ML Report

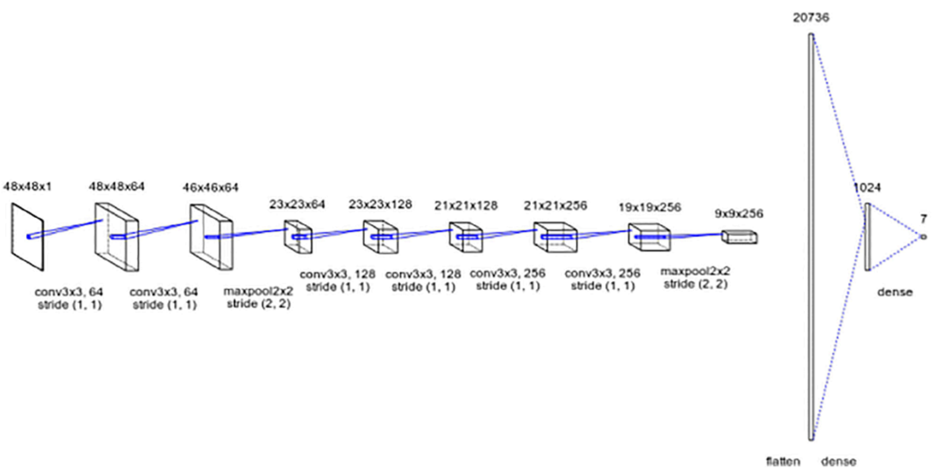
Topic: Emotion Based Music Recommendation System

Group Members:

Savitra Roy 2022A4PS1490H
Devansh Sharma 2022A3PS1298H
Lakshya Maheshwari 2022A7PS1312H
Pritish Saraf 2023A7PS1104H

Paper :Facial emotion recognition using CNNs

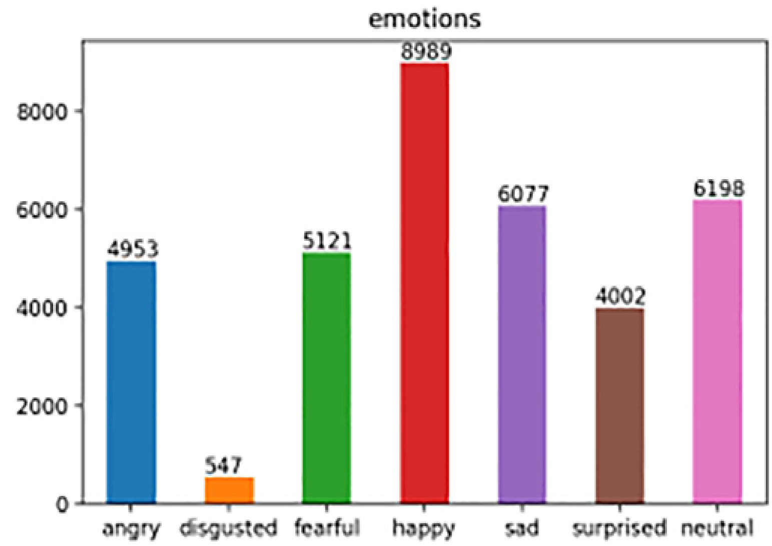
The above paper utilizes an algorithm of the CNN (convolutional neural networks) with keras library for emotion recognition . The architecture mentioned in the paper utilizes 7 Convolutional layers in sequence , MaxPooling layers and Dense layer with 1024 neurons. The regularization technique used is l2 for the dense layer and Dropout for both Convolutional layers and Dense layers.
The dataset mentioned is **FER 2013**. It has more than 35000 basic pictures: angry , outrageous, frightening , cheerful , sorrowful, neutral and surprised . The paper claims to have produced an accuracy of 54%.



Proposed CNN architecture in the Paper

RESULTS: Implementation of Paper

By implementing the CNN architecture mentioned in the paper we were able to get an **validation accuracy of 59.72%** which was higher than claimed in the paper. The **training accuracy was almost 70%** just after 15 epochs which shows overfitting . The learning rate used for the optimizer was 10e-4 and the l2 hyperparameter(kernel regularizer) was 10e-3.
