Summer 2021

Improving Facial Emotion Recognition with Image processing and Deep Learning

Ksheeraj Sai Vepuri

San Jose State University

Improving Facial Emotion Recognition with Image processing and Deep Learning

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

By Ksheeraj Vepuri

April 2020

The Designated Thesis Committee is pending approval on the Thesis Titled

Improving Facial Emotion Recognition with Image processing and Deep Learning

By

Ksheeraj Vepuri

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

May 2021

Dr. Nada Attar, Department of Computer Science

Dr. Robert Chun, Department of Computer Science

Dr. Fabio Di Troia, Department of Computer Science

ABSTRACT

Humans often use facial expressions along with words in order to communicate effectively. There has been extensive study of how we can classify facial emotion with computer vision methodologies. These have had varying levels of success given challenges and the limitations of databases, such as static data or facial capture in non-real environments. Given this, we believe that new preprocessing techniques are required to improve the accuracy of facial detection models. In this paper, we propose a new yet simple method for facial expression recognition that enhances accuracy. We conducted our experiments on the FER-2013 dataset that contains static facial images. We utilized Unsharp Mask and Histogram equalization to emphasize texture and details of the images. We implemented Convolution Neural Networks [CNNs] to classify the images into 7 different facial expressions, yielding an accuracy of 69.46% on the test set. We also employed pre-trained models such as Resnet-50, Senet-50, VGG16, and FaceNet, and applied transfer learning to achieve an accuracy of 76.01% using an ensemble of seven models.

***Keywords* - Facial Expression Recognition, Image pre-processing, Deep Learning, Transfer Learning, Convolution Neural Network [CNN].**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. INTRODUCTION

Computer vision allows a machine to see and perceive objects just like a human does. It is one of the major areas which have laid the foundation for artificial general intelligence. This paper will focus on Facial Expression Recognition, which is to classify a person’s emotion based on his/her facial expression. Improvement in Facial expression recognition (FER) has led to advancements in face detection, face recognition, face tracking, techniques, and cognitive detection [14]. Facial expression recognition (FER) is prominent in many fields such as human computer interaction, security, marketing, new analysis and so on [35], [36]. However, it is still a challenge to process the data and extract the features required for the analysis. To classify an expression into a number of finite expression categories with a high accuracy, computers need to learn and understand the different features required for each particular expression. In order to accomplish something like this, we would require a database that contains a large set of images with many expression features. Researchers have made significant progress on FER algorithms by mapping facial expressions directly into abstract classes [15][16][3]. Others have used data-driven deep learning models that are able to extract features and learn from a large input dataset [17][34].

The common databases used for FER analysis are divided into two categories. One type depends on recognizing basic facial expressions (e.g. happiness, sadness, surprise, anger, disgust, fear, and neutral) from human facial expression in RGB or greyscales images [36], [37]. The other type focuses on extracting fine-grained descriptions for facial expressions [38]. Both types have two issues, which is either the limited number of images in the dataset, or the acted expression instead of a spontaneous one in highly controlled environments. Both issues make it difficult for learning models to effectively classify expressions.

Convolutional Neural Networks (CNNs) have been extensively used for image classification tasks, especially FER, due to their ability to extract image features [31][19] [21] [26] [29]. Generally, training CNN model for FER may have some challenges. For example, it may result in overfitting on uncertain inputs that could be mislabeled. Moreover, a high ratio of incorrect labels can make the model unable to converge during the early stage of optimization.

One of the deep learning methods is called transfer learning, which has gained a lot of interest in deep learning. It uses a pre-trained CNN to solve a problem that is similar to a problem that the CNN has been originally trained to solve. It has been used in many FER studies [24][20][4]. Oztelet al, conducted an experimental study that compared transfer learning vs training the CNN from scratch using popular networks (alexnet and vgg16) [18]. They found that the transfer learning approach resulted in shorter training time and better results. Ngo and Yoon proposed a loss function called as weight-clustered loss, which is a novel loss function that they used during the fine-tuning phase [23]. This technique assisted them in handling the imbalance in a facial expression dataset by assigning each emotion class a weight based on its proportion of the total number of images.

Transfer Learning optimizes performance by freezing the pre-trained layers, which means that the layers will not be trained, and the weights will not be changed. This technique saves computation time, as the network does not need to learn how to extract generic features from scratch. However, by not updating the weights of most of the network, the optimization only occurs in a subset of the feature space. Thus, the freezing technique can decrease the accuracy of the dataset is not similar to any subset of the dataset. On the other hand, unfreezing the pre-trained network can improve accuracy. However, this process requires caution, as it can result in either over-fitting or decreasing the accuracy in the latter process [1]. Common measures to consider when deciding to unfreeze the network are usually a few epochs and a small learning rate to avoid over-fitting.

In this study, we avoid the over-fitting problem while still unfreezing the network to train the entire model. The first iteration was performed by freezing the entire pre-trained network by detaching their classification layer. Then, we added two fully connected layers and trained them for a few epochs. To still take advantage of a pre-trained model on a large number of images, unfreezing the last 20-30 layers of the pre-trained network was performed in the second iteration. We used a small learning rate and trained the network for fewer epochs. This strategy fine-tuned the parameters without training the network.

II. Project Roadmap

The first part of the project involves experimenting with Image preprocessing filters. We believe that applying preprocessing techniques can result in improvement in the model performance. We have seen how histogram equalization led to an improvement in accuracy from previous research [29]. The goal is to experiment with other Image preprocessing filters such as Image sharpening, Image smoothing, Adaptive histogram equalization, Contrast Limited Adaptive Histogram Equalization etc. Image sharpening enhances the edges of the prominent features such as eyes and mouth which play a major role in classifying the emotion. We will evaluate the difference in performance by feeding the preprocessed images to a 5-layer CNN.

The second part of the project involves applying transfer learning on the pre-trained models in order to achieve better results. We used pre-trained models such as Resnet-50, Senet-50, VGG-16, and FaceNet which were trained on millions of facial images. Our goal is to fine-tune these pre-trained models on the FER2013 dataset and finally use an ensemble to achieve the best results.

III. RELATED WORK

Previous studies have developed different methods with increasing progress in facial emotion recognition performance [17], [19]. Conventional classification has shown its robustness when it is preceded by image preprocessing techniques [7]. Rani et al. [39] used edge detection algorithms for prefiltering the raw images for FER. Other studies used Gaussian edge detectors [40], Colored edge detectors [41] for multi-view face detection, and Canny edge detection [42] for feature extraction. A study by Yu et. al [43] used standard histogram equalization for preprocessing facial images data. In unconstrained environments, obtained a sparse representation of faces for person-specific verification. The datasets available for FER are limited and consist of a small number of images. Training CNNs on such small datasets may result in overfitting. To overcome this problem, researchers have started adopting transfer learning for facial recognition tasks. Moreover, overall performance of the system can be improved by using auxiliary data. This often helps model achieve a higher capacity without over-fitting. Applying transfer learning to the FER models achieved better results, as opposed training networks from scratch on small FER datasets[27].

Though CNNs perform well on their own, performing image preprocessing and feeding the preprocessed image as input to the CNN has shown significant improvement in accuracy as opposed to feeding a raw input image [29]. Earlier implementation of CNN architectures did not employ image data augmentation and preprocessing techniques [10], [11], [26] making them less robust to rotated and deviated facial images. Tang [12] implemented CNN with a linear Support Vector Machine (SVM). Wang et al. [29]

employed the same technique, but that they used SVM to stack the result of the “softmax” activation function. They also used data augmentation with histogram equalization, which resulted in better performance. The performance of an ensemble of classifiers can be boosted by applying various preprocessing techniques and feature extract parameters. Nanni et al. [44] showed that it is possible to further boost performance by designing an ensemble of classifiers based on different preprocessing techniques and feature extraction parameters.

Existing datasets available for facial recognition have been used in fine-tuned networks databases such as FER+ and RAF-DB datasets [4], (RaFD) [18], CK+, JAFFE and FACES [b2], and FER2013 [2]. The accuracy of the CNN models to detect facial expression in the FER-2013 dataset was 65%. The highest accuracy in the Kaggle competition was 71.2% which was achieved using CNN with SVM as the loss function for training [12]. Many researchers have used pre-trained models to improve the accuracy on the FER2013 dataset. Pramerdorfer et al. achieved an accuracy of 75.2% using an ensemble of 6 models and without using auxiliary data or face registration [25]. Zhang et al. achieved an accuracy of 75.1% using auxiliary data, HoG features, and facial landmark registration [33]. Khanzada and colleagues achieved an accuracy of 75.8% using auxiliary data and an ensemble of 7 CNNs [28]. In their study, they applied transfer learning on 5 pre-trained networks and trained two CNNs from scratch. This is the highest reported accuracy to the best of our knowledge. In our method, we used Khanzada et al. findings as a base for our approach in training pre-trained models.

Our goal is to combine the best of both methods, using an image preprocessing technique with data augmentation to enhance each image features and feed it to the CNN model. We found that Image Sharpening technique enhances the prominent features of the input facial images. Our CNN architecture is tantamount to [29] with some additional changes in the convolutions and normalization, and without the need to stack SVM with the CNNs output. We used class weight in Resnet-50, Senet-50, and VGG-16 models and created an ensemble model. We also combined CNNs such as Resnet- 50, Senet-50, VGG-16 with image processing filters such as Unsharp Mask and Histogram equalization resulting in an ensemble model with an accuracy of 76.01%. Our recent method achieved the highest accuracy on the FER2013 with no auxiliary data.

IV. DATASET DESCRIPTION

FER2013 is a well-established dataset used for facial expression recognition tasks (see Figure 1). It was developed by Goodfellow et al. [13] and introduced in a Kaggle competition with the intention to promote researchers to improve FER systems. Google Image search API was used for the generation of this dataset. It consists of 28,709 training images, 3589 validation images, and 3589 test images. The “emotion” field from the dataset is the target attribute which consists of six emotions namely “Anger”, “Disgust”, “Fear”, “Happy”, “Sad”, “Surprise”, and “Neutral”, with labels 0 to 6 respectively. The dataset distribution is depicted in Table 1. The class emotions are not all equally distributed. There are only 547 facial images of category “Disgust”.

Table 1: Distribution of target class labels

Emotion type	**Number of images**	**Label**
Angry	4593	0
Disgust	547	1
Fear	5121	2
Happy	8989	3
Sad	6077	4
Surprise	4002	5
Neutral	6198	6

There is a “pixel” field that consists of a 48*48 facial images. It is stored as a flattened 1-dimensional string. Some of these images are considered noisy. For example, the comic images, images with occluded facial expressions, and completely black images. For comparison of results with previous research, none of these images were excluded from training.

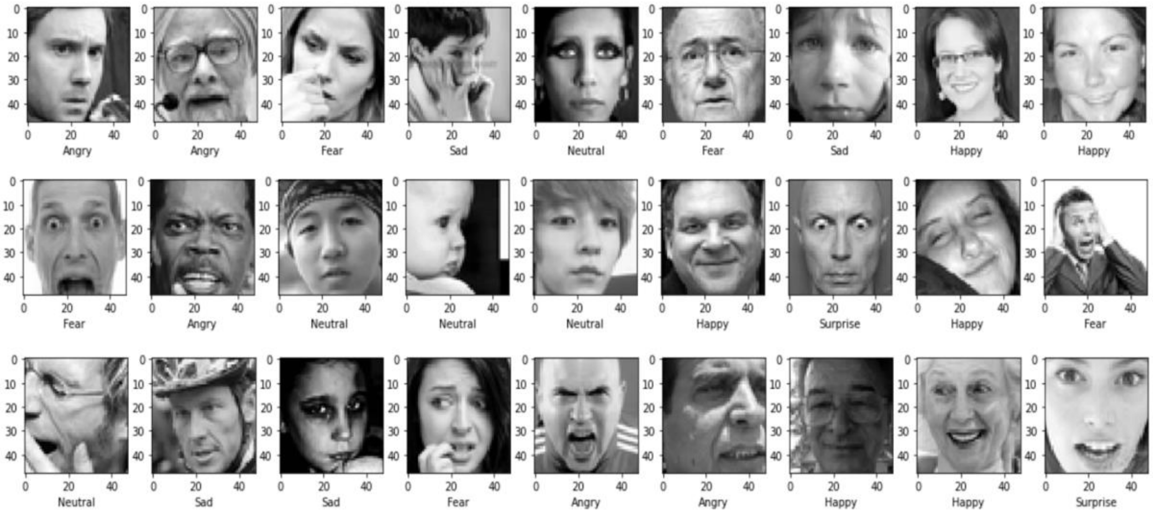


Figure 1: Sample images from the FER-2013 dataset

V. Facial Image Preprocessing

There are multiple factors that can affect the performance of CNNs such as the cluttered background, illumination, and posture deviation. Applying preprocessing filters can potentially result in improved accuracy in classifying facial expressions. For example, sharpening the images can enhance the edges of important features such as mouth and eyes [7]. These edges are essential in predicting facial expressions. Histogram equalization helps in differentiating the foreground from the background when both the colors are identical [32]. In our experiments, we use three pre-processing techniques mainly used in facial expression recognition models.

