Fall 2022

A Study on Human Face Expressions using Convolutional Neural Networks and Generative Adversarial Networks

Sriramm Muthyala Sudhakar
San Jose State University

A Study on Human Face Expressions using Convolutional Neural Networks and Generative Adversarial Networks

A Project

Presented to

The Faculty of the Department of Computer Science

San Jose State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By Sriramm Muthyala Sudhakar

December 2022

The Designated Project Committee is pending approval on the Project Titled

"A Study on Human Face Expressions using Convolutional Neural Networks and Generative Adversarial Networks"

By

Sriramm Muthyala Sudhakar

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

December 2022

Dr. Nada Attar, Department of Computer Science

Dr. Mike Wu, Department of Computer Science

Dr. Noha Elfiky, Associate Professor of Business and Data Analytics, Saint Mary's College of California

ABSTRACT

Human beings express themselves via words, signs, gestures, and facial emotions. Previous research using pre-trained convolutional models had been done by freezing the entire network and running the models without the use of any image processing techniques. In this research, we attempt to enhance the accuracy of many deep CNN architectures like ResNet and Senet, using a variety of different image processing techniques like Image Data Generator, Histogram Equalization, and UnSharpMask. We used FER 2013, which is a dataset containing multiple classes of images. While working on these models, we decided to take things to the next level, and we attempted to make changes to the models themselves to improve their accuracy.

While working on this research, we were introduced to another concept in Deep Learning known as Generative Adversarial Networks, which are also known as GANs. They are generative deep learning models which are based on deep CNN models, and they comprise two CNN models - a Generator and a Discriminator. The primary task of the former is to generate random noises in the form of images and passes them to the latter. The Discriminator compares the noise with the input image and accepts/rejects it, based on the similarity. Over the years, there have been various distinguished architectures of GANs namely CycleGAN, StyleGAN, etc. which have allowed us to create sophisticated architectures to not only generate the same image as the original input but also to make changes to them and generate different images. For example, CycleGAN allows us to change the season of scenery from Summer to Winter or change the emotion in the face of a person from happy to sad. Though these sophisticated models are good, we are working with an architecture that has two deep neural networks, which essentially creates problems with hyperparameter tuning and overfitting.

Keywords - Facial Expression Recognition, Deep Learning, ResNet, Senet, GANs

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. INTRODUCTION

Facial Expression Recognition has been the focus of study over the past two decades and research has improved face detection, face recognition, and face tracking techniques. It is a very popular concept, used in a variety of fields like HCI, security, analysis, and so on [30] [31]. However, it is still a challenge to retrieve the particularly important features from the datasets for analysis. The primary challenge is that the machine needs to understand the problem statement here and perform the required analysis, by taking in the essential features while ignoring the unnecessary features. To classify the various emotions in the dataset, the computer needs to understand the different emotions by learning from the dataset, thereby segregating the data based on the features (expressions). To perform this kind of analysis, a particularly large dataset, consisting of an ample amount of data for each expression is required. Significant progress has been made in these algorithms by segregating the various facial expressions as different classes [33] [34] [35]. CNN can retrieve the most important characteristics from these classes from classification. They are a branch of Artificial Neural Networks, for working on visual imagery, and are specifically used for FER [21] [22] as they can retrieve important features from the image data. Though the task at hand seems quite simple, the implementation of a CNN model has a lot of challenges. To begin with the dataset, we should have a balanced dataset [23], with each class containing an equal amount of images such that one of the classes would not overfit. Problems arise with the CNN model itself, as they are Deep Learning models, containing several types of layers and many more hyperparameters to adjust to get the desired results. Another problem arises during the training of the model. Usually, new Deep CNN models have to be initially trained on big raw datasets (which typically takes a long time) for the weights of the neural network to get adjusted. Hence, it is not advisable to train a new deep CNN model from scratch.

To deal with this situation, we have a concept known as Transfer Learning [26], which uses a pre-trained deep learning model to resolve the problem. Research has shown that transfer learning has a higher success rate and shorter training time than CNN models that are trained from scratch. Using a frozen pre-trained model can save a lot of time as the network does not have to learn all the general features from the data. However, as the enhancement of the model occurs only in a part of the feature space, this process can negatively impact the accuracy. But, relaxing the entire network can boost the accuracy, but it can also lead to overfitting and in some cases, can decrease the accuracy. We can avoid the overfitting problem by freezing the entire pre-trained network and then attaching a couple of layers to the model and training the model for a few epochs. For the next set of training, we unfreeze the pre-trained model and train it at a low learning rate. This strategy helps us fine-tune the parameters without training the network. The concept of transfer learning has been successfully used on the FER datasets and they have outperformed the conventional CNN models [27].