The Dataset used had 7 classes and the distribution of the classes was not even , this might have been the reason for the overfitting of the model. Imbalanced Datasets lead to models that are biased towards a certain class .

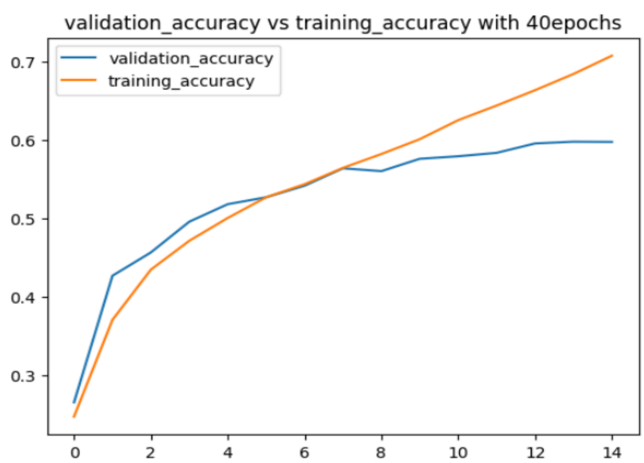


As we can clearly see here the dataset has more than 8500 instances of happy faces which is a lot more than the other classes which will lead the model to be biased towards this class. Whereas Disgusted class has only 547 instances . This class imbalance might be the reason behind the overfitting.

Overfitting is when your model performs well on the training data and does not perform that well on your testing or validation data ,basically the model does not generalize well (or we can say the model is not learning, it’s just memorizing and we want our model to learn and not memorize).

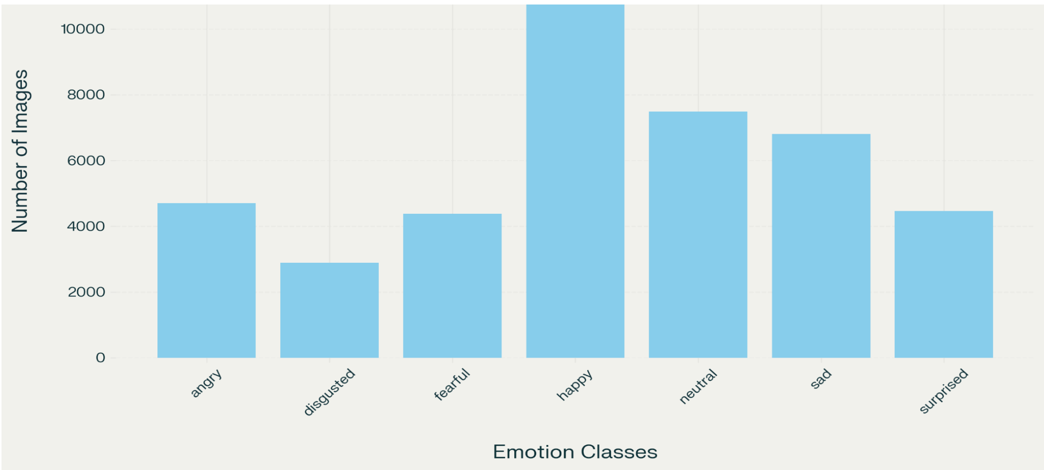
The model summary and training vs validation accuracy graph :

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_88 (Conv2D)           (None, 48, 48, 64)        640
_________________________________________________________________
conv2d_89 (Conv2D)           (None, 46, 46, 64)        36928
_________________________________________________________________
max_pooling2d_28 (MaxPooling (None, 23, 23, 64)        0
_________________________________________________________________
dropout_27 (Dropout)         (None, 23, 23, 64)        0
_________________________________________________________________
conv2d_90 (Conv2D)           (None, 23, 23, 128)       73856