A. Data Augmentation

The first is data augmentation, which was applied as a preprocessing step to all our models. We used “ImageDataGenerator” by Keras for data augmentation. It generates 32 augmented images from one image by rotating, flipping, and applying other specified techniques as shown in Figure 2.

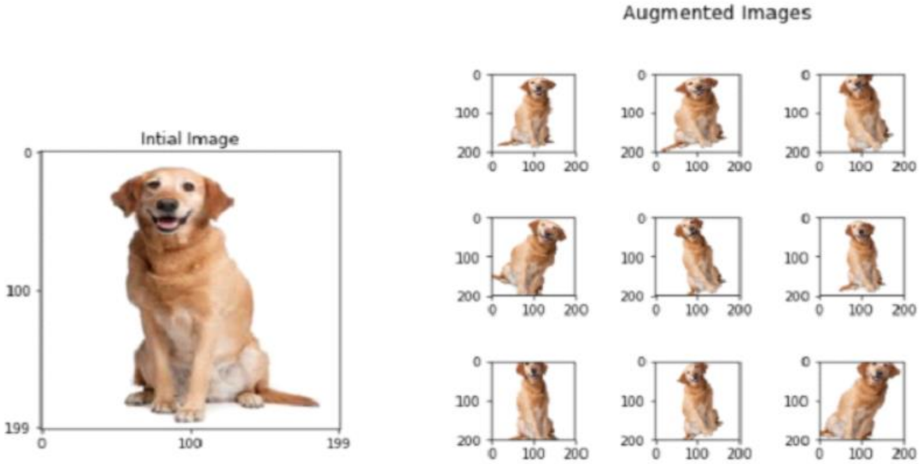


Figure 2: Data augmentation output (Patidar, P. (2020, November 20). Image data Augmentation- image processing IN TensorFlow- Part 2. Retrieved April 28, 2021, from https://medium.com/mlait/image-data-augmentation-image-processing-in-tensorflow-part-2-b77237256df0)

B. Image Sharpening

The second is image sharpening, for which we used “Unsharp Mask” filter from PIL(Python Imaging Library) package. Unsharp Mask uses a blurred or negative image to create a mask of the original image. The final positive “less blurred” image is obtained by combining the original image with the blurred image. The advantage of using Unsharp Mask over other sharpening filters like Gaussian High Pass is the ability to control the sharpening process. Unsharp Mask provides adjustable parameters which can be modified. By observing and understanding the images in FER-2013, we found that the images is slightly blurry and applying sharpening to these images helped to define the edges of prominent features, such as eyes and mouth which are relevant for detecting human emotions as shown in Figure 3.

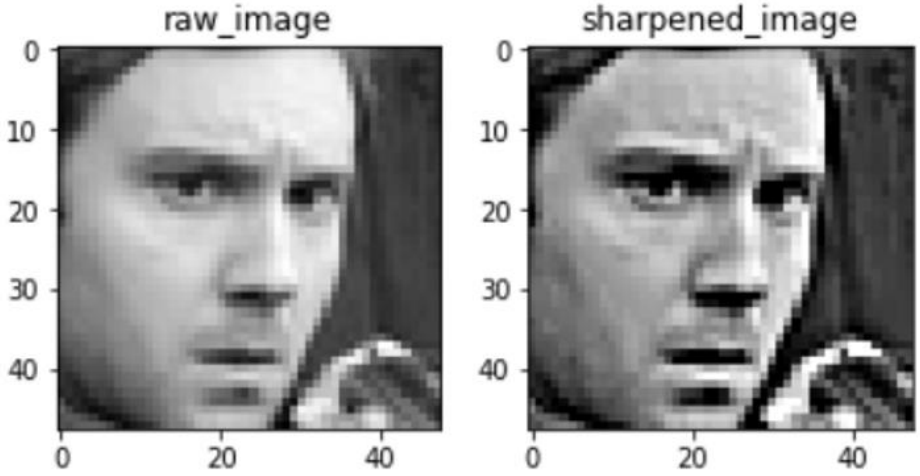


Figure 3: raw image vs sharpened image

C. Histogram Equalization

The third is histogram equalization, which we used “skimage” package. We applied sharpening technique prior to feeding the images to our baseline 5-layer CNN. The Histogram equalization was applied to the raw images as shown in Figure 4.

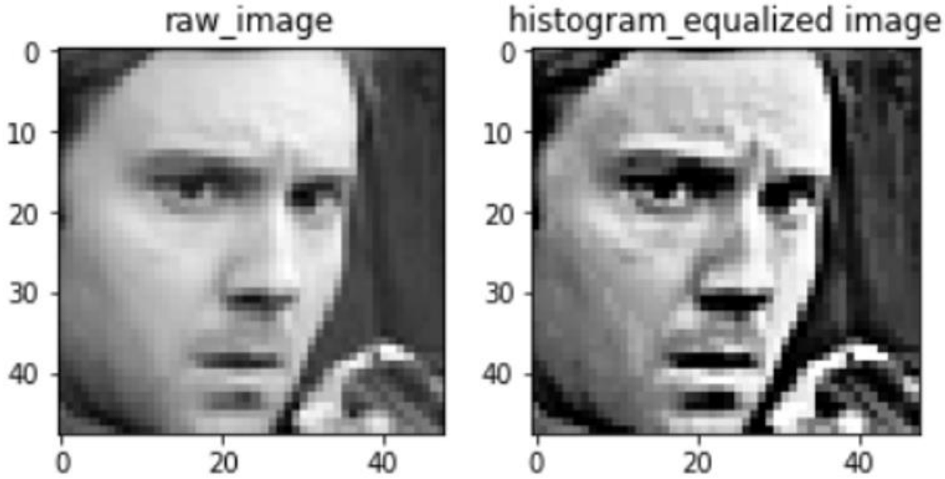


Figure 4: raw image vs histogram equalized image

However, this training technique was applied to only two models, the 5-layer CNN and Resnet-50, in the final ensemble among the seven models. The remaining five models Resnet-50, Resnet-50 with class weights, Senet-50, Senet-50 with class weights, and FaceNet were trained only by using Data Augmentation. This mask is useful for specific applications which require pixel level details.

VI. FACIAL EMOTION RECOGNITION

A. FER with traditional Machine Learning

In the traditional Machine Learning pipeline for Facial Emotion Recognition, face detection is performed first to determine the region of interest. Well known face detection algorithms like Haar-cascades and Histogram of oriented gradients (HoG) are used to identify the face in the image. The second step is to extract important features from the region of interest. Feature detection algorithms are used to identify facial landmarks in the bounding box as shown in the Figure 5.

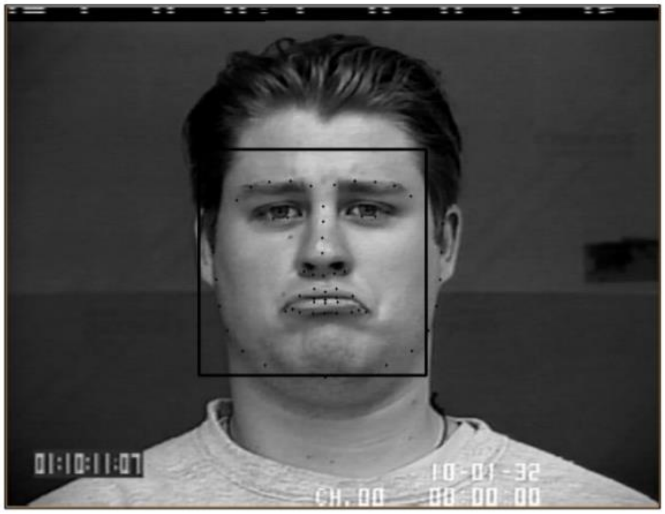


Figure 5: Facial landmark detection (Raut, Nitisha, "Facial Emotion Recognition Using Machine Learning" (2018). Master's Projects. 632. DOI: https://doi.org/10.31979/etd.w5fs-s8wd)

These features are then fed as input to Machine Learning classification models like SVM to identify the emotion category as shown in Figure 6.

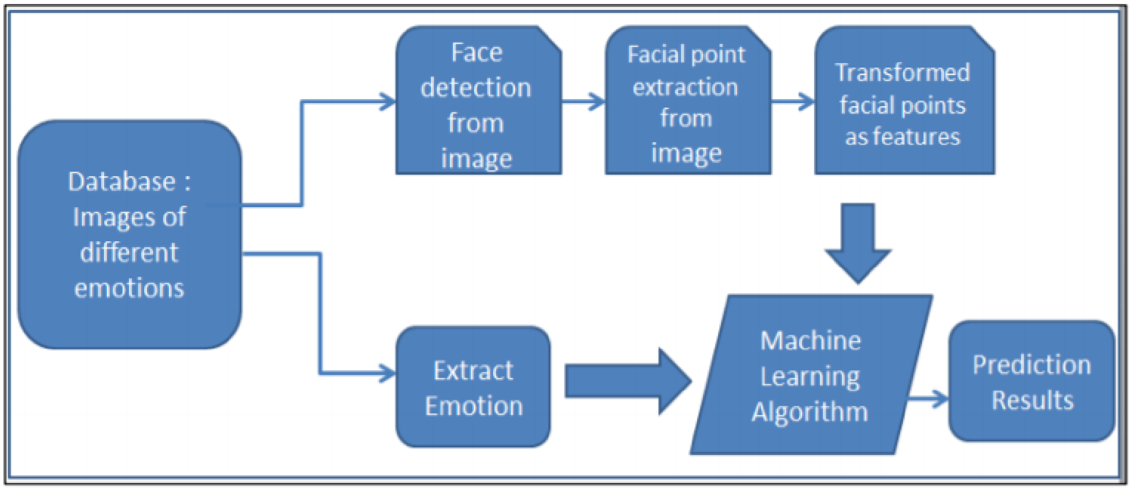

Database : Images of different emotions
Face detection from image
Facial point extraction from image
Transformed facial points as features
Extract Emotion
Machine Learning Algorithm
Prediction Results

Figure 6: Traditional Machine Learning pipeline for FER (Raut, Nitisha, "Facial Emotion Recognition Using Machine Learning" (2018). Master's Projects. 632. DOI: https://doi.org/10.31979/etd.w5fs-s8wd)

B. FER with Deep Learning

As opposed to traditional Machine Learning where we manually extract facial features from an input image to perform classification of emotions, in Deep Learning features are automatically extracted by the Neural Network. One such variation of neural networks, known to capture spatial information and extract useful features from an image is a Convolutional Neural Network.

A Convolutional Neural Network requires minimal preprocessing. It takes an image as the input and assigns weights to different aspects in the image. From Figure 7, we can see that a convolutional Neural Network typically consists of Convolutional layers followed by BatchNormalization and pooling layers followed by Fully Connected layers in the end.

