review of Facial Emotion Recognition Based on
- Visual Information,” Sensors, vol. 18, no. 2, p. 401, 2018.
- [2] R. C. Chivers, V. V Ramalingam, and A. Pandian, “Facial Emotion
- Recognition System – A Machine Learning Approach Facial Emotion
- Recognition System ± A Machine Learning,” 2018.
- [3] M. H. Siddiqi, M. Alruwaili, J. Bang, and S. Lee, “Real Time Human
- Facial Expression Recognition System using Smartphone,” vol. 17,
- no. 10, pp. 223–230, 2017.
- [4] A. Savva, V. Stylianou, K. Kyriacou, and F. Domenach, “Recognizing
- Student Facial Expressions: A Web Application,” IEEE Glob. Eng.
- Educ. Conf. EDUCON, vol. 2018–April, pp. 1459–1462, 2018