Technical Documentation

Face Emotion Recognition

Dhiraj Chaudhari Data Science Trainee, AlmaBetter, Bangalore

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 8X over the next 5 years to reach USD 2B 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 a lecturing teacher can see the faces and assess the emotion of the class and tune their lecture accordingly, whether he is going fast or slow. He can identify students who need special attention. Digital classrooms are conducted via video telephony software program (exZoom) where it's not possible for medium scale class (25-50) to see all students and access the mood. Because of this drawback, students are not focusing on content due to 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. Deep learning backed system not only solves the surveillance issue, but it also removes the human bias from the system, and all information is no longer in the teacher's brain rather translated in numbers that can be analysed and tracked.

PROBLEM STATEMENT

During online classes students often tends to lose attention, which leads to overall non-productivity. For a teacher, its often important for its students to easily grasp concept taught by them. Teachers have skills to observe their students and improve their way throughout their teaching. But due to online teaching, observing has become tough which has eventually disturbed student teacher balance and teaching methods. So, our aim was to develop a Face-Emotion-Recognition Model which can be used a micro service as well so that teachers can understand students much better and enlighten the way to teach.

INSPECTING DATASET

We use FER2013 dataset for our model making. We download the dataset in folder format from google. Then we split it into train and test. We check the images by printing and found that the image is of low resolution.

On research we get to know that FER2013 is a well-studied dataset and has been used in ICML competitions and several research papers. It is one of the more challenging datasets with human-level accuracy only at 65±5% and the highest performing published works achieving 75.2% test accuracy. The dataset Easily downloadable on [kaggle].

Figure 1: Images from each emotion class in the FER2013 dataset.

Our dataset has 35,887 images, which are normalized to 48x48 pixels in grayscale and classified into seven categories - Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral. However, the FER2013 dataset is not a balanced dataset, as it contains images of 7 facial expressions, with distributions of Angry (4,953), Disgust (547), Fear (5,121), Happy (8,989), Sad (6,077), Surprise (4,002), and Neutral (6,198).

DATA AUGMENTATION

We researched and experimented with commonly used techniques in existing FER papers and by using ImageDataGenerator achieved our best results with rotation of 20, zoom of 0.2 and height and width shifting by 0.1.

CALLBACK FUNCTION

Callback functions are those functions which are called at the end of every epoch and in this model, we use three callback functions –

ModelCheckpoint, ReduceLROnPlateau and EarlyStopping.

1. **ReduceLROnPlateau**: monitors a certain variable, in this case, validation loss and alters the learning rate when the value stops significantly changing after some certain number of epochs (patience).

2. **EarlyStopping**: At times our optimiser can land in local optima and get stuck there. There is no point in continuing the training as there won't be any further improvements.

3. **ModelCheckpoint**: Saves the best version of our model along with the weights, so that in case any crash occurs, our model can be recovered.

Plotting function is defined to plot the accuracy and loss by taking 'history' of the model as input.

<u>**Hyper-parameter used**</u>:

1. epochs = 100

2. batch_size = 64

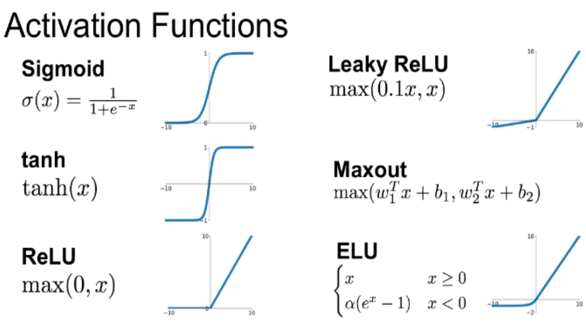
3. learning_rate = 0.0001

4. decay =le-6

TECHNIQUE USED IN MODLES

1• Maximum Pooling (or Max Pooling): like the name states; will take out only the maximum from a pool. This is actually done with the use of filters sliding through the input; and at every stride, the maximum parameter is taken out and the rest is dropped. This actually down-samples the network. Unlike the convolution layer, the pooling layer does not alter the depth of the network, the depth dimension remains unchanged.

2. Activation Function: The activation function is a node that is put at the end of or in between Neural Networks. They help to decide if the neuron would fire or not.



We have different types of activation functions just as the figure above, but for our project my focus will be on Rectified Linear Unit (ReLU). The advantage of using ReLU over other activation functions is that it does not activate all neurons at the same time. From the image for ReLU function above, we'll notice that it converts all negative inputs to zero and the neuron does not get activated. This makes it very computational efficient as few neurons are activated per time.

3. Batch Normalisation: Batch normalization, or batchnorm for short, is proposed as a technique to help coordinate the update of multiple layers in the model. It does this scaling the output of the layer, specifically by standardizing the activations of each input variable per mini-batch, such as the activations of a node from the previous layer.

It removes the ill effects of the internal covariate shift. This has the effect of stabilizing and speeding-up the training process of deep neural networks. Normalizing the inputs to the layer has an effect on the training of the model, dramatically reducing the number of epochs required. It can also have a regularizing effect, reducing generalization error much like the use of activation regularization.

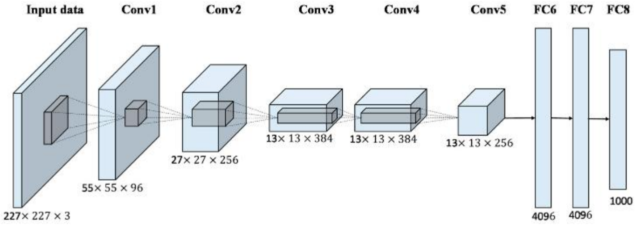
4. **Dropout**: The Dropout layer randomly sets input units to 0 with a frequency of rate at each step during training time, which helps **prevent overfitting**. Inputs not set to 0 are scaled up by 1/(1 - rate) such that the sum over all inputs is unchanged. Usually, dropout is placed on the **fully connected layers** only because they are the one with the greater number of parameters and thus, they're likely to excessively co-adapting themselves causing overfitting.