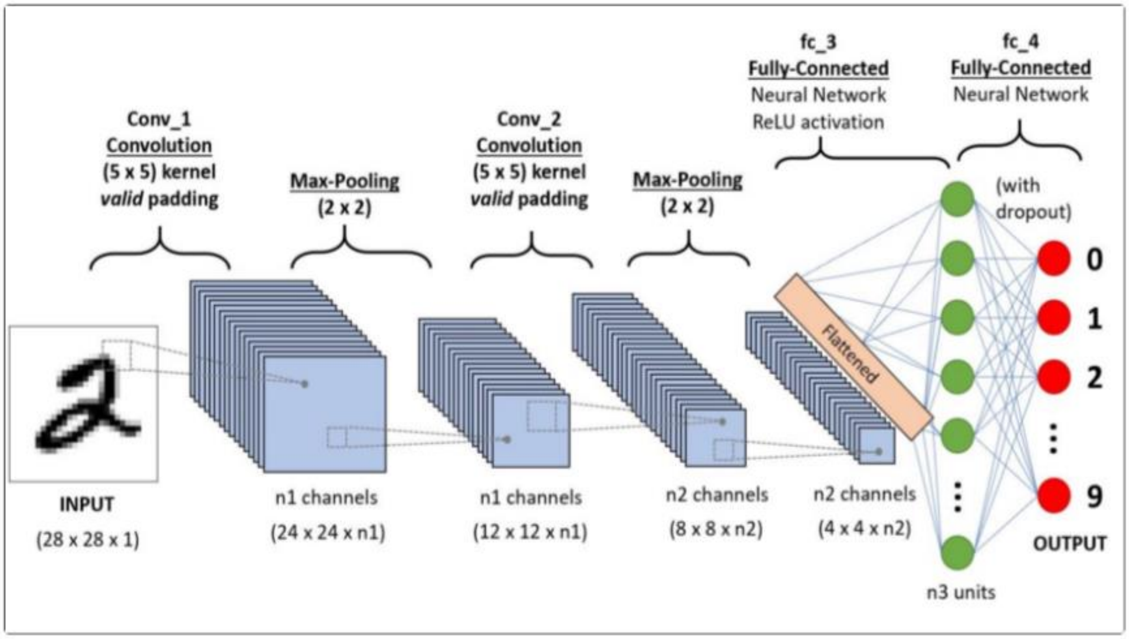
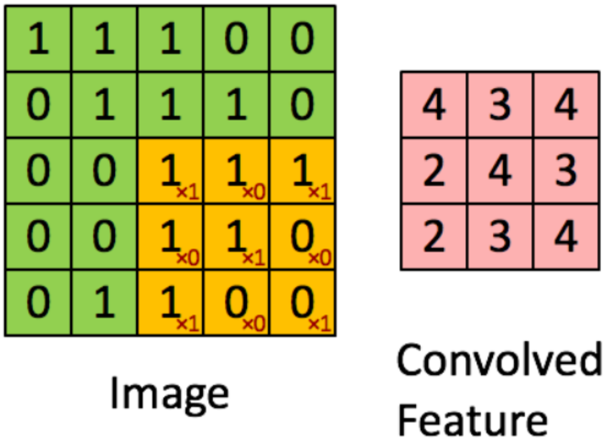


Figure 7: A typical Convolution Neural Network architecture (Saha, S. (2018, December 17). A comprehensive guide to convolutional neural networks - the eli5 way. Retrieved April 28, 2021, from https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53)



Convoluting a 5×5×1 image with a 3×3×1 kernel to get a 3×3×1 convolved feature

Figure 8: 3x3x1 kernel operation on 5x5x1 image (Saha, S. (2018, December 17). A comprehensive guide to convolutional neural networks - the eli5 way. Retrieved April 28, 2021, from https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53)

1) Convolutional Layer

A convolution layer is where a kernel is applied on the input with a certain stride to obtain the convolved feature (see Figure 8). A kernel is essentially a set of weights initialized randomly and tweaked by the network during backpropagation based on the loss function. When the kernel is applied on the input, a dot product is performed between the kernel weights and the region it is being applied on. Therefore, the convolved feature is the sliding dot product of the input image from top to bottom as shown in Figure 9.

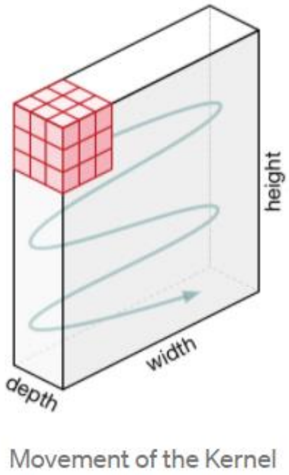


Figure 9: Movement of the kernel (Saha, S. (2018, December 17). A comprehensive guide to convolutional neural networks - the eli5 way. Retrieved April 28, 2021, from https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53)

The kernel has the same depth as the input it is being applied on. For a grayscale image, the depth of the kernel is 1 whereas for a color image with three channels (RGB – Red, Green, and Blue), the depth of the kernel would be 3. Therefore, three kernels would be applied to three different channels (see Figure 10). In Figure 10, we can observe how three different kernels with different set of weights are being applied on the three channels of the image. The dot product of all the three channels is summed up with bias to obtain the feature output.

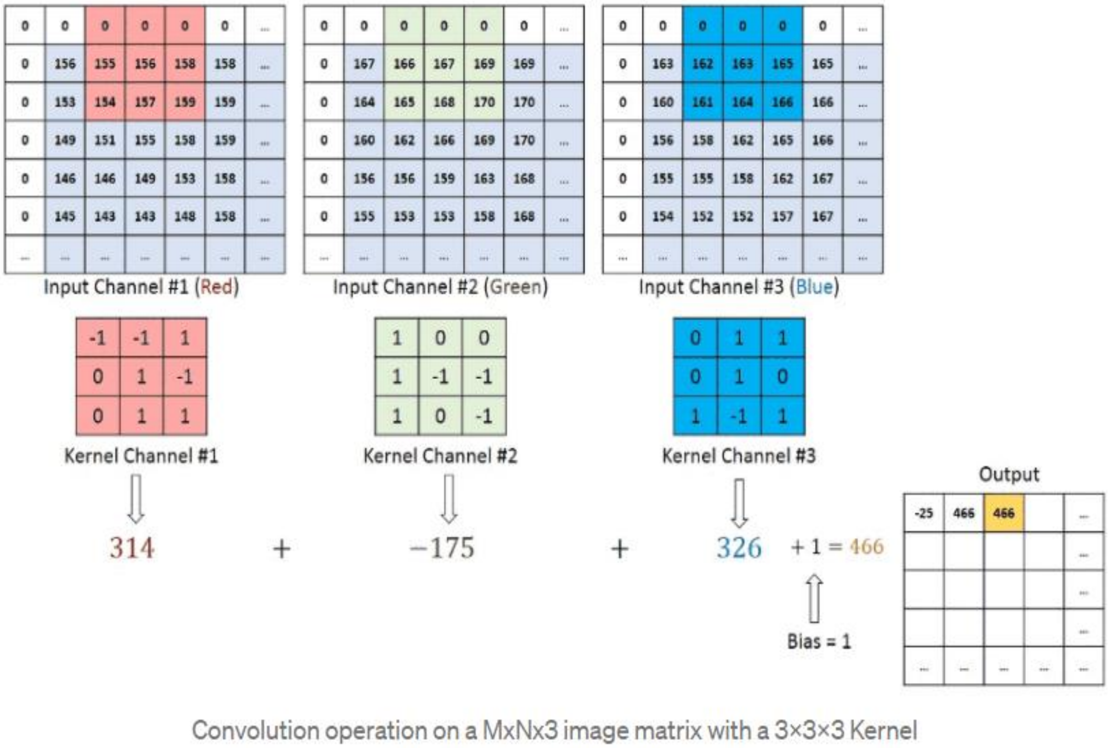
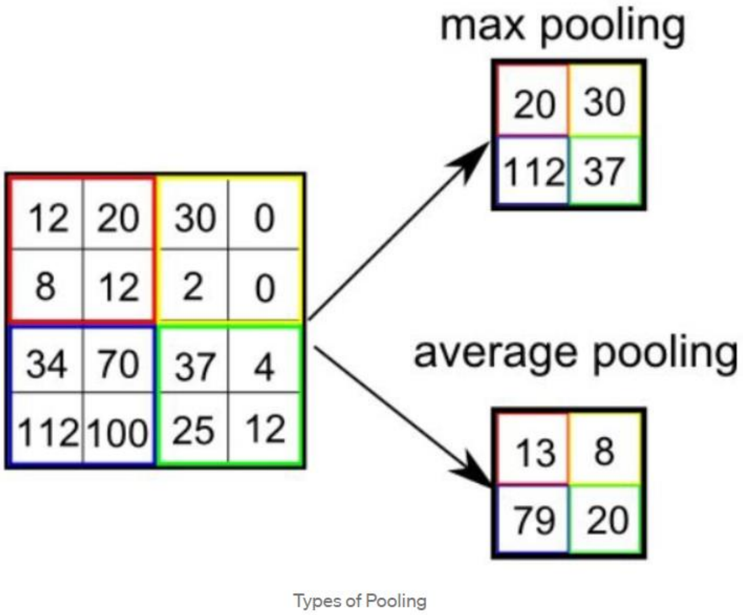


Figure 10: Kernel operation on input with depth > 1 (Saha, S. (2018, December 17). A comprehensive guide to convolutional neural networks - the eli5 way. Retrieved April 28, 2021, from https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53)

There can be multiple Convolutional layers in a CNN. The initial Convolutional layers extract the low-level features of an image such as edge, color and gradient orientation. The deeper Convolutional layers extract more complex high-level information such as shapes, objects etc. When we perform a convolution, we also choose the type of padding. There are two types of padding called as “same padding” and “valid padding”. If we do not intend to reduce the dimensionality of the input, then “same padding” can be performed which pads all sides of the image with zeroes. On the other hand, if we do not want to reduce the dimensionality of the input, “valid padding” can be selected.

2) Pooling Layer

A Pooling Layer is applied after the Convolutional layer to reduce the dimensionality and capture the most relevant information. There are two types of pooling that are commonly used: Max Pooling and Average pooling. Max pooling only selects the maximum value of the region on which the kernel is applied. Average pooling computes the average of all the values of the region on which the kernel is applied. Max pooling is heavily used as it also eliminates noise along with reducing the dimensions as we discard the less useful information. Figure 11 shows the result of a 2x2 max pooling and a 2x2 average pooling with a stride of 2 being applied an input feature map.



12	20	30	0
8	12	2	0
34	70	37	4
112	100	25	12

max pooling

20	30
112	37

average pooling

13	8
79	20

Types of Pooling

Figure 11: max pooling and average pooling on input feature map (Saha, S. (2018, December 17). A comprehensive guide to convolutional neural networks - the eli5 way. Retrieved April 28, 2021, from https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53)

3) Fully Connected Layer

To capture the non-linear properties of the features obtained by the convolutional layers, we use fully connected layers after the convolutional layers. The output is flattened before feeding it to the Fully connected layers. To perform classification, we typically use a Softmax layer right after the fully connected layers.

Different variations of Convolutional Neural Networks have been developed overtime with the intent to reduce training time, improve accuracy, and deal with the vanishing gradient problem that is common in Deep Convolutional Neural Networks. Let us look at a few models used in this project.

C. VGG-16

VGG-16 was the winner of the ImageNet competition in 2014. It uses a 3x3 filter with a stride of 1 and padding as “same”. It uses a 2x2 filter for max pooling with a stride of 2. It uses 13 convolutional layers followed by 2 fully connected layers and a softmax layer (see Figure 12). It has about 138 million parameters due to which the training time is much higher compared to other CNNs.