_________________________________________________________________
conv2d_91 (Conv2D)           (None, 21, 21, 128)       147584
_________________________________________________________________
conv2d_92 (Conv2D)           (None, 19, 19, 256)       295168
_________________________________________________________________
conv2d_93 (Conv2D)           (None, 9, 9, 256)         590080
_________________________________________________________________
max_pooling2d_29 (MaxPooling (None, 4, 4, 256)         0
_________________________________________________________________
dropout_28 (Dropout)         (None, 4, 4, 256)         0
_________________________________________________________________
flatten_14 (Flatten)         (None, 4096)              0
_________________________________________________________________
dense_28 (Dense)             (None, 1024)              4195328
_________________________________________________________________
dropout_29 (Dropout)         (None, 1024)              0
_________________________________________________________________
dense_29 (Dense)             (None, 7)                 7175
=================================================================
Total params: 5,346,759
Trainable params: 5,346,759
Non-trainable params: 0
```



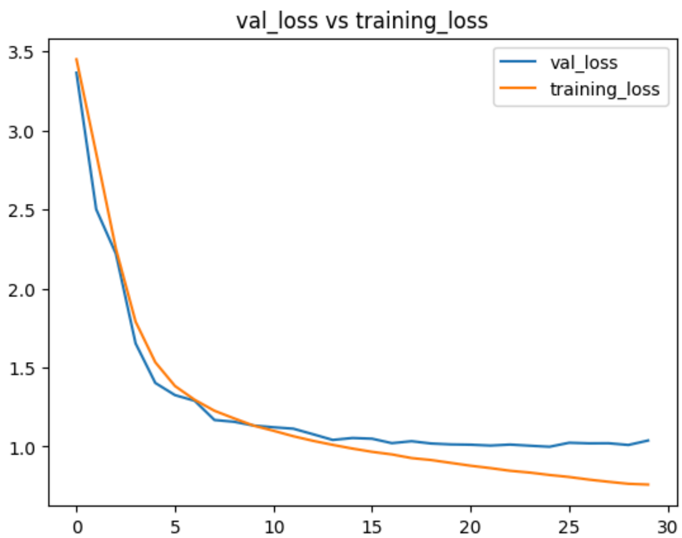
RESULTS : after some changes

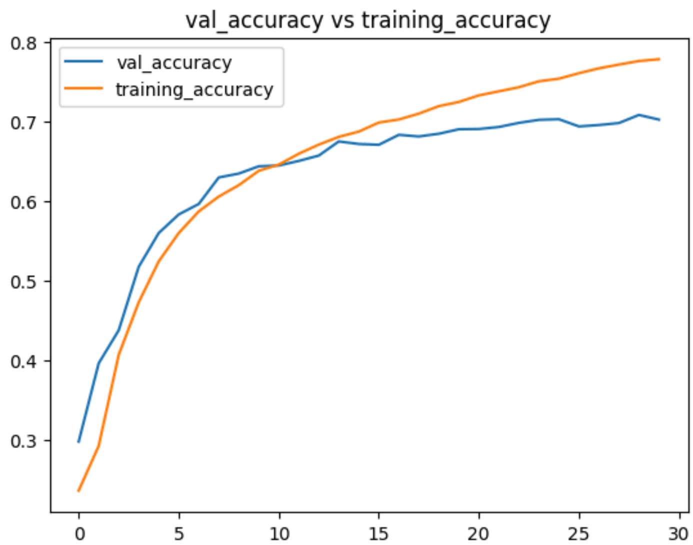
As mentioned before there was some class imbalance , so we added some more images from other datasets youngAffectNet HQ and RAF DB ,to create a sort of class balance and try to reduce the overfitting.