We also explore Generative Adversarial Networks, which, simply explained are a couple of CNN models – A Generator and a Discriminator bonded together, where the former tries to deceive the latter, to create an image that resembles the original input image. Apart from the basic GAN architecture, there are many advanced GAN models such as StyleGAN, and CycleGAN, which perform advanced image generations like improving the characteristics of the input data, changing the landscape, etc. Initially, we create a simple GAN architecture for running on FER 2013 dataset, and we attempt to use a Deep CNN model like Resnet 50 in the place of the discriminator. Later, we also try to explore a complicated GAN architecture such as CCycleGAN where we try to use the Resnet-50 as the discriminator and monitor the performance of the model

II. PROJECT ROADMAP

This project could be split into two parts. The first part of the project experiments with improving the accuracy of some of the most popular neural network architectures such as ResNet – 50 and Senet-50. This section could further be divided into two parts, where we experiment with some of the popular preprocessing techniques such as Data Augmentation, Histogram Equalization, and UnSharp mask. Data Augmentation is useful when we have a smaller dataset, like in our case, as the preprocessing technique yields additional images to work on. Using Histogram Equalization helps to improve the contrast of the images, and thereby could contribute positively to the models' performance. UnSharp Mask is used to sharpen the input images. This preprocessing technique is useful in our case as the images in FER 2013 dataset are a little blurred. We would experiment with models and will record their performance with and without these preprocessing techniques.

The second part of the project focuses on Generative Adversarial Networks (GANs) which is a fairly new concept in the Deep Learning world. We attempt to build a basic GAN architecture from scratch to run on FER 2013 dataset, after which we would substitute the existing discriminator architecture with a ResNet model and record their effectiveness. We would try to employ a weight-normalized ResNet model trained on datasets like imagenet or FER 2013 itself. Later, we use a CCycleGAN architecture which generates new types of images from the original image like changing a horse into a zebra as the Generator and Resnet-50 as the discriminator.

III. RELATED WORK

Several previous researchers have performed research and developed various models for facial expression recognition, and these studies have shown steady progress. Sarwesh et al [16] used CNN and OpenCV to perform live facial emotion recognition. Classifications on CNN models have shown steady improvements when using diverse image processing techniques. Rani et al. [11] have made use of techniques to detect edges and pre-filter the original data for Facial Emotion Recognition. Some of the other researchers have used other types of detectors such as Gaussian [18], Colored [19] for multi-view face detection, and Canny [20] for feature extraction. Histogram Equalization, which is used to improve the contrast of the images was used by Yu et al. for preprocessing [40]. Apart from preprocessing of the data, there are only a few datasets available for FER and they consist of a small number of images, which might overfit the model. To deal with this problem, researchers started implementing transfer learning, where the model is trained on a large raw dataset for a large number of epochs, which is then used on a smaller dataset like FER 2013. The application of transfer learning to these architectures has improved their overall performance and produced greater results when compared to training them from scratch on smaller datasets [24]. CNN models tend to perform well on their own, so the introduction of preprocessing techniques contributes to the improvement in the model accuracies by a huge margin when compared to raw images [40]. Some of the earlier CNN models did not make use of data augmentation and preprocessing techniques and they did not work well with rotated and deviated images [41] [42] [43]. Tang [44] created a CNN model using a linear Support Vector Machine (SVM).

Several datasets are available for facial recognition such as RAF-DB [49], RaFD [50], CK+, JAFFE [32], FER 2013 [48], and many more. FER 2013 model had an accuracy of 65%

when it was trained on CNN models. The highest accuracy that was achieved in a Kaggle competition using FER 2013 was 71.2% using CNN with SVM.

We attempt to make use of Image Data Augmentation to produce extra images. Image Sharpening technique has helped to enhance the images and has helped to enhance the CNN model. We attempted these two techniques on two CNNs, Resnet- 50 and Senet-50. We did not make use of any other auxiliary data while using FER 2013 on these models.

Generative Adversarial Networks [13] make use of two CNN models to generate images that are identical to the input original image. Ever since Goodfellow et. al [44] created this architecture, several kinds of research have been put forth to create different kinds of images from the original image. For example, CycleGANs are used to change certain styles in the original images, like changing the scenery and changing the facial expression. Tesei [45] made use of CycleGANs to create images with different facial expressions, such as generating an image where a happy person has a sad expression. Currently, there are several studies on making use of Deep CNN models as a part of the GAN architecture. Prabhat et. al [47] compared DCGANs [12] with Conditional GAN using Handwritten digits. Hyejeong et. al [46] used GANs for Network Intrusion Detection.