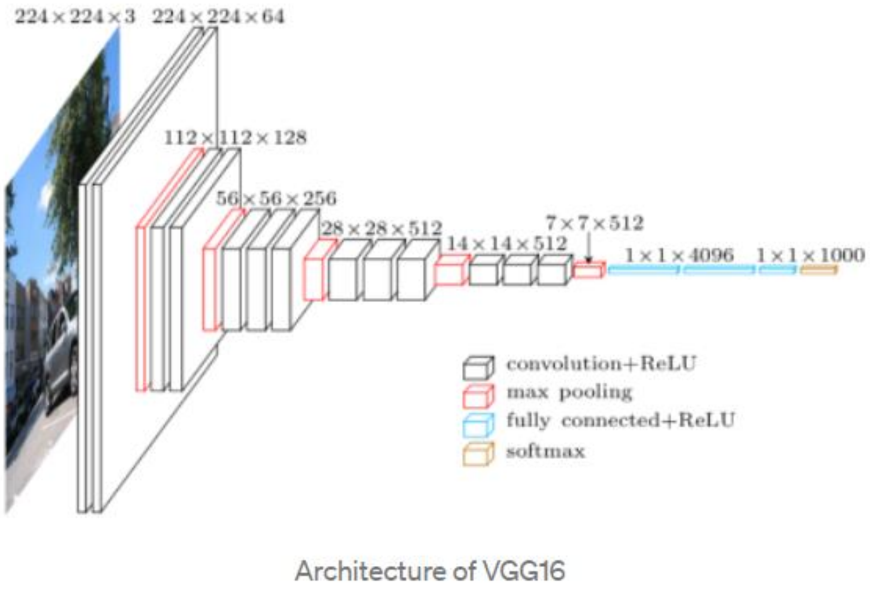


Figure 12: VGG Architecture (Thakur, R. (2020, November 24). Step by step VGG16 implementation in Keras for beginners. Medium. https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners a833c686ae6c#:%7E:text=VGG16%20is%20a%20convolution%20neural,competition%20in%202014.&text=It%20follows%20this%20arrangement%20of,by%20a%20softmax%20for%20output.)

D. Resnet-50 (Residual Networks)

Resnet-50 was the winner of the ImageNet competition in 2015. Resnet-50 made it possible to drain really deep CNNs while avoiding the vanishing gradient problem. It achieved this with the use of “skip connections”. Skip connections are responsible for adding the original input of previous convolutions to the output of the current convolution. This provides a shorter path for the gradient to flow while backpropagation. Another advantage is that this identity function enables the model to learn useful features in the higher layers as well. Figure 13 depicts the Resnet-50 architecture.

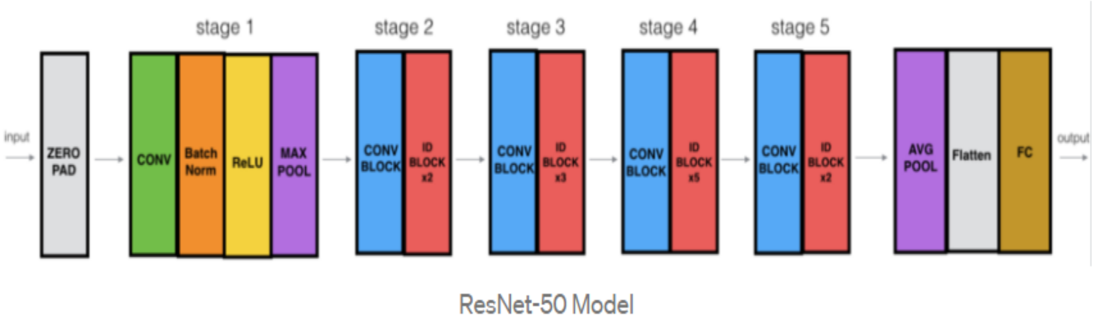


Figure 13: Resnet-50 architecture (Dwivedi, P. (2019, March 27). Understanding and Coding a ResNet in Keras - Towards Data Science. Medium. https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras 446d7ff84d33#:%7E:text=The%20ResNet%2D50%20model%20consists,over%2023%20million%20trainable%20parameters.&text=Our%20ResNet%2D50%20gets%20to,in%2025%20epochs%20of%20training.)

E. SeNet – 50 (SqueezeNet)

SqueezeNet was developed by researchers at UC Berkeley, DeepScale, and Stanford University. SqueezeNet uses a smaller CNN architecture while retaining the performance. Firstly, it replaces 3x3 filters with 1x1 filters, which results in 9x fewer parameters. Secondly, it decreases the number of input channels to 3x3 using Squeeze layers (see Figure 14). Finally, it performs delayed downsampling (uses downsampling late in the network) in order to get large activation maps.

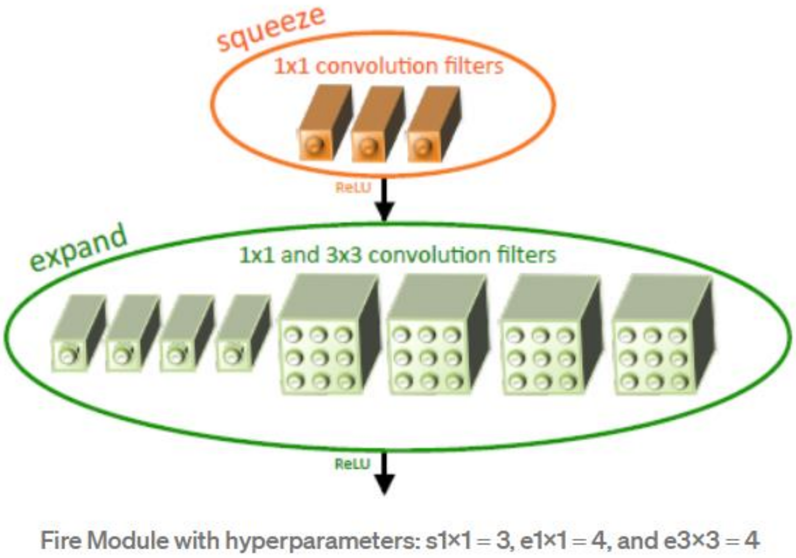
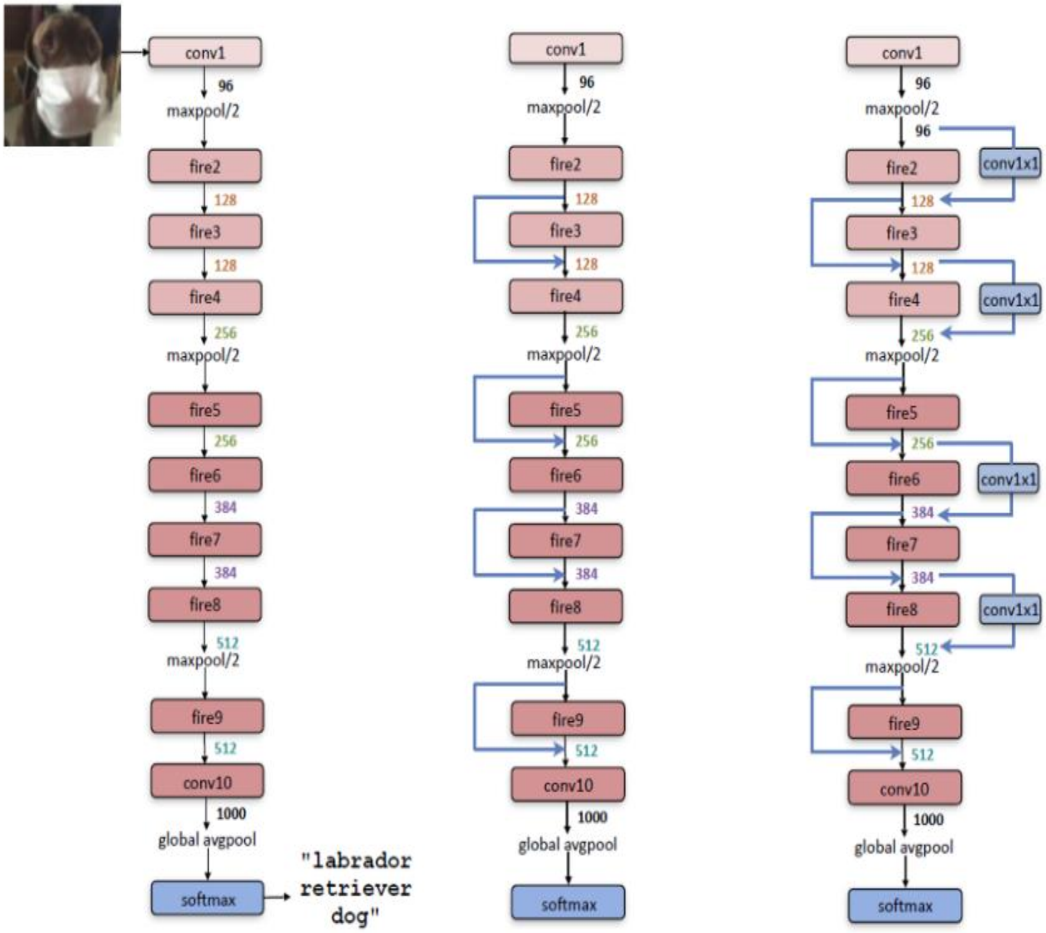

squeeze
1x1 convolution filters
ReLU
expand
1x1 and 3x3 convolution filters
ReLU
Fire Module with hyperparameters: s1×1 = 3, e1×1 = 4, and e3×3 = 4

Figure 14: Fire modulde in SqueezeNet – Sqeeuze+expand (Tsang, S. (2019, April 22). Review:SqueezeNet (Image Classification) - Towards Data Science. Medium. https://towardsdatascience.com/review-squeezenet-image-classification-e7414825581a)

SqueezeNet comprises of multiple FireModules as shown in Figure. Each Fire Module is composition of a Squeeze Convolutional layer with 1x1 filters and an expand layer which consists of both 1x1 and 3x3 filters. The complete Squeeze architecture can be seen in Figure 15.



SqueezeNet (Left), SqueezeNet with simple bypass (Middle), SqueezeNet with complex bypass (Right)

Figure 15: SqueezeNet architecture (Tsang, S. (2019, April 22). Review: SqueezeNet (Image Classification) - Towards Data Science. Medium. https://towardsdatascience.com/review-squeezenet-image-classification-e7414825581a)

F. FaceNet

FaceNet is one of the state-of-the-art models for Face recognition. It creates embeddings of the image by mapping each image into a Euclidean space. These embeddings can then be used as feature vectors and fed to models like k-NN for face recognition. Clustering techniques can also be applied on the embeddings. FaceNet uses a loss function called as the triplet loss function. It makes sure that the distance between the positive image and the anchor image is as small as possible and the distance between the negative image and the anchor image is as large as possible. This principle is demonstrated in Figure 16 below.

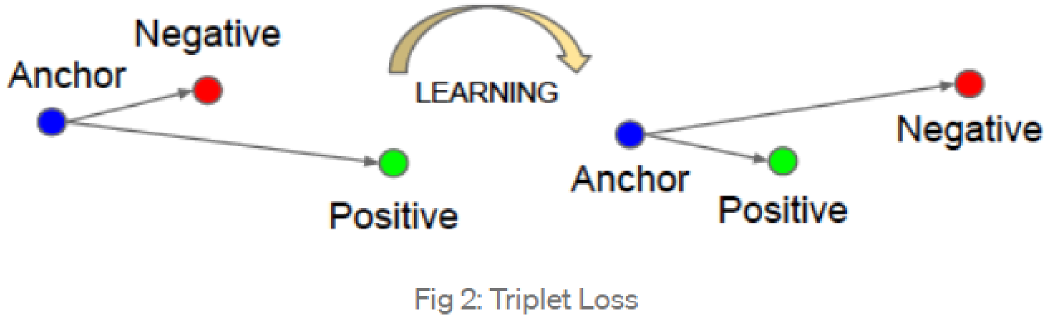


Figure 16: working of Triplet loss (Kumar, D. (2020, June 21). Introduction to FaceNet: A Unified Embedding for Face Recognition and Clustering. Medium. https://medium.com/analytics-vidhya/introduction-to-facenet-a-unified-embedding-for-face-recognition-and-clustering-dbdac8e6f02)

VII. Proposed Models

We trained one 5-layer CNN. We fine-tuned three pretrained models: Resnet-50, Senet-50, and VGG-16, which were available from keras-vggface library [8]. We performed two stages of fine-tuning for each of these pretrained models. In the first iteration, the entire network is frozen and only the fully connected layers are trained.