We reduced the strides in some ConvLayers from 2 to 1 as this would decrease the loss of information , used dropout more aggressively, increasing its value and adding it after each block of ConvLayer . We also added one more layer ConvLayer with 512 filters and with a dropout of 0.5 along with Batch Normalization to reduce the overfitting and increase the convergence for faster training. We added ConvLayer in the later layers because it's seen when you're trying to increase the accuracy the ConvLayer helps best in the later layers



Training Data Class count

Results: The performance metrics after 30 epochs were:
Validation accuracy : 70.29%
Training accuracy: 77.87%
Training loss :0.9268
Validation loss: 1.2336

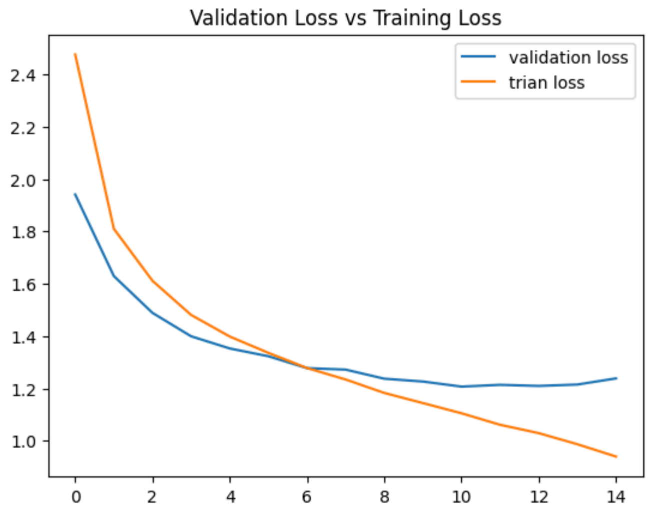


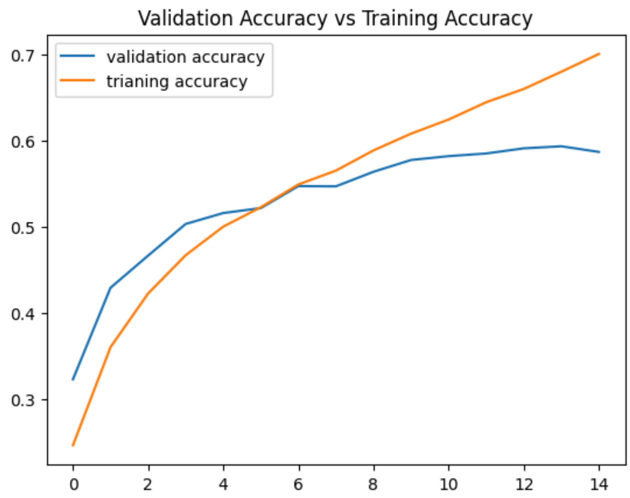


Experiments Performed

Attempt 1:(15 epochs)

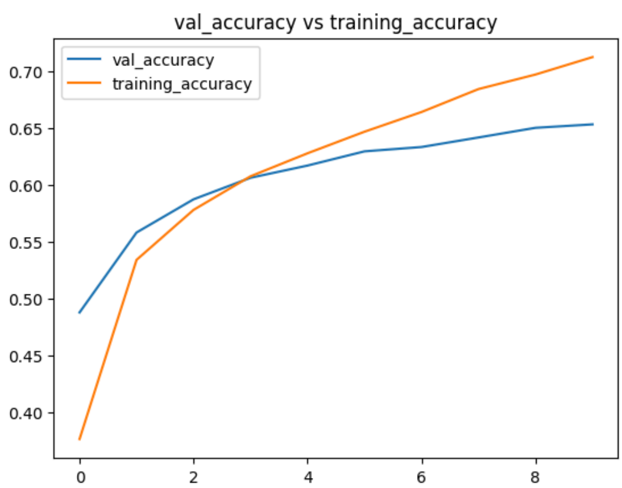
We started with trying to reduce the overfitting which was mainly caused by the class imbalance so we used Keras Data Generator to create more images from the under-represented class(disgusted) . We generated 1000 images from the existing 400 images in the training set.
By using the same model mentioned in the paper we were able to get a validation accuracy of 59.06% and training accuracy of 60.09% . Here we saw reduced overfitting as hypothesized .