IV. DATASET DESCRIPTION

FER-2013 is a popular dataset consisting of faces of people that were generated using Image Search API that was created in the year 2001 by Google. The dataset was generated by Goodfellow et al. [51] and was introduced in the year 2013 for a competition on the Kaggle website. The dataset contains over 35,000 images, of which 28,709 belong to the training set, and 3589 belong to the validation and test set respectively. The primary purpose for which the dataset was introduced is to classify them based on their emotions. The dataset can be classified into 7 labels, each of which is an emotion. They are “Anger”, “Disgust”, “Fear”, “Happy”, “Sad”, “Surprise”, and “Neutral”, which are labeled from 0 to 6 respectively. However, the images that are present in this dataset are unevenly distributed. The distribution of the dataset is represented in Table 1.

Table 1: Distribution of different classes in the FER 2013 dataset

Emotion	**Image Count**	**Class**
Angry	4593	0
Disgust	547	1
Fear	5121	2
Happy	8989	3
Sad	6077	4
Surprise	4002	5
Neutral	6198	6

From the table above, we can see that the dataset is not a balanced dataset and the Label 1 has just over 500 images in total over the rest of the labels. Each image in the dataset has a size of 48x48. To use this dataset for processing on Resnet 50 and Senet 50, we have to resize the images to at least 197x197, or else the model breaks.

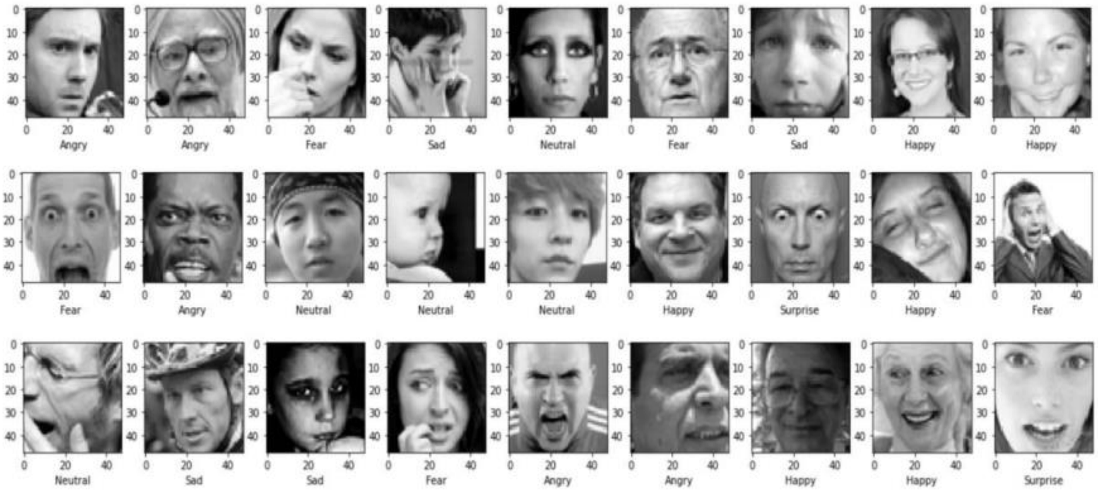


Figure 1: Images from FER - 2013

From the above image, we can see that many images are blurred, some have a bright background, etc. We have adapted some of the popular image processing techniques to deal with these situations, such as Histogram equalization, and UnSharp Mask.

V. IMAGE PREPROCESSING

The presence of a good dataset is important for a CNN model to perform well. Multiple factors could be wrong with a dataset such as the presence of blurry images, the face of the person might be dark due to a brighter background, etc. Image pre-processing techniques perform some image formatting to make the image ready to be passed into the Neural networks. Some of these preprocessing techniques include operations like rotating the image, compressing them, increasing the brightness and contrast, increasing the sharpness of the image, normalization, dimensionality reduction, etc. Image processing techniques provide a form of uniformity to the images. All the images that need to be passed to the network should be of the same size. Some techniques augment the image and generate additional images, thus increasing the size of the dataset.

Some commonly used preprocessing techniques are as follows.

1) Image Data Augmentation
2) Histogram Equalization
3) UnSharp Mask

A. Image Data Augmentation-

Image Data Augmentation is a Keras library that expands the given input. This library is particularly useful for smaller datasets as the augmentation technique applies various transformations on the images such as Rotations, Shifts, and Flips, Adjusting the brightness and zooming the images [3], thus providing at least 5 extra images for each image in the dataset. The resultant images from the library are relatively similar (apart from the transformations that are applied) but the neural networks would treat them as different images, thus not making any biased judgments. Image Data Generator allows the user to import data from a Data frame or a directory, given that the directory contains subdirectories that act as the classes in the dataset.

