Live Class Monitoring System (Face Emotion Recognition)

Mahin Arvind C
Almabetter, Bangalore

Abstract

Face emotion recognition technology detects emotions and mood patterns invoked in human faces. This technology is used as a sentiment analysis tool to identify the six universal expressions, namely, happiness, sadness, anger, surprise, fear and disgust.
Identifying facial expressions has a wide range of applications in human social interaction detection for industries like digital learning and market research.
In this project, the Face Emotion Recognition 2013 Dataset was used to train five different types of architectures built using convolutional layers. Three custom convolutional networks were designed using traditional convolutional layers, VGG blocks and a simple inception block. Apart from this, transfer learning was administered on VGG-16 and ResNet-50 architectures using ImageNet weights.
A sum total of five models were trained and evaluated during the course of the project. The best-performing model was deployed as a Streamlit application to perform real-time emotion recognition.

1. Problem Statement

In this project, we'll be using the Face Emotion Recognition 2013 dataset to

1. Effectively identify student emotions using minimum reference images,
2. Perform face emotion recognition on a webcam video feed in real-time,
3. Create a web application to access the model and
4. Deploy model on the cloud as an end-to-end solution.

2. Introduction

The Indian education landscape has been undergoing rapid changes for the past 10 years owing to the advancement of web-based learning services, specifically, eLearning platforms. Global E-learning is estimated to witness an 8 times growth over the next 5 years to reach two billion USD in 2021. India is expected to grow with a CAGR of 44% crossing the 10M users mark in 2021. Although the market is growing on a rapid scale, there are major challenges associated with digital learning when compared with brick and mortar classrooms. One of many challenges is how to ensure quality learning for students. Digital platforms might overpower physical classrooms in terms of content quality but when it comes to understanding whether students are able to grasp the content in a live class scenario is yet an open-end challenge.In a physical classroom during lecturing teachers can see the faces and assess the emotion of the class and tune their lecture accordingly, whether it is going fast or slow. He can identify students who need special attention. Digital classrooms are conducted via video telephony software programs where it is not possible for medium-scale classes to see all students and access the mood. Because of this drawback, students are not focusing on content due to a lack of surveillance. While digital platforms have limitations in terms of physical surveillance but it comes with the power of data and machines which can work for you. It provides data in the form of video, audio, and texts which can be analysed using deep learning algorithms.
A deep learning-backed system not only solves the surveillance issue, but also removes the human bias from the system, and all information is no longer in the teacher's brain but rather translated into numbers that can be analysed and tracked.

3. Data Description

3.1. Data Preparation

The FER-2013 dataset was created by gathering the results of a Google image search of each emotion and synonyms of the emotions This dataset consists of 22,965 labelled images in the training set, and 5,741 labelled images in the validation set while 7,178 images were used as the test set for evaluation.

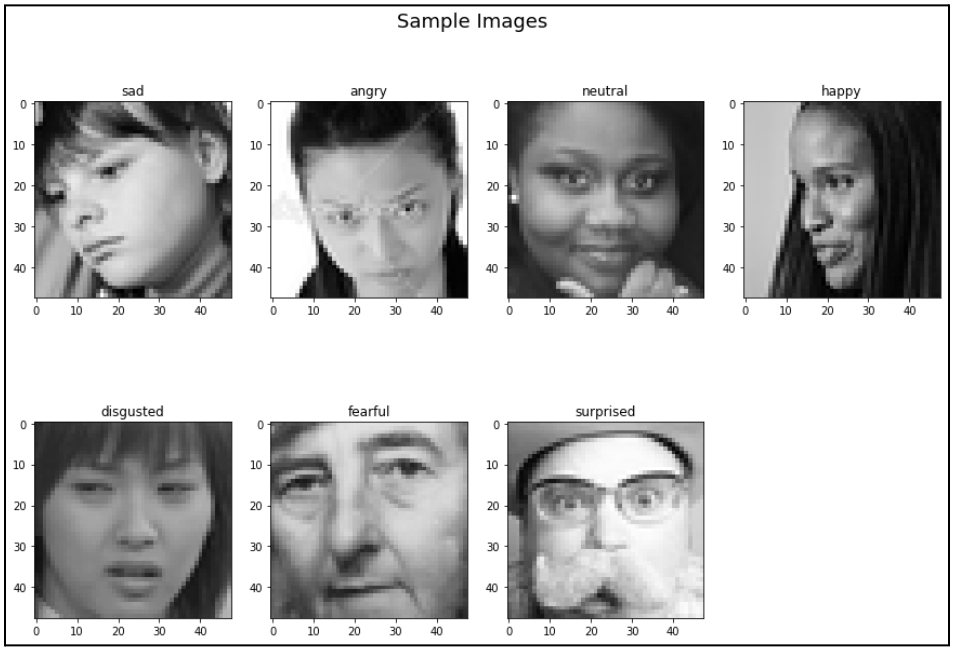

Fig. 1: Sample Images from FER-2013 dataset

Each image in FER-2013 is labelled as one of seven emotions: happy, sad, angry, afraid, surprise, disgust, and neutral, with 'happy' being the most prevalent emotion. The images in FER-2013 consist of both posed and unposed headshots, which are in grayscale and 48x48 pixels in dimension.