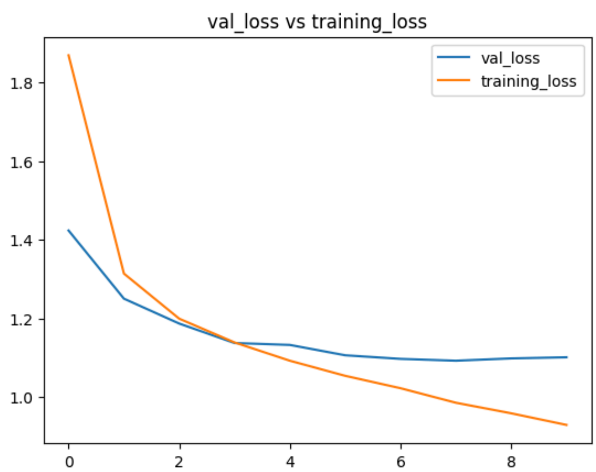




Attempt 2:(10 epochs)

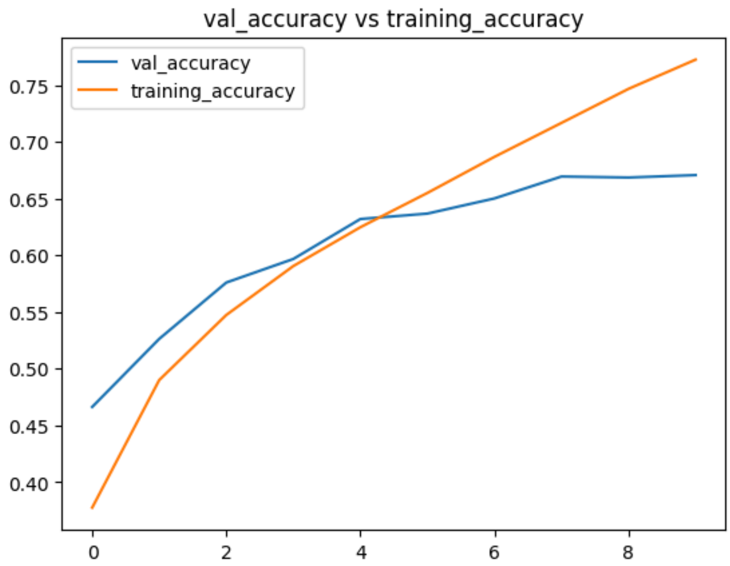
Adding more images to our dataset as it will help our model generalize well . We added all the images from RAF-DB according to the classes and just the disgusted class from the young Affect Net Dataset.
Keeping the model same here we were able to get :
Training Accuracy : 71.25%
Validation Accuracy : 65.34%
Training Loss: .9294
Validation Loss: 1.1012

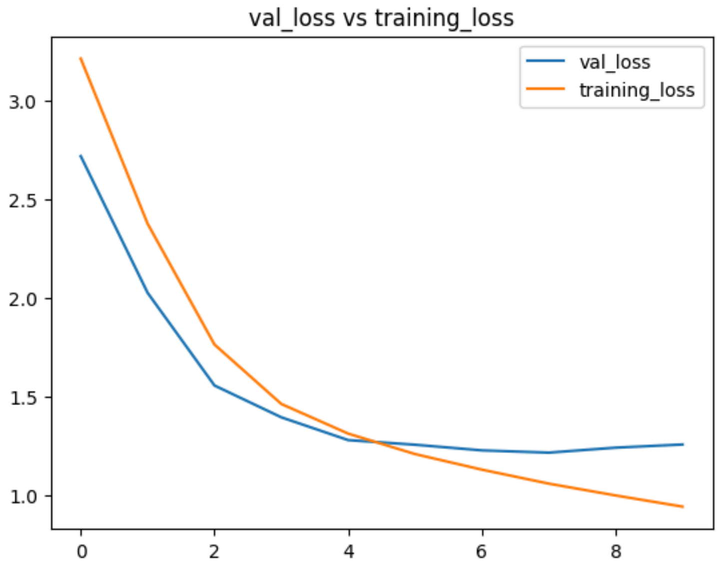




Attempt 3 :(10 epochs)

Added Batch Normalization to fight the overfitting and increase the rate of convergence .Batch Normalization was used after every 2 ConvLayers. But it didn't go as planned .The overfitting increased. This might have been caused by the batch normalization layer.