Image Data Generator performs random rotations between 0 and 360 which is provided by the user.

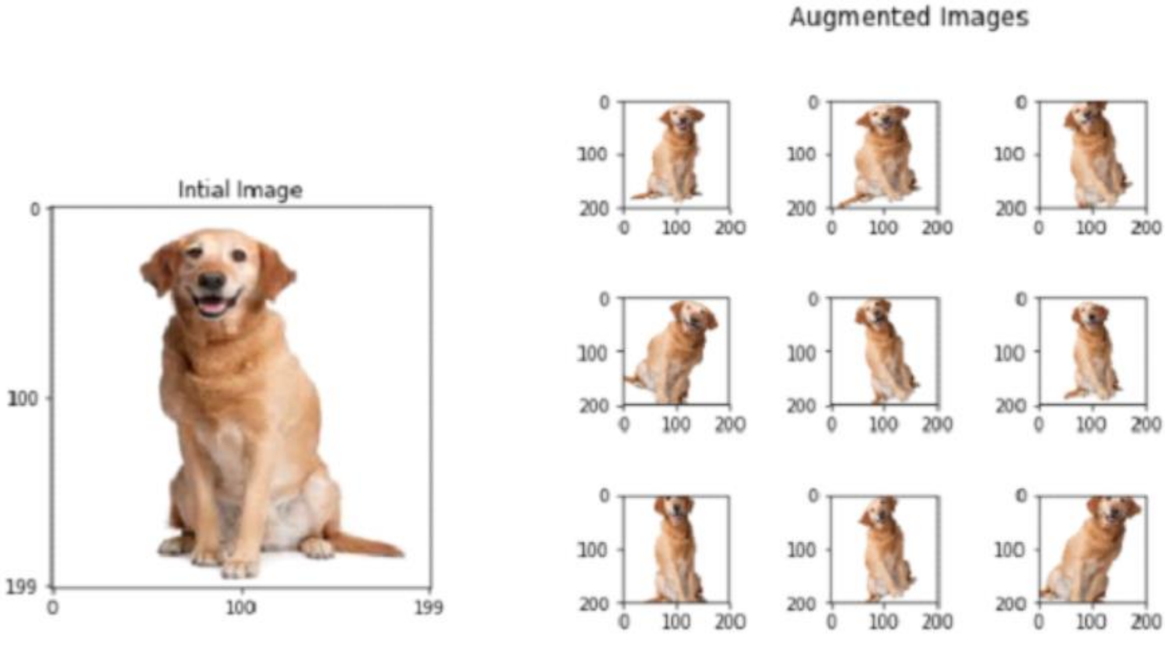


Figure 2: Data augmentation output (Patidar, P. (2020, November 20). Image data Augmentation- image processing IN TensorFlow- Part 2. Retrieved September 28, 2022, from Medium. https://medium.com/mlait/image-data-augmentation-imageprocessing-in-tensorflow-part-2-b77237256df0)

B. Histogram Equalization -

Histogram Equalization is a preprocessing method to improve the image intensities by increasing the contrast of the images. It helps to improve the quality of images by differentiating the foreground from the background when they are in the same color [28]. The technique works particularly well for grayscale images, and it has been proven to be efficient for improving the contrasts of X-ray and MRI images. Every image can be represented in the form of pixels. The grayscale images have a single intensity value ranging from 0 to 255, while the color images have three intensity values within the same range of 0 to 255, each denoting red, green and blue.

In general, a histogram denotes the distribution of the frequency [5] in the input data. For example, if we have a 10x30 image, we have 300 pixels in it. Each pixel can be represented in the range of 0 to 255 based on the intensity of the grayscale.

Using this data, we can essentially generate a histogram containing 256 bins, where each bin is calculated as

$$p_n = \frac{Number\ of\ pixels\ with\ the\ intensity\ n}{Total\ number\ of\ pixels}$$

Figure 3: Histogram Formula

Consider the following grayscale image.

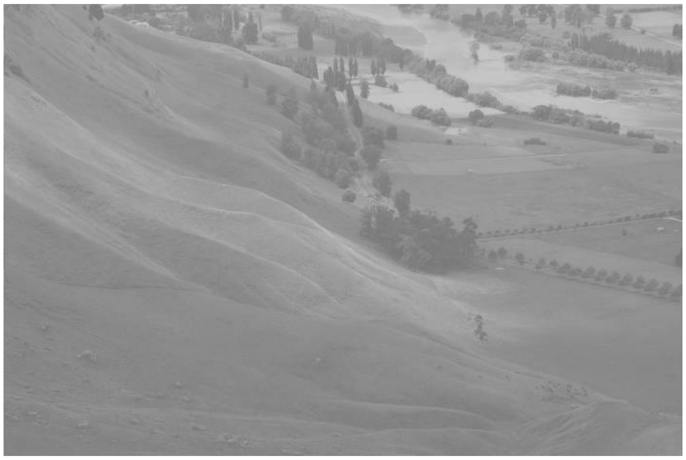


Figure 4: Faded grayscale image (Sudhakar, S. (2017, July 9). Histogram Equalization. Retrieved September 28, 2022, from Medium. https://towardsdatascience.com/histogram-equalization-5d1013626e64)