3.2. Data Augmentation

Image data augmentation is a technique that can be used to artificially expand the size of a training dataset by creating modified versions of images in the dataset. Training deep learning neural network models on more data can result in more skilful models, and the augmentation techniques can create variations of the images that can improve the ability of the fit models to generalize what they have learned to new images.[1] The Keras deep learning neural network library has been used to perform image data augmentation via the ImageDataGenerator class. The train set's image augmentation specifications have been mentioned in **Table 1**.

Apart from augmenting images, all the image pixel values in the dataset are rescaled from the [0, 255] range to the [0,1] range.

Feature	Range
Zoom	[0, 0.2]
Horizontal Flip	True/False.
Brightness	[0.2, 1.2.]

Table 1. Data Augmentation Specifications

4. Methodology

The performance of five convolutional network architectures was evaluated and compared to effectively classify face emotion. Keras library's functional API was utilized in implementing the models.

4.1 Generic ConvNet

The generic CNN was arbitrarily designed with four convolutional layers activated using the ReLu function. Convolutional filters of sizes three by three and five by five were used. Each level used batch normalization and a max-pooling layer for efficient computation. After flattening, two dense layers and a

softmax layer were used to classify the image into one of the seven universal facial expressions. As a regularisation measure, a 25 per cent dropout was enforced after each activation function. The architecture of the generic model has been visualised in **Fig. 2**. The model had 4,474,759 trainable parameters and 3,968 non-trainable parameters.

4.2 VGG block ConvNet

The VGG convolutional neural network architecture, named for the Visual Geometry Group at Oxford, achieved top results in the LSVRC-2014 computer vision competition.
The key innovation in this architecture was the definition and repetition of VGG-blocks. These are groups of convolutional layers that use small filters (e.g. 3×3 pixels) followed by a max pooling layer[2].
In this model, each VGG block consisted of a specified number of convolutional operations producing one among 64, 128 and 256 output filters. These convolutional operations used three-by-three filters and performed the same convolution. The combination of convolutional layers was followed by a max-pooling with two-by-two filters and with the same padding. Each VGG block was followed by a batch normalization layer and a dropout layer for regularisation.
After flattening, two dense layers and a softmax layer were used to recognize face emotion. The architecture of the implemented model is depicted in **Fig. 3**. The model had 4,821,575 trainable parameters and 2,432 non-trainable parameters.

4.3 Inception block ConvNet

The inception module was described and used in the GoogLeNet model. Like the VGG model, the GoogLeNet model achieved competitive results in the 2014 ILSVRC challenge.
The key innovation of the inception model is the use of a block of parallel convolutional layers with different sized filters (e.g. 1×1, 3×3, 5×5) and 3×3 max pooling layer, the results of which are then concatenated. This is a very simple and powerful architectural unit that allows the model to learn not only parallel filters of the same size but parallel filters of differing sizes, allowing learning at multiple scales[2].
After flattening the inception block output, a dense layer and a softmax layer are implemented. The architecture of the model is depicted in **Fig. 4**. The model had 151,121,479 trainable parameters and 512 non-trainable parameters.

4.4 VGG-16 with ImageNet weights

VGG-16 is a convolutional neural network that is 16 layers deep. With the help of transfer learning, it is used to load a pre-trained version of the network trained on more than a million images from the ImageNet database. The pre-trained network can classify images into thousand object categories. As a result, the network has learned rich feature representations for a wide range of images.
There are thirteen convolutional layers, five Max Pooling layers, and three Dense layers which sum up to 21 layers in the VGG-16 model of which there are sixteen learnable layers.
As the VGG-16 model, accepts only three channel images, the grayscale input images were converted into three channel images. The top layer of the model was omitted to suit the dataset specifications and all the layers but the last four were frozen with ImageNet weights.
The model was followed by two fully connected ReLu layers and a softmax layer for emotion recognition. The architecture of the implemented model is depicted in **Fig. 5**. The model had 7,477,767 trainable parameters and 7,637,824 non-trainable parameters.

4.5 ResNet-50 with ImageNet weights

The Residual Network, or ResNet, architecture for convolutional neural networks, proposed by Kaiming, achieved notable success on the 2015 version of the ILSVRC challenge.