Training Accuracy : 77.26%
Validation Accuracy : 67.08%
Training Loss: .992
Validation Loss: 1.2250

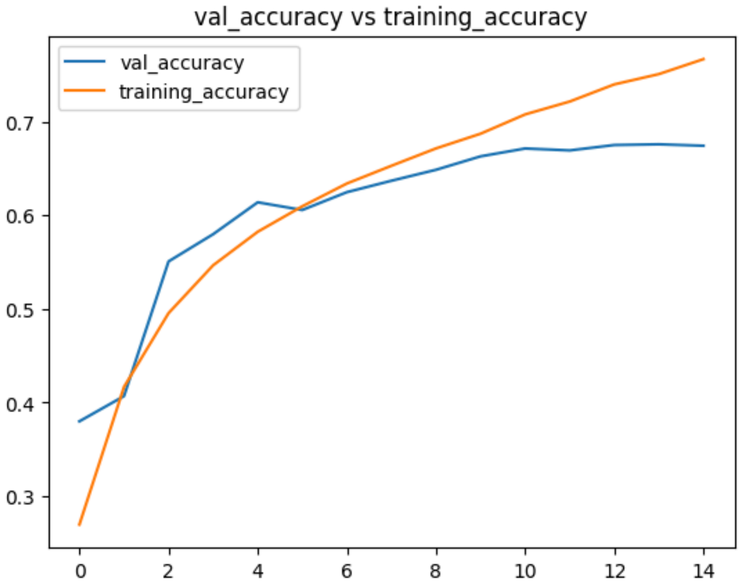


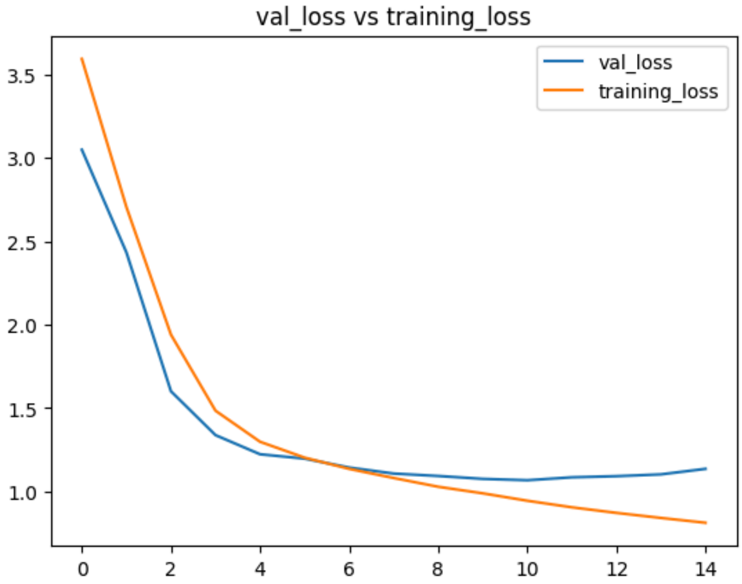


Attempt 4 :(15 epochs)

Added one more Convolutional Layer in the end with 512 filters .This increased the training parameter from around 5.3Million to 7.01 million which made the training longer.

Training Accuracy : 76.68%
Validation Accuracy : 67.43%
Training Loss: .8136
Validation Loss: .6743

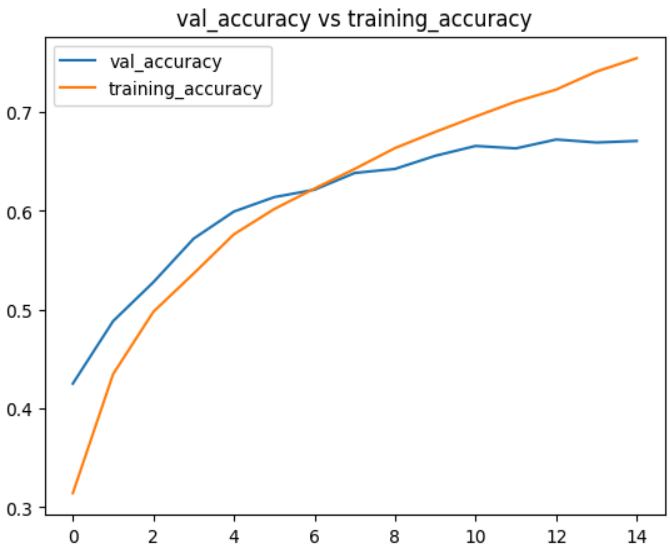


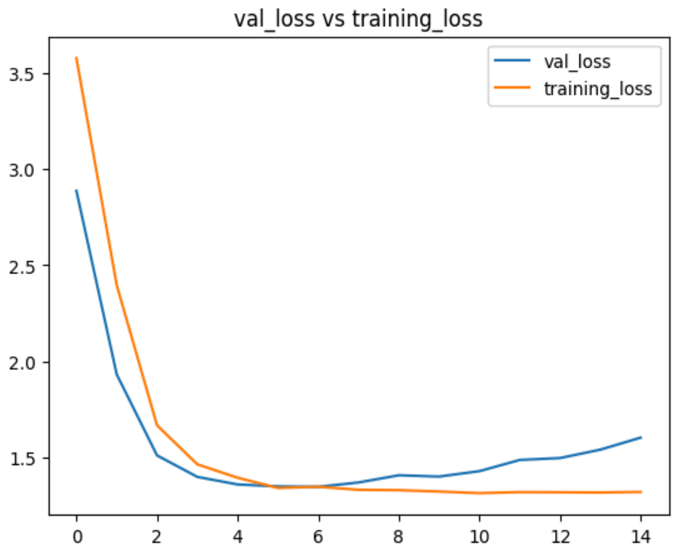


Attempt 5 :(15 epochs)

Reduced the stride in the initial ConvLayers and coupled them with MaxPooling. This led the parameters to increase from 7 Million to 28 Million We had to train it with fewer epochs because of the increase in parameters.