In the second iteration, the entire network or the last 30-40 layers is unfrozen depending on the depth of the network and fine-tuned with a small learning rate of 0.0001. We used the VGGFace library that provides Resnet-50 and Senet-50 models. Those models were trained previously on VGGFace2 dataset that consists of 3.3 million facial images. The library also provides the VGG16 model, which was trained on VGGFace dataset. The VGGFace consists of 2.6 million facial images. Another model we fine-tuned is a pre-trained model called FaceNet [9][5]. FaceNet was trained previously on MS-Celeb-1M dataset [5]. Since these three models were trained on millions of facial images, all of them must have learned to capture the essential information or embeddings from a face which can be helpful in recognizing facial expressions. We applied transfer learning on these four models, Resnet-50, Senet-50, VGG-16, and FaceNet, and customized their weights based on the FER2013 dataset.

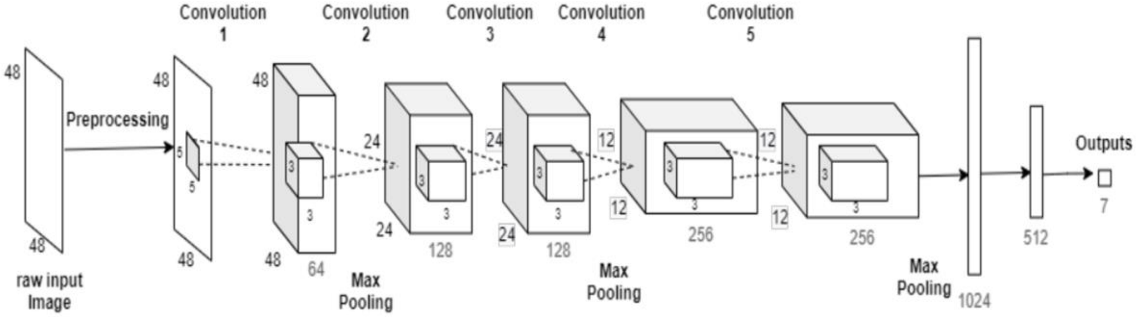


Figure 17: 5-layer CNN architecture

A. Baseline 5-layer CNN

The grey-scale input images of size 48*48 for 5-layer CNN were obtained by the ImageDataGenerator, which produced rescaled, rotated, flipped, and sharpened images. The output of the CNN was the probability of seven categories of facial expressions. The model was composed of five convolutional layers, three max-pooling layers, and three fully connected layers as shown in Figure 17. All the convolutional layers used the same padding and ReLU activation function. The first two fully connected layers used ReLU as their activation function. The last fully connected layer used "Softmax" activation function. The first, third, and fifth convolutional layers were followed by batch normalization, max-pooling of size 2x2, stride 2x2, and dropout of 20%. The first convolutional layer was composed of a 5x5 kernel with 64 filters. The second and third convolutional layers were composed of 3x3 kernels with 128 filters. The fourth and fifth convolutional layers were composed of 3x3 kernels with 256 filters.

The first fully connected layer consisted of 1024 neurons. The second fully connected layer consisted of 512 neurons followed by a drop out of 20%. The final fully connected layer which produced the output consisted of 7 neurons representing the 7 categories. The model used the categorical crossentropy loss function and adam optimizer. The total trainable parameters in the model were 11, 075, 847 and non-trainable parameters were 896. Keras callback ReduceLROnPlateau was used to modify the learning rate if the validation accuracy does not increase for every 10 epochs. The model was trained for a total of 100 epochs using a batch size of 128. Applying Unsharp Mask as the preprocessing function in ImageDataGenerator resulted in around a 2% increase in accuracy. The 5-layer CNN with Data Augmentation and sharpening resulted in 69.6% accuracy. Figures 18 and 19 depict the training accuracy vs validation accuracy and training loss vs validation loss of the 5-layer CNN.

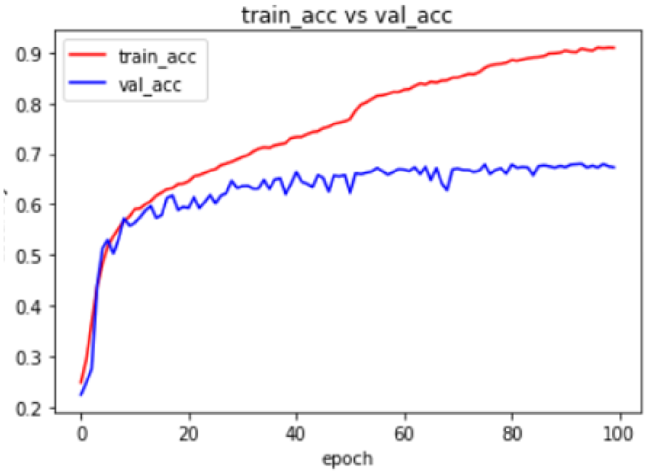


Figure 18: train accuracy vs validation accuracy - 5 Layer CNN

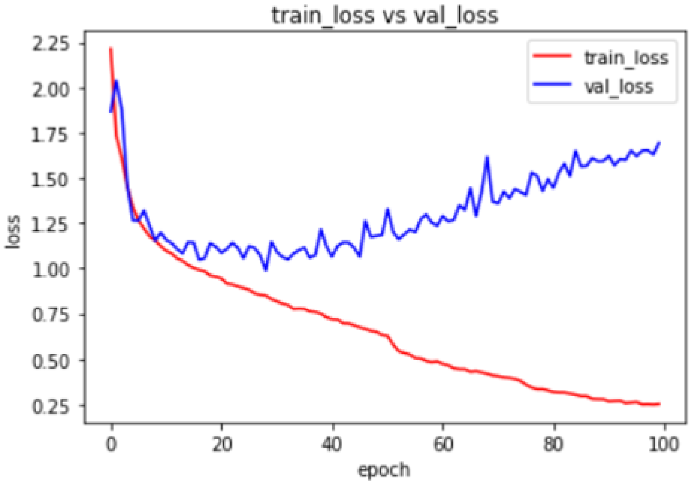


Figure 19: train loss vs validation loss - 5 Layer CNN

B. *Resnet – 50*

We used Resnet-50 is a deep residual network that was originally trained on 240x240 images. As FER2013 dataset consists of images with dimensions 48x48, resizing a 48x48 image to 240x240 cannot be ideal. The network breaks if we use a width or height lesser than 197 as the dimension becomes smaller than the applied convolution. Therefore, we reduced the Resnet-50 input dimensions to 197x197 [28]. We froze the entire network except for the Batch Normalization (BN) layers. The BN layers represent the statistics captured on VGGFace2 model. We modified the BN layers to have the mean, standard deviation, and parameters corresponding to FER2013 [6]. Then, we added two fully connected layers with 4,096 and 1,024 neurons. We added a drop out of 50% after each of these fully connected layers, and before the first fully connected layer. We used Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.01, momentum of 0.9, decay of 0.0001, and nesterov set to True. Keras callback "ReduceLROnPlateau" was used to reduce the learning rate by a factor of 0.5 with patience 10. The model was trained for 100 epochs with a batch size of 128 resulting in an accuracy of 72.8% without fine-tuning the entire network.

After completing the first iteration of training, we unfroze the entire network. As Keras's implementation of Resnet-50 consists of 175 layers, we used SGD optimizer with a very small learning rate of 0.0001 and trained the entire network for 100 epochs with a batch size of 128. We used Keras callback early stopping to stop training if validation accuracy did not increase for 20 epochs. This technique can boost the accuracy by 1.5% in most cases, but it must be monitored cautiously to avoid the risk of over-fitting when training the entire network. Our model achieved an accuracy of 73.8%. The transfer learning approach we used is summarized in Figure 20.

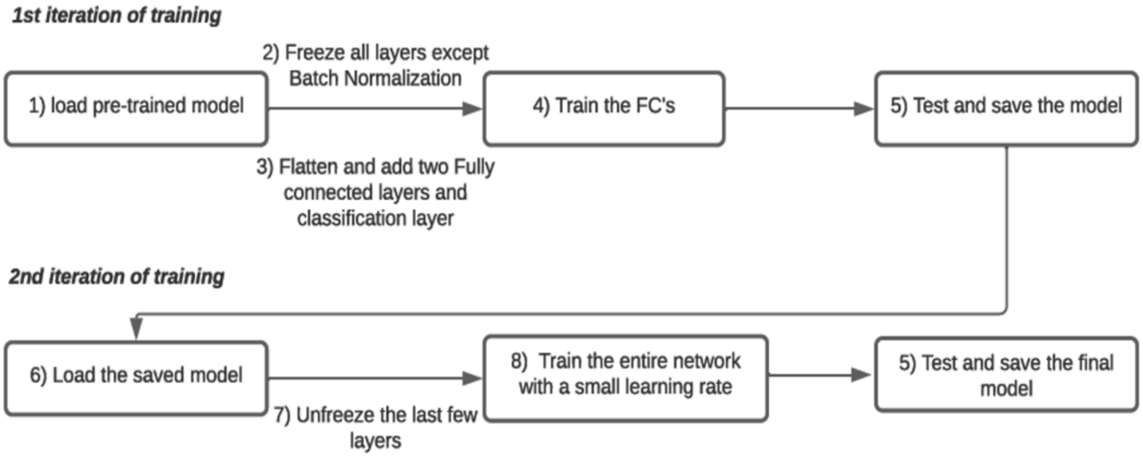

1st iteration of training
1) load pre-trained model
2) Freeze all layers except Batch Normalization
3) Flatten and add two Fully connected layers and classification layer
4) Train the FC's
5) Test and save the model
2nd iteration of training
6) Load the saved model
7) Unfreeze the last few layers
8) Train the entire network with a small learning rate
5) Test and save the final model

Figure 20: Transfer learning approach pipeline

C. Resnet-50 with class weights

Keras class weights help to solve the class imbalance problem. The classes are not equally distributed in the FER2013 dataset, hence class weight is a useful method to emphasize the minority classes. When we applied this method to our model, it reduced the misclassification rate from 60% to 28% comparing with the experiments conducted in [28]. We used the same approach and trained Resnet-50 as in section 5.2 with added class weights.

In the first iteration, we froze the entire network except for BN layers and trained the last two fully connected layers. In the second iteration, we unfroze the entire network and fine-tuned the model with a small learning rate which achieved an accuracy of 73.4%.

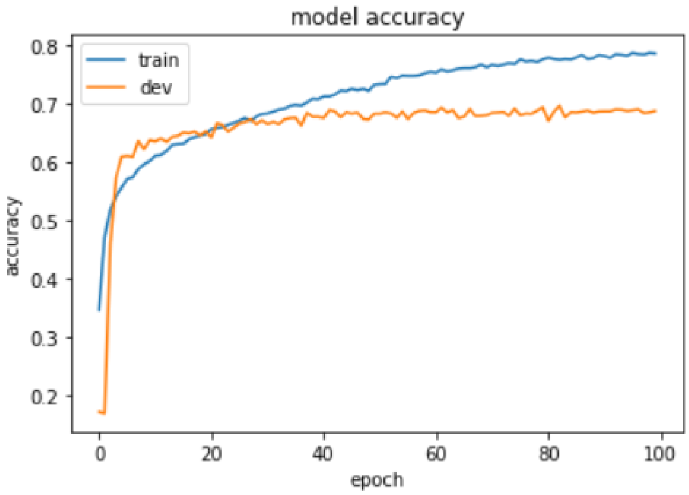


Figure 21: train accuracy vs validation accuracy - Resnet-50

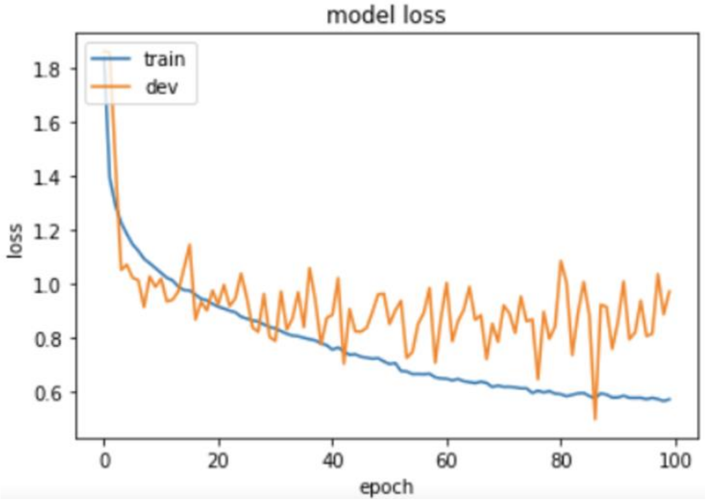


Figure 22: train loss vs validation loss – Resnet-50

D. *Resnet-50 with Image preprocessing filters*

In this method, we applied two image preprocessing filters: Unsharp Mask and histogram equalization to Resnet-50, because it demonstrated improved performance while training the baseline 5-layer CNN. By applying Unsharp Mask, we achieved test accuracy of 73.3% on Resnet-50. By applying Histogram equalization, we achieved an accuracy of 72.4% on Resnet-50. While the accuracy is lesser than the Resnet-50 that was trained without any image preprocessing filters, the classification rate for emotion such as “Anger” was increased. This improvement in the performance is due to the fact that these filters enhance important facial details in an image.