MODELS

Baseline Model

In order to better understand the problem, we decided to first try to tackle this problem from scratch, building a simple CNN using Le-Net 5 and using Relu and max pooling instead of softmax and average pooling. Our model stops training at 62 epoch and the best accuracy was at 53 epoch with training accuracy of 44% and validation accuracy of 48%.

AlexNet : After base line model we train alexnet model, this architecture was publish in 2012 and train roughly on 1.2 million high resolution images into 1000 different classes. After applying this model we got quite good accuracy on both training and validation of 69% and 66% respectively.



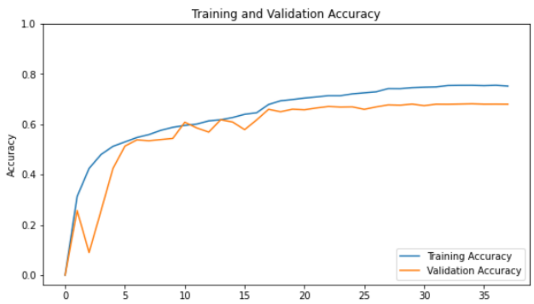
Transfer Learning

Since the FER2013 dataset is quite small and unbalanced, we found that utilizing transfer learning significantly boosted the accuracy of our model. We explored transfer learning, using the Keras library and each of VGG16, VGG19 and Resnet as our pre-trained models. To match the input requirements of these new networks which expected RGB images of no smaller than 197x197, we resized and recolored the 48x48 grayscale images in FER2013 during training time.

Fine Tuning VGG16

VGG16 is more complex and has many more parameters. We kept all pre-trained layers frozen and added two FC layers of size 4096 After 100 epochs of training with the Adam optimizer, we achieved an accuracy of 67.61% on the test set on 28th epoch and our model stops training at 37 epoch.

Loss & Accuracy Plot of VGG-16



Accuracy plot

Loss plot

RESULTS / DISCUSSION

Below table shows the accuracies our best models achieved on the FER2013

Model	Parameter	Training accuracy	Validation accuracy
Le-net	288,343	**44%**	**48%**
Alexnet	61,960,903	**69.8%**	**66.3%**
VGG-16	33,643,365	**74%**	**67%**
VGG-19	38,953,061	**68.9%**	**66%**
Mobilenet	2,396,581	**67.8%**	**62%**
Resenet152	71,351,333	**58.7%**	**59.7%**
Basic Inception	76,210,647	**56%**	**56%**

CONCLUSION

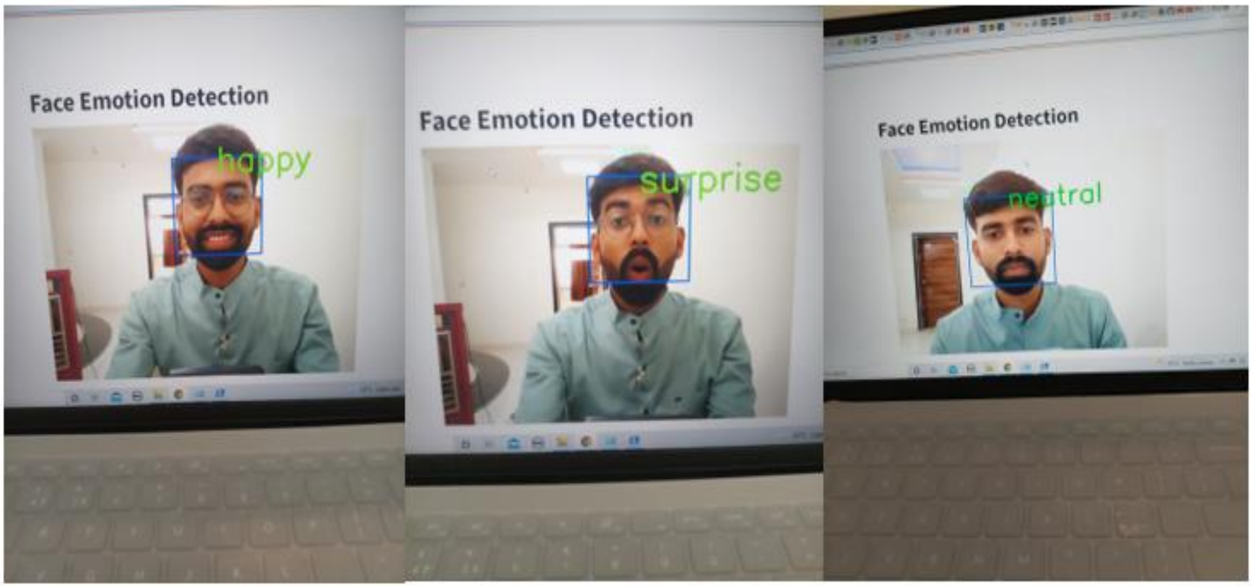
• Our model is giving an accuracy of 74% and validation accuracy of 67%

● It is robust in that it works well even in a dim light environment.

● The application is able to detect face location and predict the right expression while checking it on a local webcam as below.

● The front-end of the model was made using streamlit for webapp and running well on local webapp link.

● And we believe that through this model teachers can understand the students' perception during online classes and change the way of teaching if needed by understanding the students’ motive.



Reference :

1. Towards data science
2. Geeks for geeks
3. Machine learning mastery
4. Analytic Vidhya
5. Wikipedia