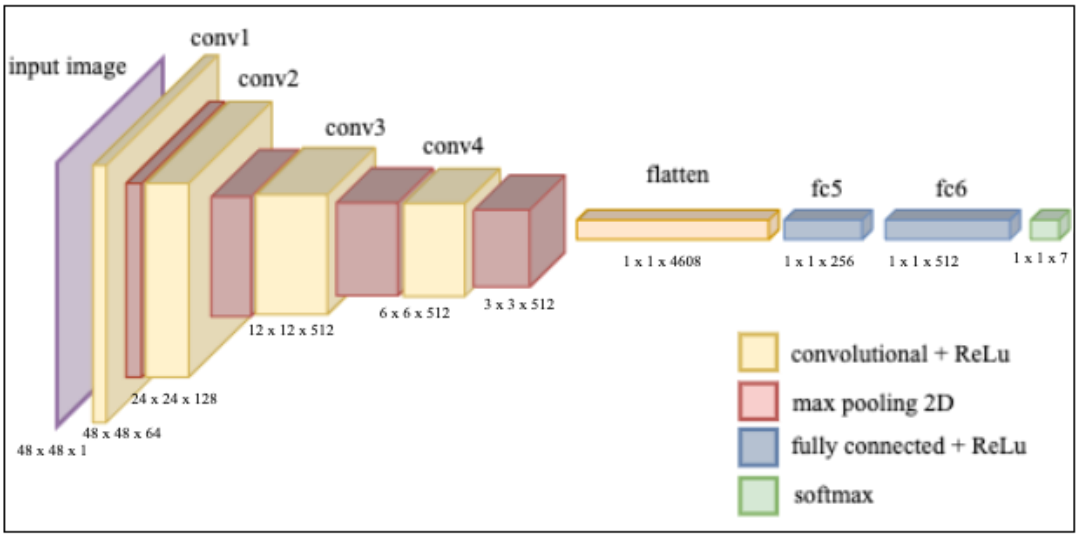

Fig. 2 : Generic Convolutional Network

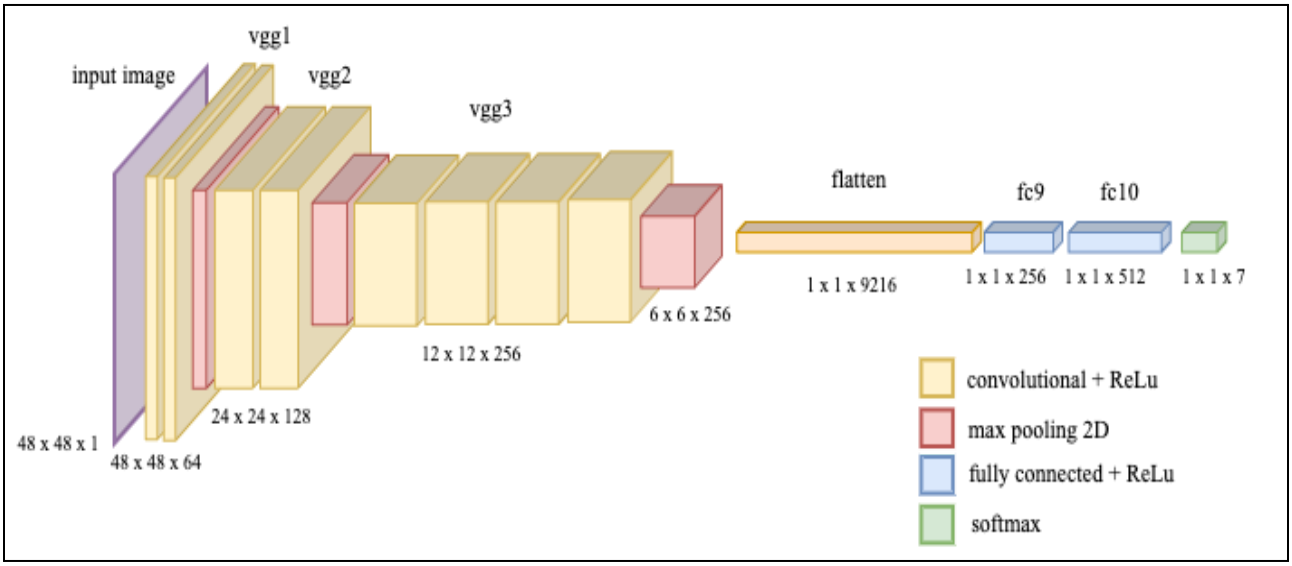

Fig. 3 : ConvNet with VGG Blocks

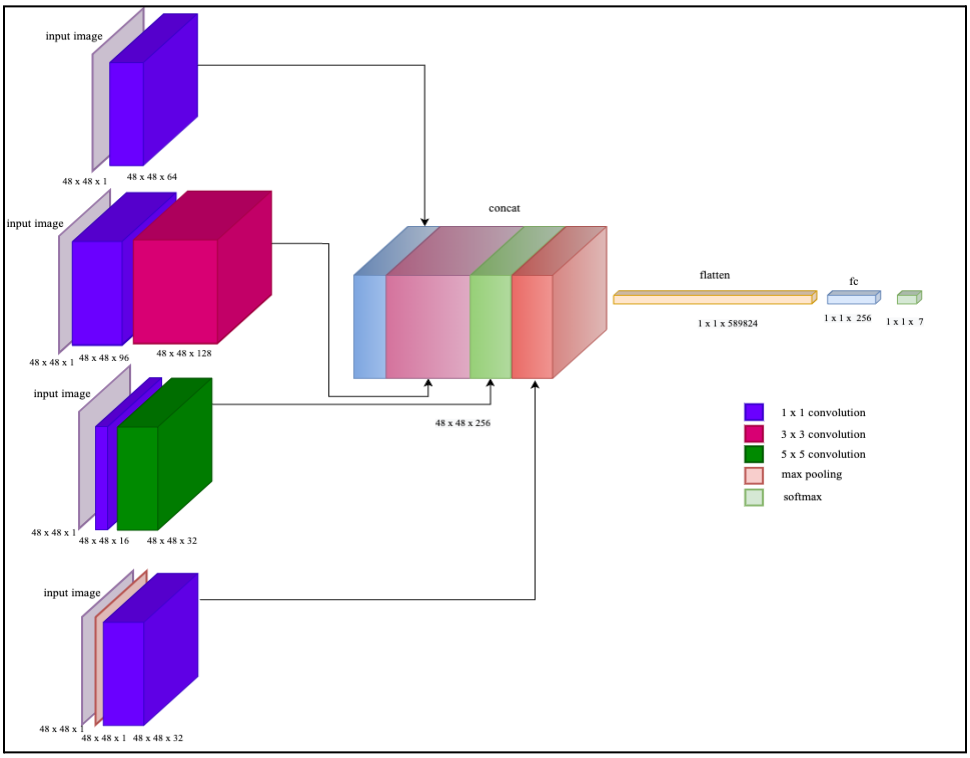

Fig. 4 : Simple Inception Block + Dense Layer

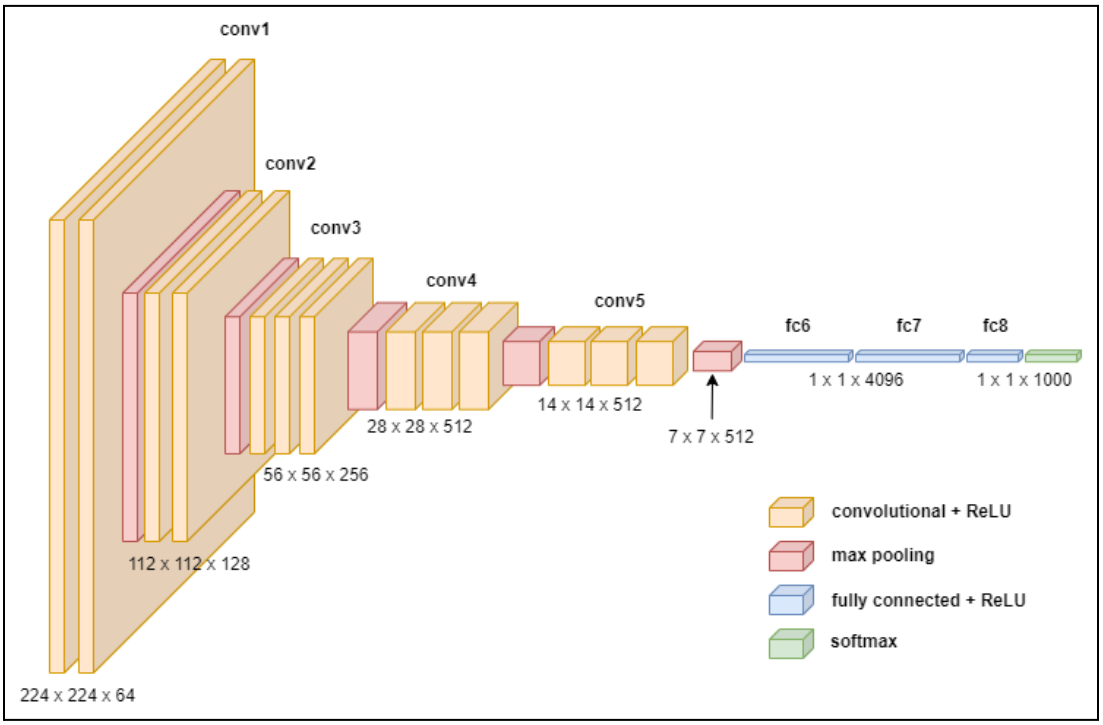

Fig. 5: VGG-16

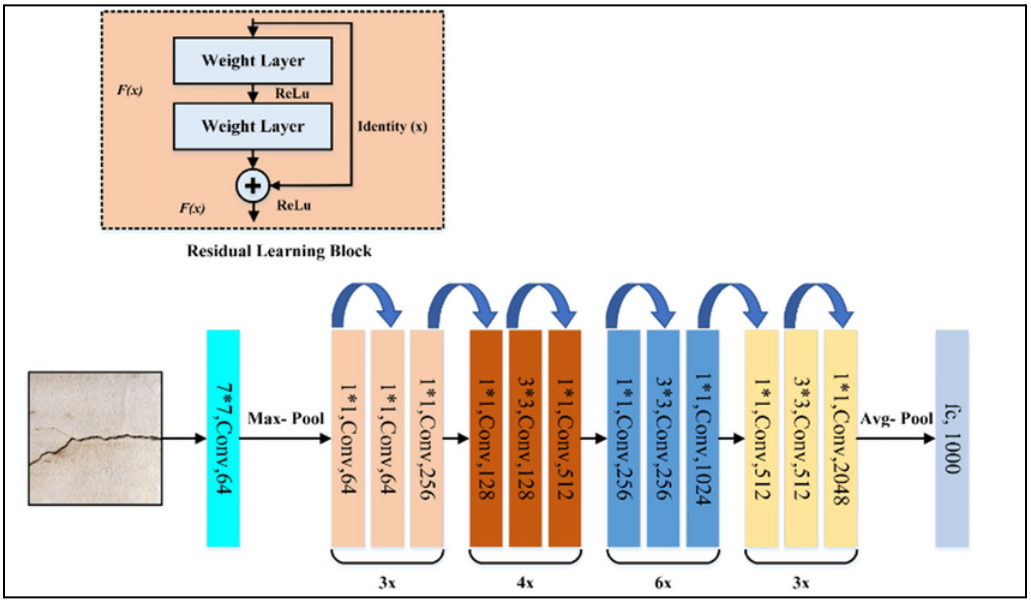

Fig. 6: ResNet-50

A key innovation in the ResNet architecture was the residual module. The residual module, specifically the identity residual model, is a block of two convolutional layers with the same number of filters and a small filter size where the output of the second layer is added with the input to the first convolutional layer. The input to the module is added to the output of the module and is called a shortcut connection (depicted in **Fig. 6**).