Converting the above grayscale image into a histogram yields the following:

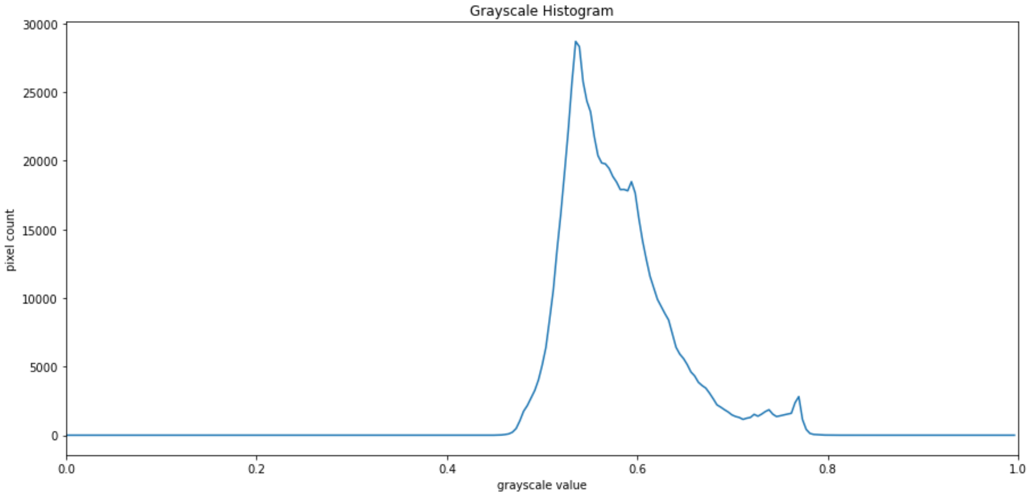


Figure 5: Histogram of the Faded image

As we can see, the pixels peaked between the ranges 0.5 and 0.7. The higher values of the pixels indicate that the image is faded and has higher contrast. Similarly, the following darker image yields a histogram having lower values.

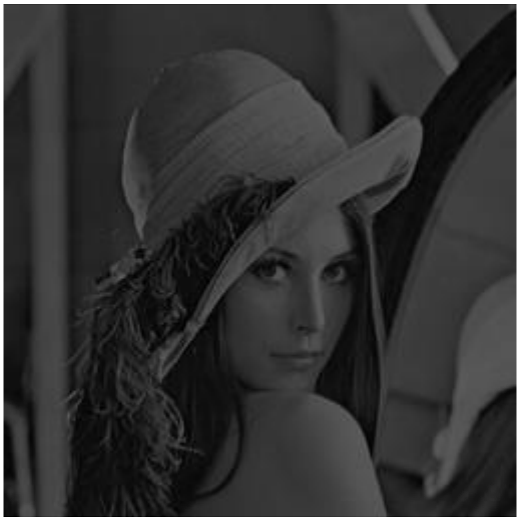


Figure 6: Dark Image (Samuel, S. (2021, January 10). Introduction to Histogram Equalization for Digital Image Enhancement. Retrieved September 28, 2022, from Medium. https://levelup.gitconnected.com/introduction-to-histogram-equalization-for-digital-image-enhancement-420696db9e43)