For the baseline 5-layer CNN, we observed an improvement in performance when the image preprocessing filters were applied. In this case, the images were not resized as the input to the CNN is a 48*48 image. However, all the pre-trained models required the images to be resized to higher width and height. Keras’s ImageDataGenerator resizes the image first and then applies the preprocessing filter which might have caused a decline in model performance. Given limited compute resources, we decided not to experiment further with this. In future work, we want to experiment with these filters again by first applying them and then resizing the image.

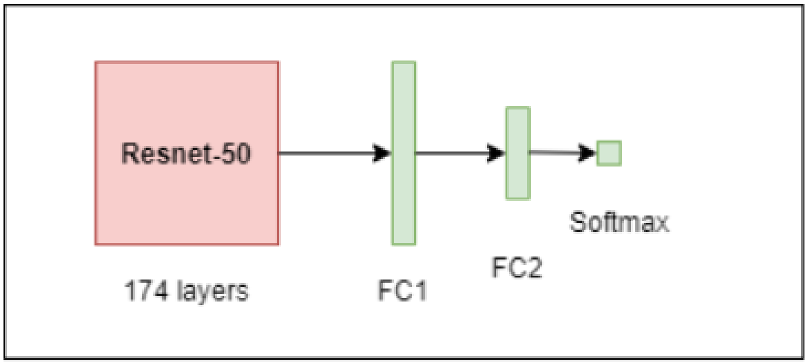


Figure 23: Resnet-50 architecture (layers include Batchnormalization and pooling)

E. Senet-50 without class weights

The same approach that was used for training Resnet-50 was employed for Senet-50. The input images were resized to 197x197, the entire network was frozen except BN layers, and 2 fully connected layers were added with a dropout of 50%. SGD optimizer with a learning rate of 0.01, the momentum of 0.9, and decay of 0.0001 was used with nesterov set to True. The model was trained for 100 epochs using a batch size of 128 resulting in a test accuracy of 70.02%.

Keras’s implementation of Senet-50 consists of 286 layers. In the second iteration of training, instead of unfreezing the entire network, we kept the first 200 layers frozen as there were too many model parameters resulting in “Out of memory” exception on Google Collab pro. The layers starting from 200 were unfrozen and fine-tuned along with the fully connected layers for 100 epochs with a batch size of 128, the learning rate of 0.0001, and early stopping enabled. We achieved a test accuracy of 73.3% post-finetuning which is almost a 3% boost in accuracy as opposed to no fine-tuning.

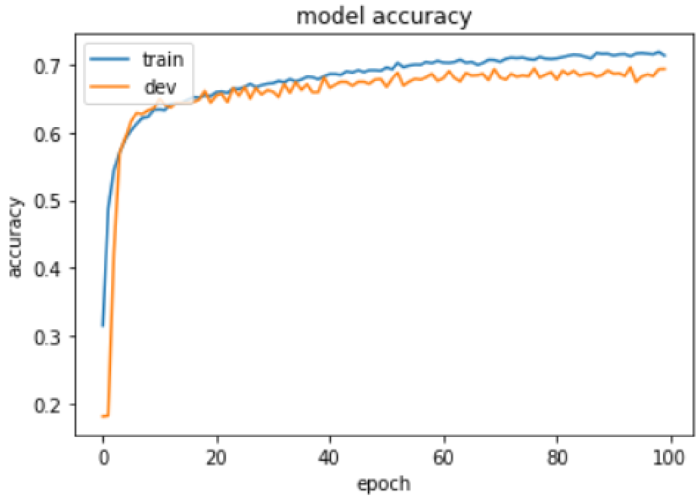


Figure 24: train accuracy vs validation accuracy - Senet-50

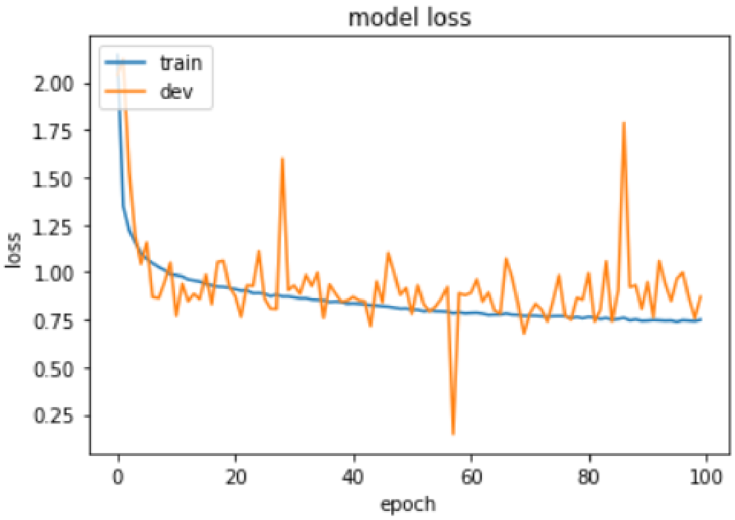


Figure 25: train loss vs validation loss-Senet-50

F. Senet-50 with class weights

Keras class weights were used to emphasize the minority classes. Senet-50 with Keras's class weights resulted in a test accuracy of 72.4%. Since the usage of image preprocessing filters in ImageDataGenerator degraded the performance of Resnet-50 and due to limited compute resources, we decided not to proceed with this model.

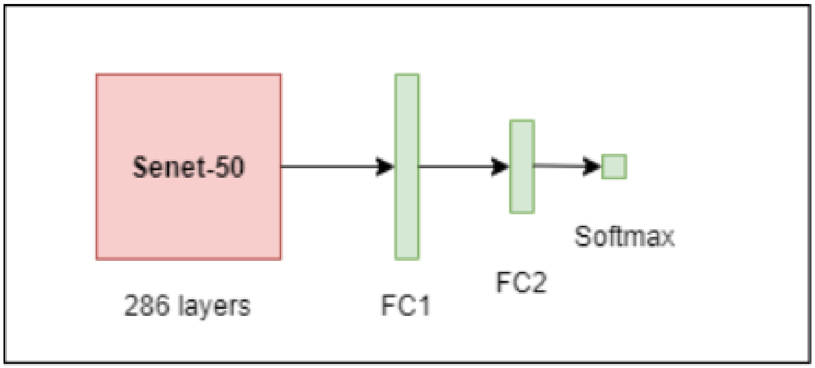


Figure 26: Senet-50 architecture

G. VGG-16 without class weights

VGG-16 was trained in the same way as Resnet-50 and Senet-50. Unlike Resnet-50 and Senet-50, VGG-16 does not have any BN layers. Keras's implementation of VGG16 consists of 19 layers. We froze the entire network, added 2 fully connected layers with drop-out layers in the same pattern as in Resnet-50 and Senet-50. The dropout was reduced to 0.3 due to VGG16 being a smaller network. The model was trained for 100 epochs using a batch size of 128 and SGD optimizer resulting in an accuracy of 69.5%.

In the second iteration of training, the entire network was unfrozen, the learning rate of 0.0001 was used, and the entire network was fine-tuned for 100 epochs with a batch size of 128 and early stopping enabled. This resulted in a test accuracy of 71.2%.

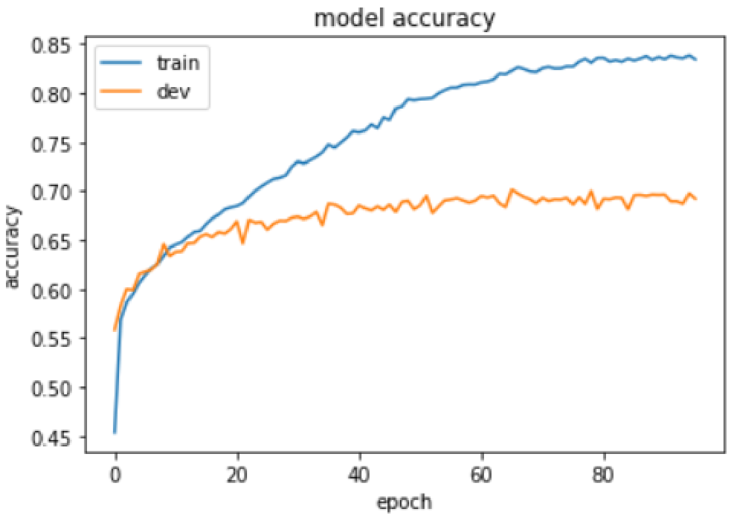


Figure 27: train accuracy vs validation accuracy - VGG-16

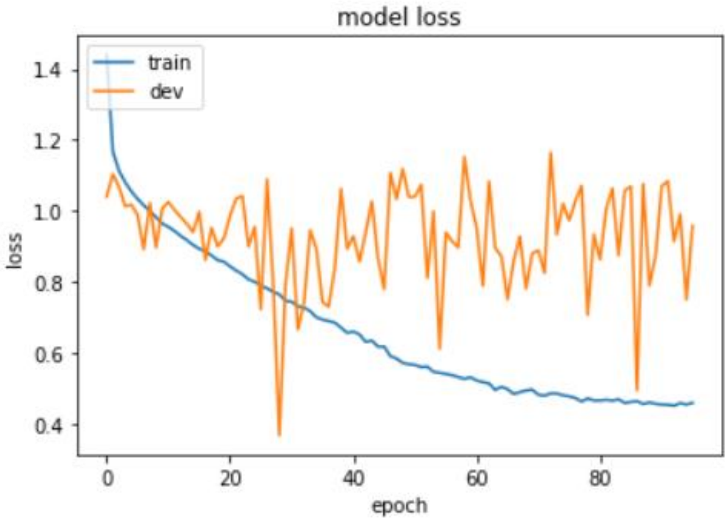


Figure 28: train loss vs validation loss VGG-16

H. VGG-16 with class weights

Keras class weights were used to emphasize the minority classes. VGG-16 with class weights resulted in an accuracy of 70.9%.

VGG16 and VGG16 with class weights were excluded from the ensemble as their predictions were not improving the test accuracy.