Like the VGG-16 model, transfer learning was implemented using the ImageNet weights for the ResNet-50 model and was trained after freezing the penultimate layers. The model had 5,384,199 trainable parameters and 22,534,528 non-trainable parameters.

5. Evaluation Metrics

In order to quantitatively evaluate the performance of the classifier, classification metrics are used to gauge model performance. The classification metrics used in this project are

Categorical Crossentropy (compile time metric): The categorical cross-entropy loss function calculates the loss of an example by computing the following sum:

$$\text{Loss} = -\sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i$$

Here, $\hat{y}_i$ is the ith scalar value in the model output and y_i is the corresponding target value, and the output size is the number of scalar values in the model output.

Categorical Accuracy (compile time metric): Categorical Accuracy calculates the percentage of predicted values that match with actual values for one-hot labels.

Balanced Accuracy: It is defined as the average of specificity and sensitivity obtained in each class.

Precision: Precision explains how many of the correctly predicted cases actually turned out to be positive. Precision is useful in cases where False Positives are a higher concern than False Negatives. Precision for a label is defined as the number of true positives divided by the number of predicted positives

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

Recall: Recall explains how many of the actual positive cases we were able to predict correctly with our model. It is a useful metric in cases where False Negatives are of higher concern than False Positives.
Recall for a label is defined as the number of true positives divided by the total number of actual positives.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

F1 score: The F1 score combines the precision and recall of a classifier into a single metric by taking their harmonic mean. It is primarily used to compare the performance of two classifiers.

$$F1 = 2.\frac{Precision \times Recall}{Precision + Recall}$$

The F1 score punishes extreme values more. F1 Score is used in cases where FP and FN are equally costly.

6. Model Compilation

During the model compilation and training phase, the loss and metrics monitored were categorical cross entropy and categorical accuracy. Adam Optimization was used with a learning rate of 0.001. Adam is a replacement optimization algorithm for stochastic gradient descent for training deep learning models. It combines the properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems. The model was trained in data with a batch size of 64 for a period of 50 epochs. For efficient training, early stopping was used to stop training when the delta value of monitored validation loss was less than zero for ten consecutive epochs (patience). A callback for reducing the learning rate by a factor of 0.02 was used to find global minima when the validation loss was observed to plateau.

7. Results and Discussion

The classifying models that were trained and evaluated are Generic ConvNet, VGG-block ConvNet, Inception-block ConvNet, ImageNet VGG-16 and ImageNet ResNet-50. Their performances were assessed during compilation with categorical accuracy and categorical cross-entropy loss with the validation dataset. Precision, Recall, F1 score and Balanced Accuracy were the metrics used to evaluate and compare models using the test dataset (depicted in **Table 2**).

The Generic ConvNet achieved a training categorical accuracy of 68.42 per cent and a validation categorical accuracy of 65.2 per cent before plateauing after 34 epochs(graph depicted in **Fig. 9**). When evaluated with the test dataset, the model produced a weighted F1 score of 64.38 per cent and a balanced accuracy score of 61.12 per cent. The multi-class confusion matrix can be observed in **Fig. 11.**

The Inception block ConvNet achieved a training categorical accuracy of 58.36 per cent and a validation categorical accuracy of 51.3 per cent before plateauing after 31 epochs (graph depicted in **Fig. 10**). When evaluated using the test dataset, the model produced a weighted F1 score of 49.94 per cent and a balanced accuracy score of 46.62 per cent. The multi-class confusion matrix can be observed in **Fig. 12.**

The VGG block ConvNet achieved a training categorical accuracy of 72.30 per cent and a validation categorical accuracy of 65.01 per cent before plateauing (graph depicted in **Fig. 7**). When evaluated using the test dataset, the model produced a weighted F1 score of 64.95 per cent and a balanced accuracy score of 62.26 per cent. The multi-class confusion matrix can be observed in **Fig. 8.**

The VGG-16 architecture with ImageNet weights achieved a training categorical accuracy of 61.81 per cent and a validation categorical accuracy of 55.14 per cent before plateauing after 44 epochs (graph depicted in **Fig. 13**). When evaluated using the test dataset, the model produced a weighted F1 score of 54.22 per cent and a balanced accuracy score of 52.47 per cent. The multi-class confusion matrix can be observed in **Fig. 15.**