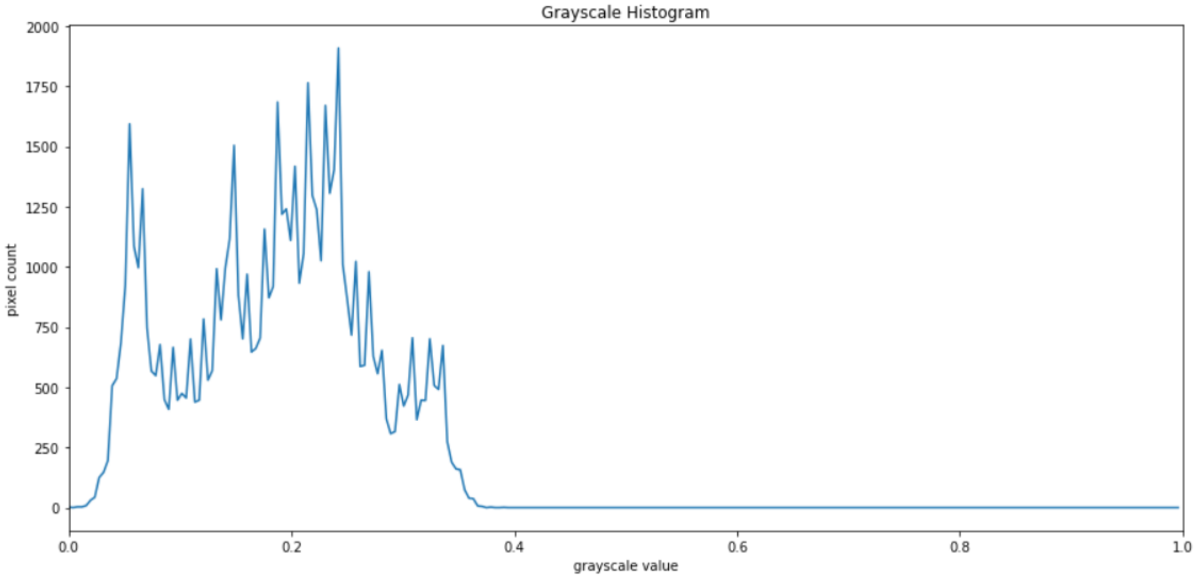


Figure 7: Histogram of the Dark Image

Hence, to get a balanced image, it is necessary to perform histogram equalization to stretch the pixel intensities. By performing the same, the contrast of the image will be enhanced by brightening the brighter pixels and vice versa. Here is a comparison of an image from the FER 2013 dataset before and after we perform the Histogram Equalization.

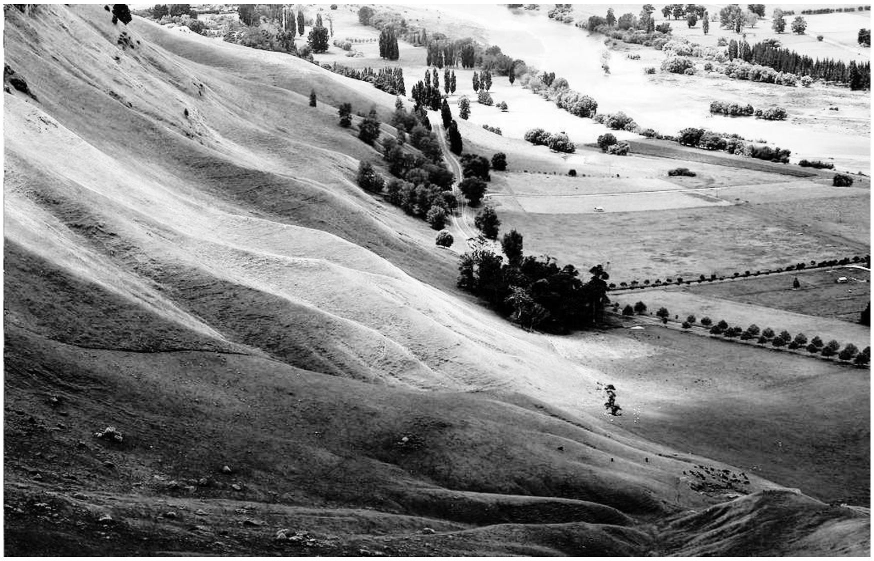


Figure 8: *Histogram Equalized Image of Figure 1(Sudhakar, S. (2017, July 9). Histogram Equalization. Retrieved September 28, 2022, from Medium. https://towardsdatascience.com/histogram-equalization-5d1013626e64)*

As we can see from the above image, after performing histogram equalization, the original image is much better.

C. UnSharp Mask-

UnSharp Mask is an image preprocessing technique that sharpens the texture of the input image, thereby improving its quality. This technique works by identifying the Laplacian of the image, which is a derivative operator to identify the edges in an image. Typically, edges have larger Laplacian values which can be positive or negative, while smoother areas would have lower Laplacian values. Calculation of Laplacian for an image would return an image where it is gray in the smoother places without edges, and black and white in the areas containing edges. By removing the Laplacian from an image, the image would end up with crisper edges, thereby returning a sharper image than the original image.

Consider the following set of images from the FER 2013 dataset.