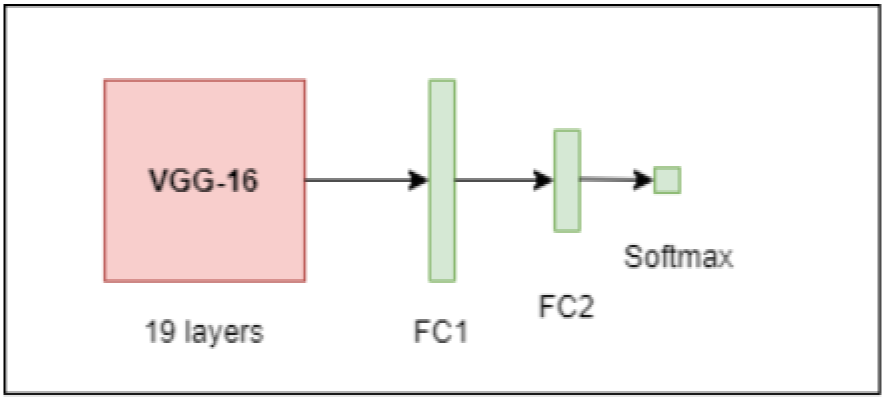


Figure 29: VGG-16 architecture

I. FaceNet

FaceNet is a facial recognition system proposed by Google Researchers in 2015 and was used on many face recognition datasets [5]. It uses deep learning architecture such as ZF-net and Inception. In this method, we tested fine-tuning on a pre-trained model using FER2013 dataset. We used Keras's implementation of FaceNet that consists of426 layers. The input dimensions for FaceNet are 160x160. In the first iteration of training, we resized FER2013 images to 160x160, froze the entire network except for BN layers, added two fully connected layers with 4096 and 1024neurons, respectively. Each of these two layers is followed by a dropout of 30%. This dropout was added before the first fully connected layer as well. We usedSGD optimizer with a learning rate of 0.01, the momentum of 0.9, the decay of0.0001, and nesterov that was set to true. We used Keras callbackReduceLROnPlateau to reduce the learning rate by a factor equals to 0.5 when the validation accuracy does not improve for 10 epochs. The model was trained for 150 epochs with a batch size of 128 and early stopping enabled. We recorded a test accuracy of 70.2%. In the second iteration of training, we unfroze the entire network, used a small learning rate of 0.0001 with SGD optimizer, and fine-tuned the model. This resulted in a test accuracy of 70.8%.

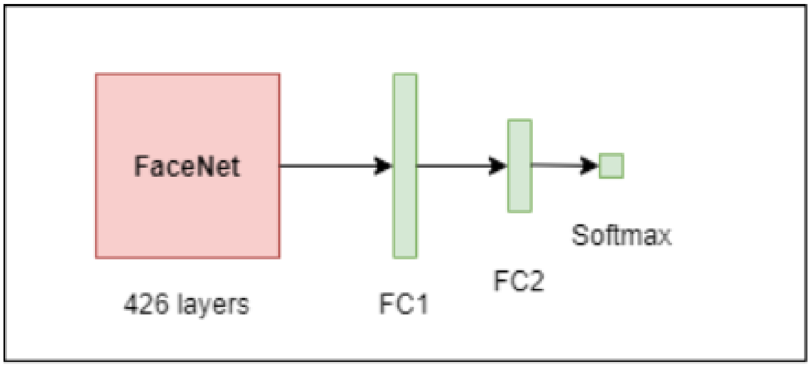


Figure 30: FaceNet architecture

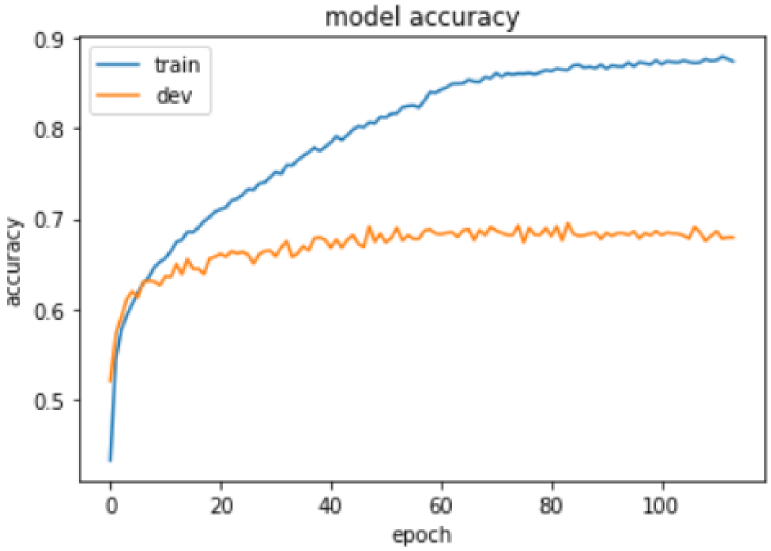


Figure 31: train accuracy vs validation accuracy - FaceNet

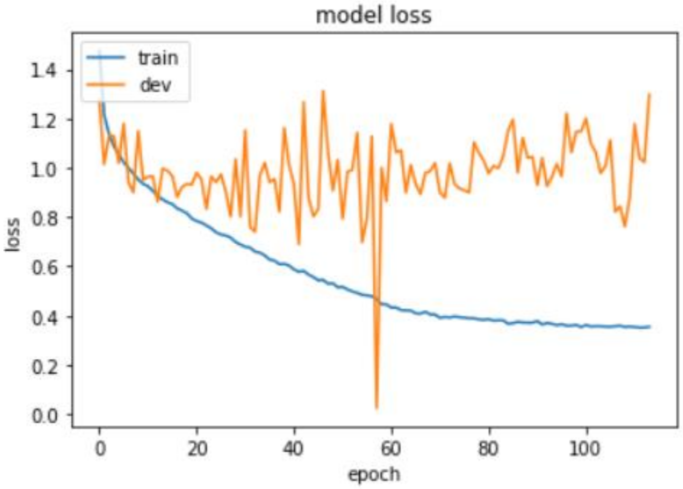


Figure 32: train loss vs validation loss – FaceNet

VIII. Results

Our ensemble which resulted in test accuracy of 76.01% is composed of seven models (5-layer CNN, Resnet- 50, Resnet-50+ class weights, Resnet-50 using Histogram equalization, Senet-50 without class weights, Senet-50 + class weights, and FaceNet) as shown in Table 2. The prediction from VGG16 and VGG16 models of class weights did not improve the test accuracy. In an ensemble model, it is beneficial to have different models performing well in different classes. Though models of class weights resulted in lower individual accuracy, they contributed to a better ensemble model.

Our method of using 5-layer CNN has an accuracy of 69.6%. This is a relatively simple model that takes less time to train. Applying preprocessing techniques such as Data Augmentation and Image Sharpening resulted in improved performance comparing with most of the previous CNN models [30][10][11]. Our method using Resnet-50, Senet- 50 and VGG-16 models had accuracies of 73.8%, 73.3%, and 71.2% respectively. This outperformed previous studies' results that have employed pre-trained models [28][25] (see Table 2 that summarized previous results). Our ensemble as shown in Table 2 composed a unique blend of models, some of which were trained on sharpened images and histogram equalized images led to an accuracy of 76.01%.

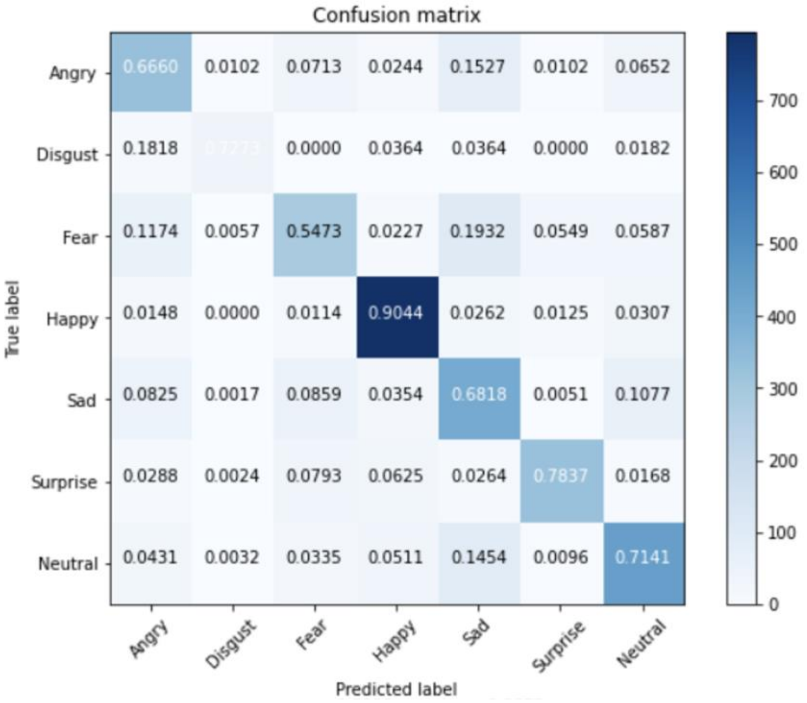


Figure 33: Confusion matrx – Resnet-50

Figures 21 and 22 displays the training and validation accuracy of Resnet-50. Most pre-trained networks displayed a similar training and validation accuracy. After the 15th epoch, the validation accuracy stagnates while the training accuracy keeps increasing. Even the best individual model such as Resnet-50 performed poorly on emotions such as Anger, Fear, and Sadness (see confusion matrix in Figure 33). Some facial expressions can have multiple interpretations [28] such as Fear and Sadness or Fear and Anger which make it even harder for models to identify differentiating patterns. We used a small learning rate with early stopping to avoid over-fitting when we unfroze the entire network.

Table 2: Ensemble model and summary of test accuracies (no auxiliary data was used). The ensemble is composed of seven models (5-layer CNN , Resnet-50, Resnet-50+ class weights, Resnet-50 using Histogram equalization, Senet-50 without class weights, Senet-50 +

Model	**Preprocessing**	**Test Accuracy (%)**	**Used in Ensemble Model?**
5-layer CNN	Data Augmentation (DA) + Unsharp Mask	69.6	✓
Resnet-50	DA	**73.8**	✓
Resnet-50 + class weights	DA	73.4	✓
Resnet-50	DA + Histogram equalization	72.4	✓
Resnet-50	DA + Unsharp Mask	73.3	
Senet-50	DA	73.3	✓
Senet-50 + class weights	DA	72.4	✓
VGG-16	DA	71.2	
VGG-16 + class weights	DA	70.9	
FaceNet	DA	70.8	✓
Ensemble	-	**76.01**	

Table 3: Test accuracies of the previous studies (No auxiliary data was used).

Previous Studies	**Model**	**Test Accuracy (%)**
Khanzada et al.	Resnet-50	73.2
	Senet-50	70.0
	VGG-16	69.5
	Ensemble	**74.8**
Pramerdorfer et al.	Resnet-50	72.4
	Inception	71.6
	VGG-16	72.7
	Ensemble	**75.2**

IX. CONCLUSION AND FUTURE WORK

In this study, we were able to demonstrate that the use of sharpening technique to preprocess data for a CNN model boosted performance even though the CNN model is relatively simple. This is improved performance vis a vis previously published methods such as Wang et al. [29] and Shin et al. [11]. Our image preprocessing technique led to emphasizing the prominent edges in facial images resulting in higher accuracy. Our baseline model has resulted in an accuracy of 69.46%. We were able to achieve a higher test accuracy for each of the pre-trained models compared to the previous FER studies [28][25] using the FER2013, no auxiliary data and without freezing the entire network.

Future work can include performing data cleaning such as eliminating noisy images from the FER2013 and employing auxiliary data to boost the model performance. Better detection of human emotions can help children with autism, blind people to read facial expressions, robots to better interact with humans, and ensure driver safety by monitoring attention while driving. FER can also enhance the emotional intelligence of applications and improve customer experience by using emotion recognition.

REFERENCES

[1] A. Gupta, and M. Gupta. Transfer Learning for Small and Different Datasets: Fine-Tuning A Pre-Trained Model Affects Performance. Journal of Emerging Investigators. 2020

[2] A. Sajjanhar, Z. Wu and Q. Wen, ”Deep Learning Models for Facial Expression Recognition,” 2018 Digital Image Computing: Techniques and Applications (DICTA), Canberra, ACT, Australia, 2018, pp. 1-6, doi: 10.1109/DICTA.2018.8615843.