The ResNet50 architecture with ImageNet weights achieved a training categorical accuracy of 36.96 per cent and a validation categorical accuracy of 37.61 per cent before plateauing (graph depicted in **Fig. 14**). When evaluated using the test dataset, the model produced a weighted F1 score of 35.46 per cent and

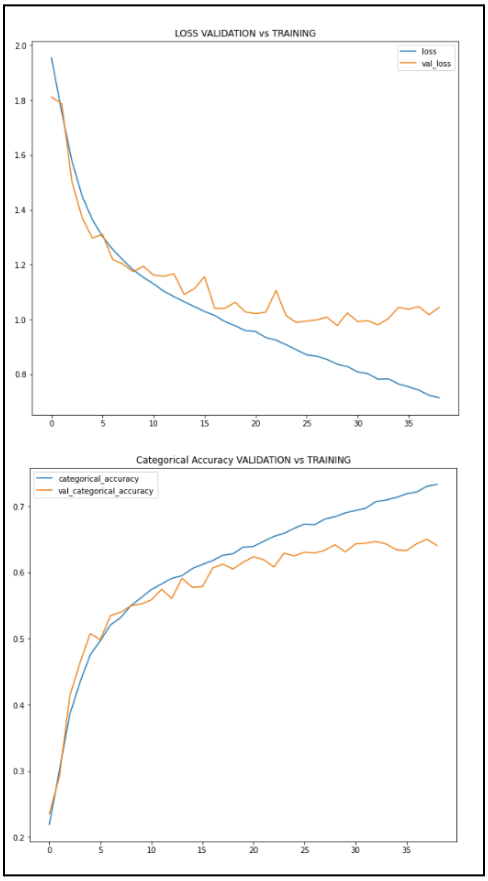

Fig. 7:VGG block ConvNet: Compile Time Categorical Cross entropy Loss and Categorical Accuracy

Fig. 8:VGG block ConvNet: Confusion Matrix

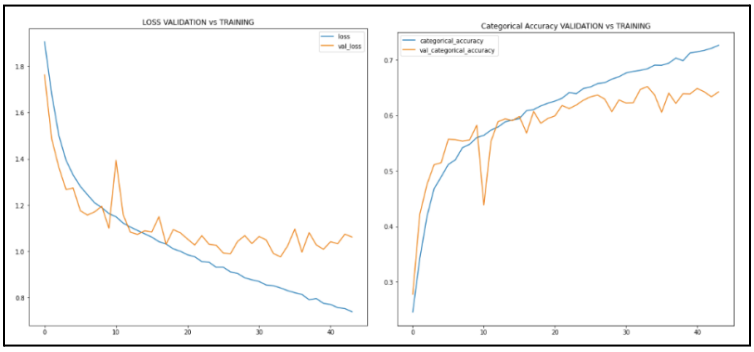

Fig. 9: Generic ConvNet: Compile Time Categorical Cross entropy Loss and Categorical Accuracy

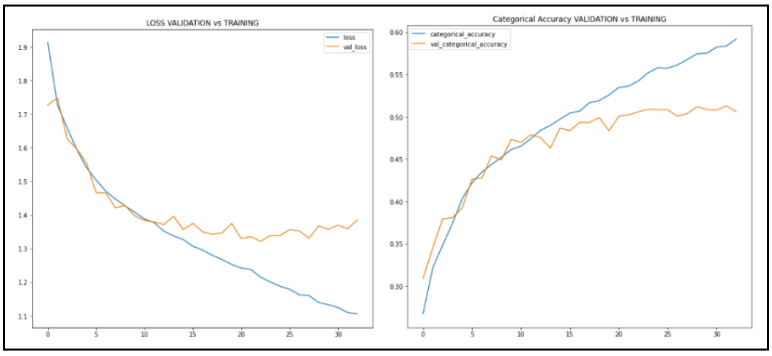

Fig. 10: Inception block ConvNet: Compile Time Categorical Cross entropy Loss and Categorical Accuracy