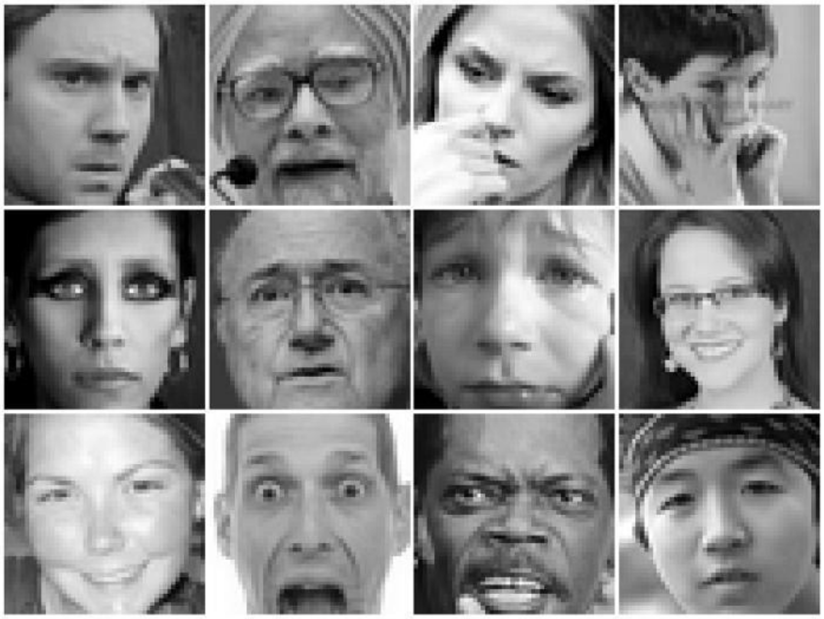


Figure 9: *A blend of different expressions from the FER 2013 dataset*

Upon sharpening the above image,

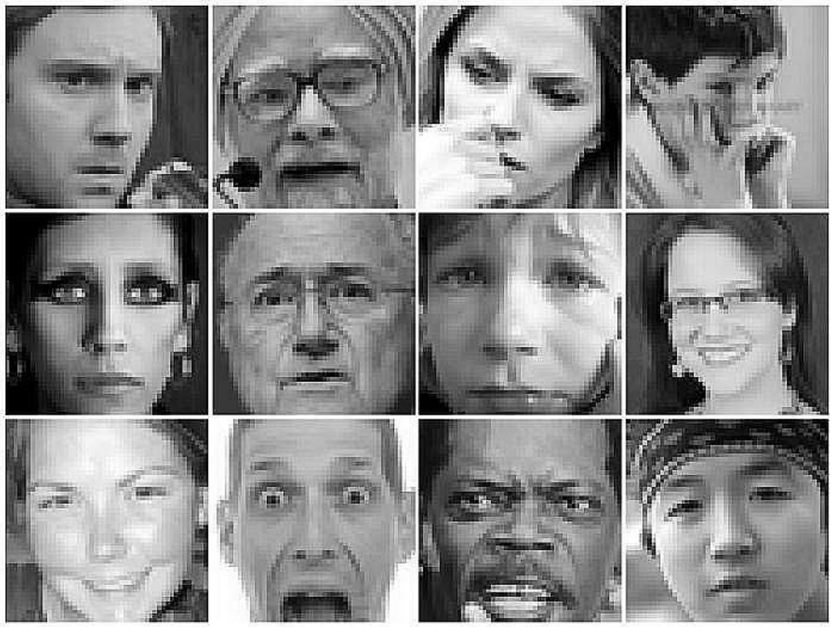


Figure 10: *Sharpened image*

Neural networks that use these images have shown an improvement in accuracy when compared to the original images. Although UnSharp Mask works well for FER 2013 dataset, it does not work for some of the other datasets like CelebA as the dataset consists of HD images while FER 2013 consists of blurry images.

VI. TRANSFER LEARNING

Transfer learning [9] is a very recent process introduced in deep learning that has gained popularity over the years. This technique makes use of a pre-trained CNN model to solve a problem rather than building a CNN architecture from scratch. A pre-trained model is a deep learning architecture that has been trained on a huge raw dataset for a long time. This process would adjust the weights of the model, and this model could be utilized for transfer learning [7]. The primary advantage of using a pre-trained model is that we do not need to train the entire deep learning model from scratch as the model's weights are already adjusted. Apart from the training time, the transfer learning model itself has shown better results than the same model that is built from scratch. Another key advantage of using the transfer model is when there is not enough data. Usually, the CNN models that are built from scratch would need a lot of data to adjust their weights, but transfer learning would not require a lot of data as the weights are already adjusted.

Transfer learning can be used to handle different types of scenarios, based on the similarity of the input and the required domains. Based on that, the method is of 3 types. They are:

- Inductive Transfer Learning
- Unsupervised Transfer Learning
- Transductive Transfer Learning

A. Inductive Transfer Learning -

Inductive Transfer Learning [7] is a technique in which the origin and required tasks are different, but they exist in a similar domain. The origin domain has a larger dataset, while the target domain has a limited dataset. So, the target labels induce some knowledge from the input domain to use in its functions. This leveraging from the source, in turn, improves the performance of the target task. This type of learning is frequently used and the one which would be used in this research.