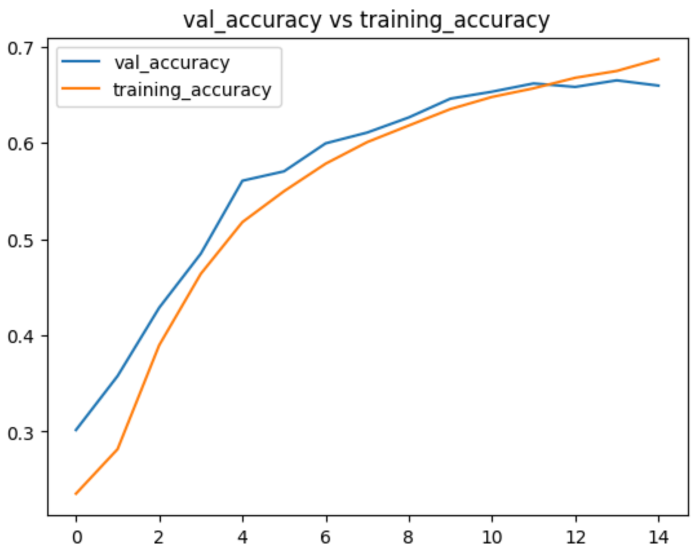
Training Accuracy : 75.46%
Validation Accuracy : 67.09%
Training Loss: 1.32
Validation Loss: 1.6026

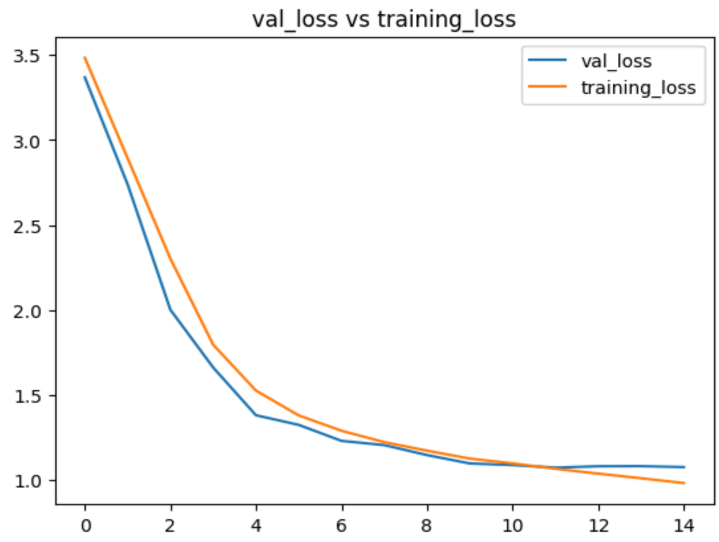




Attempt 6 :(15 epochs)
Increased and added some dropout. Added Max Pooling after each block of ConvLayer to get the most dominant facial features. Increased the strides of the Pooling layers

Training Accuracy : 68.70%
Validation Accuracy : 65..96%
Training Loss: 0.9838
Validation Loss: 0.6596

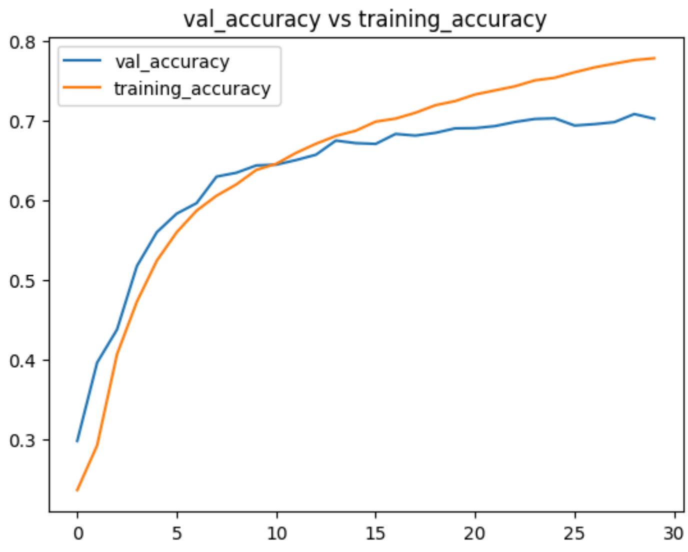


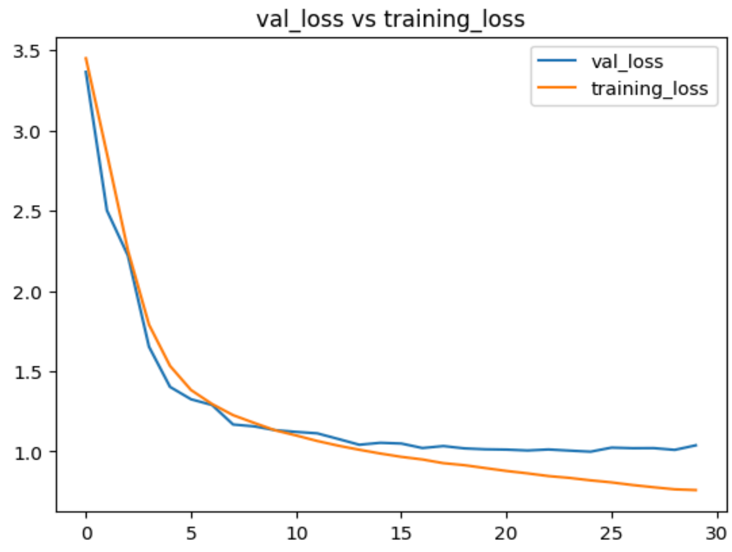


Attempt 7 :(30 epochs)

Just increased the epochs because with the 15 epochs the results were promising.

Training Accuracy :77.87%
Validation Accuracy : 70.29%
Training Loss: .7591
Validation Loss: 1.038





Findings and Accomplishments

- Class imbalance can cause overfitting and which does not let our model generalize well , which is not favourable for real world applications
- After attempt 1 we found that by increasing the instances of the under-represented classes, we were able to decrease overfitting significantly
- In order to increase the efficiency of the model you need to increase the database. Increase in the data points helps the model to generalize well.
- Found new regularization technique , BatchNormalization which helps reduce overfitting and increases the rate of convergence of the optimizer
- The accuracy of the model tends to increase when you add ConvLayers at the later layers of the pre -existing model.

	Research Paper	Group(SDLP)
Training Loss	0.9268	0.7591
Validation Loss	1.2336	1.0380
Training Accuracy	70.70%	77.87%
Validation Accuracy	59.72%	70.29%

GithubLink

Members and their contributions

Savitra Roy	Found the paper and reviewed it for the group. Performed experiment 6&7. Implemented the paper
Devansh Sharma	Performed Exp 2 and 3 . Found the DB to increase the sample size of most of the classes , for the model to perform well
Lakshya Maheswari	Performed Experiment 1 .Found ways to generate more images from existing ones in order to reduce overfitting
Pritish Saraf	Performed experiment 4 and 5 . Researched about ways to increase accuracy of the model