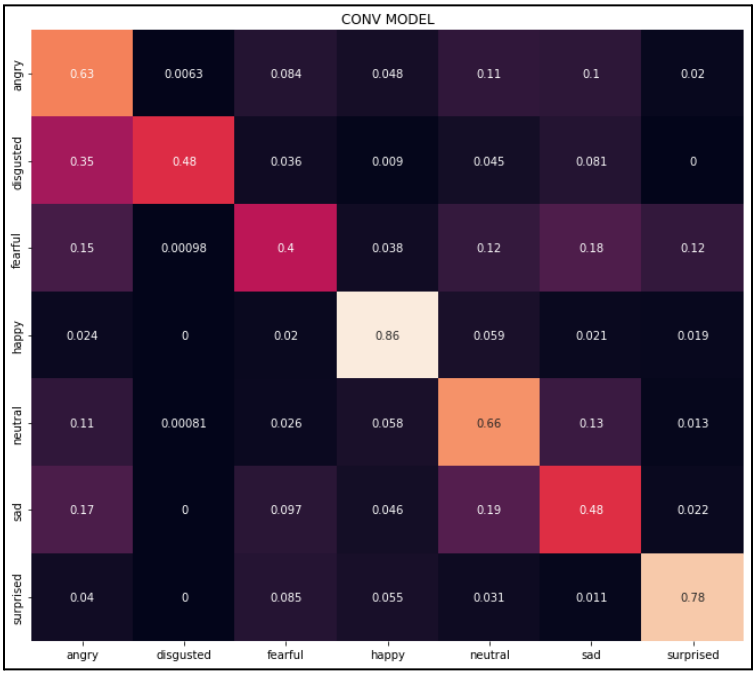

Fig. 11: Generic ConvNet: Confusion Matrix

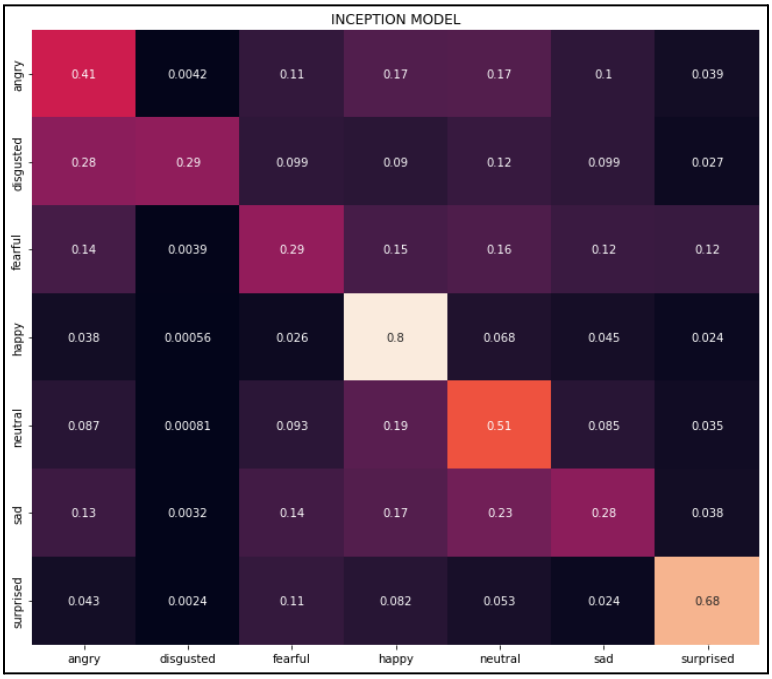

Fig. 12: Inception block ConvNet: Confusion Matrix

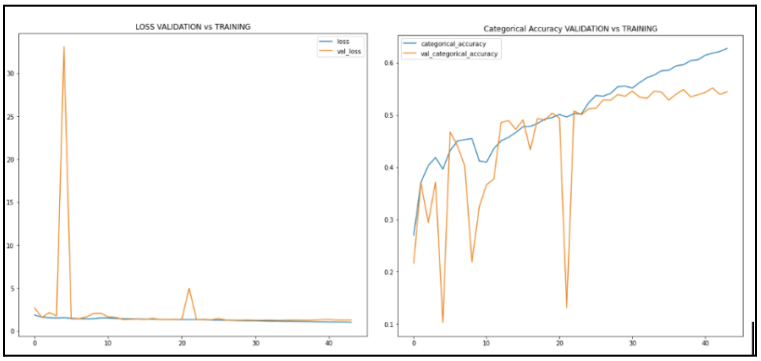

Fig. 13: VGG-16 ImageNet: Compile Time Categorical Cross entropy Loss and Categorical Accuracy

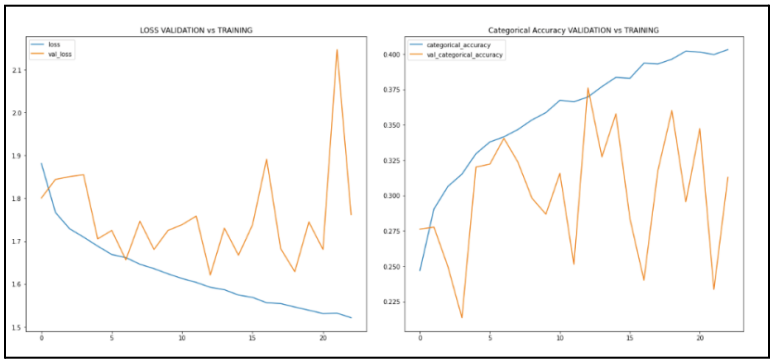

Fig. 14: ResNet-50 ImageNet: Compile Time Categorical Cross entropy Loss and Categorical Accuracy

Conv Model	Recall	Precision	F1 score	Balanced Accuracy
Generic	0.6473	0.6493	0.6438	0.6118
VGG Block	0.6509	0.6527	0.6495	0.6226
INCEPTION Block	0.5139	0.5004	0.4994	0.4662
RESNET50	0.3735	0.3848	0.3546	0.3024
VGG16	0.5527	0.5567	0.5433	0.5247