[3] Behnam Kabirian Dehkordi, javad Haddadnia,”Facial Expression Recognition in Video Sequence Images by Using Optical Flow”, IEEE Proceedings of 2nd International conference on Signal Processing Systems (ICSPS),

[4] B. Houshmand and N. Mefraz Khan, ”Facial Expression Recognition Under Partial Occlusion from Virtual Reality Headsets based on Transfer Learning,” 2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM), New Delhi, India, 2020, pp. 70-75, doi: 10.1109/BigMM50055.2020.00020.

[5] F. Schroff, D. Kalenichenko and J. Philbin, ”FaceNet: A unified embedding for face recognition and clustering,” 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 815-823, doi: 10.1109/CVPR.2015.7298682.

[6] C. Garbin, X. Zhu and O. Marques, Dropout vs. batch normalization: an empirical study of their impact to deep learning. Multimed Tools Appl 79,

12777–12815 (2020). https://doi.org/10.1007/s11042-019-08453-9

[7] J. Kaur, and A. Sharma, “Review Paper on Edge Detection Techniques in Digital lmage Processing” lnternati0nal Journal of lnnovations and Advancement in Computer Science ljiacs lssn 2347 – 86l6, Volume 5, lssue ll, November 2016.

[8] J. Luttrell, Z. Zhou, C. Zhang, P. Gong and Y. Zhang, ”Facial Recognition via Transfer Learning: Fine- Tuning Keras vggface,” 2017 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2017, pp. 576-579, doi: 10.1109/CSCI.2017.98.

[9] M. Abdelmaksoud, E. Nabil, I. Farag and H. A. Hameed, ”A Novel Neural Network Method for Face Recognition With a Single Sample Per Person,” in IEEE Access, vol. 8, pp. 102212-102221, 2020, doi:10.1109/ACCESS.2020.2999030.

[10] K. Liu, M. Zhang, and Z. Pan. 2016. “Facial Expression Recognition with CNN Ensemble”. In International Conference on Cyberworlds. 163–166.

[11] M. Shin, M. Kim and D. Kwon, ”Baseline CNN structure analysis for facial expression recognition,” 2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), New York, NY, 2016, pp. 724-729, doi: 10.1109/ROMAN. 2016.7745199.

[12] Y. Tang, “Deep Learning using Support Vector Machines,” in International Conference on Machine Learning (ICML) Workshops, 2013.

[13] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner,W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, Y. Zhou, C. Ramaiah, F. Feng, R. Li, X. Wang, D. Athanasakis, J. Shawe-Taylor, M. Milakov, J. Park, R. Ionescu, M. Popescu, C. Grozea, J. Bergstra, J. Xie, L. Romaszko, B. Xu, Z. Chuang, and Y. Bengio. Challenges in representation learning: A report on three machine learning contests. Neural Networks, 64:59–63, 2015. Special Issue on ”Deep Learning of Representations”

[14] P. Ekman and E.L. Rosenberg, ”What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System”. USA: Oxford University Press, 1997.

[15] S L Happy, Anjith George, Aurobinda Routray, ”A Real Time Facial Expression Classification System Using local Binary patterns”, IEEE Proceedings of 4th International Conference on IHCI, Kharagpur, India, December 27-29,2012.

[16] T. Ojala, M Pietikainen, and T. maenpaa, ”multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” IEEE transactions on pattern Analysis and machine Intelligence, vol. 24, no. 7, pp 971-987, 2002.

[17] G. B. Huang, H. Lee, and E. Learned-Miller. Learning hierarchical representations for face verification with convolutional deep belief networks. In Proc. CVPR, 2012. 4

[18] I. Oztel, G. Yolcu and C. Oz, ”Performance Comparison of Transfer Learning and Training from Scratch Approaches for Deep Facial Expression Recognition,” 2019 4th International Conference on Computer Science and Engineering (UBMK), Samsun, Turkey, 2019, pp. 1-6, doi: 10.1109/UBMK.2019.8907203.

[19] K. Liu, M. Zhang, and Z. Pan. 2016. “Facial Expression Recognition with CNN Ensemble”. In International Conference on Cyberworlds. 163–166.

[20] K. Hasan et al., ”Facial Expression Based Imagination Index and a Transfer Learning Approach to Detect Deception,” 2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII), Cambridge, UK, 2019, pp. 634-640, doi: 10.1109/ACII.2019.8925473.

[21] M. Shin, M. Kim and D. Kwon, ”Baseline CNN structure analysis for facial expression recognition,” 2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), New York, NY, 2016, pp. 724-729, doi: 10.1109/ROMAN. 2016.7745199.

[22] M. Xu, W. Cheng, Q. Zhao, L. Ma and F. Xu, ”Facial expression recognition based on transfer learning from deep convolutional networks,” 2015 11th International Conference on Natural Computation (ICNC), Zhangjiajie, China, 2015, pp. 702-708, doi: 10.1109/ICNC.2015.7378076.

[23] Ngo, Quan T, and Seokhoon Yoon. “Facial Expression Recognition Based onWeighted-Cluster Loss and Deep Transfer Learning Using a Highly Imbalanced Dataset.” Sensors (Basel, Switzerland) vol. 20,9 2639. 5 May.

2020, doi:10.3390/s20092639

[24] N. Darapaneni, R. Choubey, P. Salvi, A. Pathak, S. Suryavanshi and A. R. Paduri, ”Facial Expression Recognition and Recommendations Using Deep Neural Network with Transfer Learning,” 2020 11th IEEE Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), New York, NY, USA, 2020, pp. 0668-0673, doi: 10.1109/UEMCON51285.2020.9298082.

[25] Pramerdorfer, C., Kampel, M.: Facial expression recognition using convolutional neural networks: state of the art. Preprint arXiv:1612.02903v1, 2016.

[26] S. Zhou, Y. Liang, J. Wan. 2016. Facial Expression Recognition Based on Multi-scale CNNs. In Biometric Recognition. Springer International Publishing, 128–135.

[27] S. Wang and Z. Li, ”A new transfer learning Boosting application on facial expression recognition,” 2014 International Joint Conference on Neural Networks (IJCNN), Beijing, China, 2014, pp. 432-439, doi: 10.1109/IJCNN.2014.6889504.

[28] Amil Khanzada and Charles Bai and Ferhat Turker Celepcikay, 2020 ”Facial Expression Recognition with Deep Learning” arXiv.

[29] X. Wang, J. Huang, J. Zhu, M. Yang, and F. Yang. “Facial expression recognition with deep learning”. In: Proceedings of the 10th International Conference on Internet Multimedia Computing and Service. New York, NY,

USA: ACM, 2018. (ICIMCS ‘18), p: 10:1 10:4. Doi: (http://doi.acn.org/10.1145/3240876.3240908).

[30] X. Wang, J. Huang, J. Zhu, M. Yang, and F. Yang. “Facial expression recognition with deep learning”. In: Proceedings of the 10th International Conference on Internet Multimedia Computing and Service. New York, NY, USA: ACM, 2018. (ICIMCS ‘18), p:

[31] Y. Tang. 2013. Deep Learning using Linear Support Vector Machines. Computer Science (2013).

[32] Y. Liang, S. Liao, L. Wang, and B. Zou, ”Exploring regularized feature selection for person specific face verification,” 2011 International Conference on Computer Vision, Barcelona, 2011, pp. 1676-1683, doi: 10.1109/ICCV.2011.6126430.

[33] Z. Zhang, P. Luo, C.-C. Loy, and X. Tang, “Learning Social Relation Traits from Face Images,” in Proc. IEEE Int. Conference on Computer Vision (ICCV), 2015, pp. 3631–3639.

[34] Z. Xie, Y. Li, X. Wang, W. Cai, J. Rao, and Z. Liu, ”Convolutional Neural Networks for Facial Expression Recognition with Few Training Samples,” 2018 37th Chinese Control Conference (CCC), Wuhan, 2018, pp. 9540-9544, doi: 10.23919/ChiCC.2018.8483159.

[35] A. Lonare, and S. V. Jain. “A Survey on Facial Expression Analysis for Emotion Recognition”. International Journal of Advanced Research in Computer and Communication Engineering 2.12

[36] M. Pantic, M. Valstar, R. Rademaker and L. Maat, "Web-based database for facial expression analysis", IEEE International Conference on Multimedia and Expo (ICME), pp. 1-5, 2005.

[37] M. Kamachi, M. Lyons, and J. Gyoba, The japanese female facial expression (jaffe) database, 1998.

[38] C. Shan, S. Gong, and P. Meowan, "Facial expression recognition based on local binary patterns: A comprehensive study", Image & Vision Computing, vol. 27, no. 6, pp. 803-816, 2009.

[39] S. Rani, V. Tejaswi, B. Rohitha, and B. Akhil,"Pre filtering techniques for face recognition based on edge detection algorithm. J. Eng. Technol. 13–218 (2017)

[40] M. Abo-Zahhad, R. Gharieb, S. Ahmed, and A. Donko.. Edge Detection with a Preprocessing Approach. Journal of Signal and Information Processing. (2014) 5. 123-134. 10.4236/jsip.2014.54015.

[41] J. Prasad, and G. P. Chourasiya, and N.S. Chauhan, "Face detection using color based segmentation and edge detection," International Journal of Computer Applications (0975-8887), voL72, no.16, pp.49-54, June 2013.

[42] M. Ali, and D. Clausi, "Using the Canny edge detector for feature extraction and enhancement of remote sensing images," IGARSS 2001. Scanning the Present and Resolving the Future. Proceedings. IEEE 2001 International Geoscience and Remote Sensing Symposium (Cat. No.01CH37217), Sydney, NSW, Australia, 2001, pp. 2298-2300 vol.5, doi:

10.1109/IGARSS.2001.977981.

[43] Z. Yu, and C. Zhang.”Image based Static Facial Expression Recognition with Multiple Deep Network Learning”. In Proceedings of the 2015 ACM on International Conference on Multimodal Interaction (ICMI '15). Association for Computing Machinery, New York, NY, USA, 435–442.

[44] L. Nanni, A. Lumini, and S. Brahnam.. “Ensemble of texture descriptors for face recognition obtained by varying feature transforms and preprocessing approaches”. Applied Soft Computing. 61. 10.1016/j.asoc.2017.07.057. (2017)

[45] Vepuri, K., Attar, N. (2021), 'Improving the Performance of Deep Learning in Facial Emotion Recognition with Image Sharpening', World Academy of Science, Engineering and Technology, Open Science Index 172, International Journal of Computer, and Information Engineering, 15(4), 234 – 237.

[46] Raut, Nitisha, "Facial Emotion Recognition Using Machine Learning" (2018). Master's Projects. 632. DOI: https://doi.org/10.31979/etd.w5fs-s8wd

[47] Patidar, P. (2020, November 20). Image data Augmentation- image processing IN TensorFlow- Part 2. Retrieved April 28, 2021, from https://medium.com/mlait/image-data-augmentation-image-processing-in-tensorflow-part-2-b77237256df0

[48] Saha, S. (2018, December 17). A comprehensive guide to convolutional neural networks - the eli5 way. Retrieved April 28, 2021, from https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-

neural-networks-the-eli5-way-3bd2b1164a53

[49] Thakur, R. (2020, November 24). Step by step VGG16 implementation in Keras for beginners. Medium. https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c#:%7E:text=VGG16%20is%20a%20convolution%20neural,competition%20in%202014.&text=It%20follows%20this%20arrangement%20of,by%20a%20softmax%20for%20output.

[50] Dwivedi, P. (2019, March 27). Understanding and Coding a ResNet in Keras - Towards Data Science. Medium. https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33#:%7E:text=The%20ResNet%2D50%20model%20consists,over%2023%20million%20trainable%20parameters.&text=Our%20ResNet%2D50%20gets%20to,in%2025%20epochs%20of%20training.

[51] Tsang, S. (2019, April 22). Review: SqueezeNet (Image Classification) - Towards Data Science. Medium. https://towardsdatascience.com/review-squeezenet-image-classification-e7414825581a

[52] Kumar, D. (2020, June 21). Introduction to FaceNet: A Unified Embedding for Face Recognition and Clustering. Medium. https://medium.com/analytics-vidhya/introduction-to-facenet-a-unified-embedding-for-face-recognition-and-clustering-dbdac8e6f02