Table 2 : Model Evaluation Report

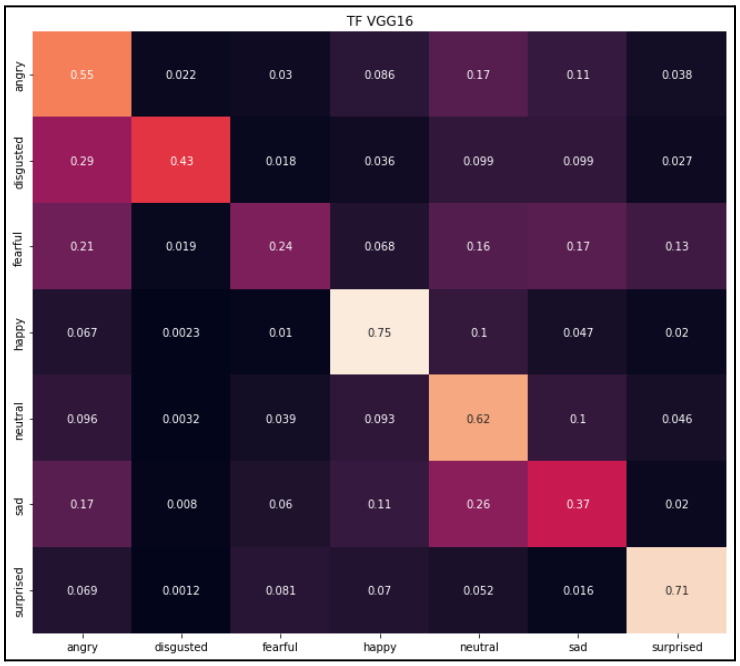

Fig. 15: VGG-16 : Confusion Matrix

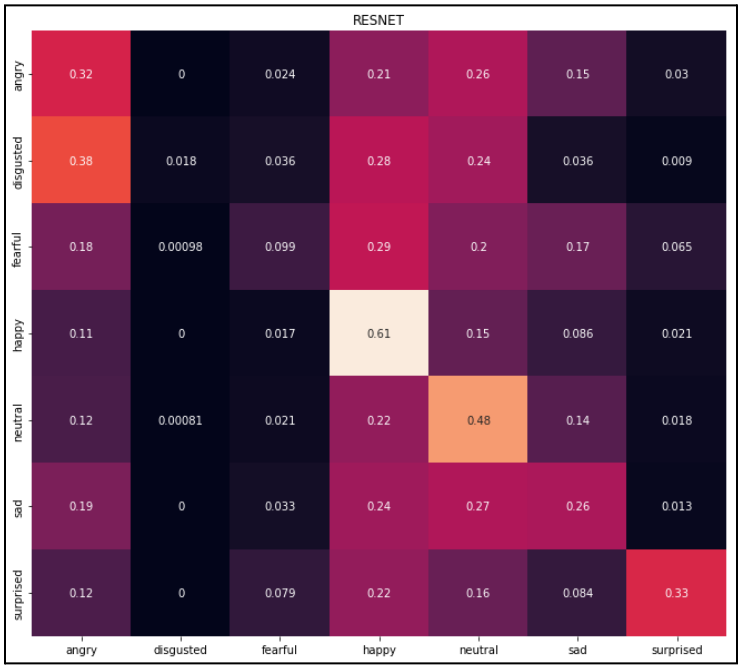

Fig. 16: ResNet-50 : Confusion Matrix

a balanced accuracy score of 30.24 per cent. The multi-class confusion matrix can be observed in **Fig. 16.**

From **Table 2**, it can be observed that the VGG-block ConvNet produced the most optimal results for classifying human face emotions effectively.

It was also observed from the confusion matrix that almost all models had no difficulty in identifying emotions of happiness, surprise and neutral expressions which might be attributed to the abundance of sample images available for use while emotions like '*disgusted'* and *'fear'* were generally harder to classify. This setback was negated as these emotions were irrelevant in the context of digital learning as they fail to provide any actionable feedback on the scope for improvement.

8. Model Deployment

For model deployment, OpenCV's Haarscascade Face Frontal file was used to capture the real-time video feed to detect and extract faces from video for emotion classification. Once the model was successfully executable, it was deployed into a web application for a user-friendly interface. Streamlit library was used to build the front-end for the application.

After running saliently on the local host, the model was deployed in the cloud using Heroku's platform and Streamlit Cloud for remote access. Remote access was observed to have a considerably long loading time as the compressed application size crossed the slug size soft limit of 300 megabytes. (Links: Heroku, Streamlit)

9. Conclusions

During the course of the project, five models were built and compared to effectively identify student emotions using the FER-2013 dataset. The best classifying model, ConvNet with VGG blocks, had an F1 score of 65 per cent and a balanced accuracy score of 62.2 per cent.

It was observed from the confusion matrix that the model could effectively classify emotions like Happy, Angry, Surprised and Neutral while emotions like Fear and Disgusted were classified with some difficulty. However, this issue only posed mild concern as those

emotions were considered irrelevant in the context of digital learning.

The model was used to create a web application to access the model and was deployed as a model on the cloud as an end-to-end solution.

Further improvements that can be sought to be made are the inclusion of images with different levels of illumination and using sophisticated tools to make the web application lighter to suit the parallel processing of multiple video snippets for a real-life classroom environment.

9. References

1. How to Develop VGG, Inception and ResNet Modules from Scratch in Keras by Jason Brownlee
2. Jason Brownlee, “Imbalanced Classification with Python: Better Metrics, Balance Skewed Classes, Cost-Sensitive Learning”
3. Emotion Detection using Convolutional Neural Networks and OpenCV | Keras | Realtime