B. Unsupervised Transfer Learning -

Unsupervised learning is a type of Machine Learning in which the dataset does not contain labels to classify the data, and it is totally up to the algorithm to perform classification on the data using the available details. Likewise, unsupervised transfer learning [7] is the same as Inductive transfer learning, but the source and the target datasets lack labels.

C. Transductive Transfer Learning -

In this scenario, the origin and the required tasks are similar, but they differ by domain. In addition to that, the target dataset does not contain labels.

VII. FACIAL EXPRESSION RECOGNITION WITH NEURAL NETWORKS

Facial Expression Recognition with neural networks involves extracting the facial features from the images using a branch of Artificial Neural Network called Convolutional Neural Network [25]. Convolutional Neural Network has been specially developed to deal with image datasets and classify the images [2]. This branch of Deep Learning involves the use of various techniques to extract the facial features from the images, including convolutions, and pooling, as well as the use of Fully connected layers from the Artificial Neural Networks, most commonly to perform classifications.

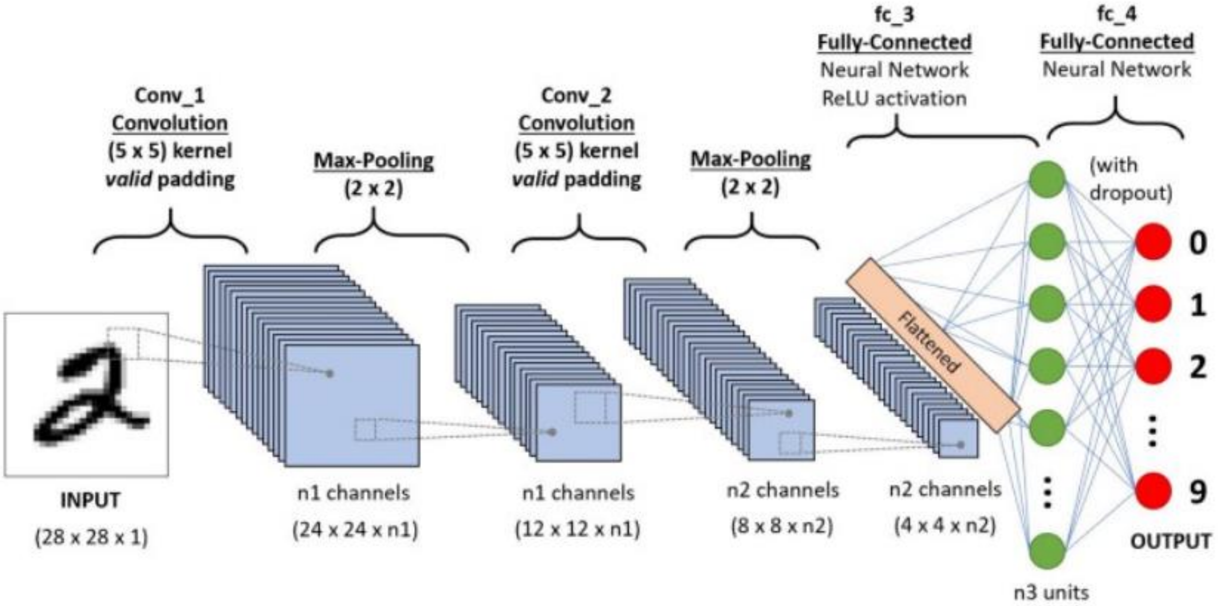


Figure 11: *A typical Convolution Neural Network architecture (Saha, S. (2018, December 17). A comprehensive guide to convolutional neural networks - the eli5 way. Retrieved September 28, 2022, from Medium. https://towardsdatascience.com/acomprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53)*

The primary advantages of using Convolutional Neural Networks are they are much faster and require less time to perform classification when compared to other techniques [ref from the image above]. An image is a matrix of pixel values. The convolutional neural networks perform way better as they can capture the temporal and spatial dependencies of an image, which is why is it preferred over feed-forward neural networks.

A. Convolutional Layer -

The convolutional layer is the primary component, which contains a number of filters or kernels, using which, during the training process we learn their parameters. The layer performs strided-convolutions on the input image to essentially gather features from the input. Note that, if we are using a convolutional layer with 256 neurons, the values of each of these neurons or filters will not be the same. Some filters when applied would blur the image, some would brighten or darken the image. Using different filters gives us many advantages, as certain features can be extracted from the image only under certain conditions. Each filter would stride over the input image and would perform dot product using the pixel data from the image, and its matrix data.

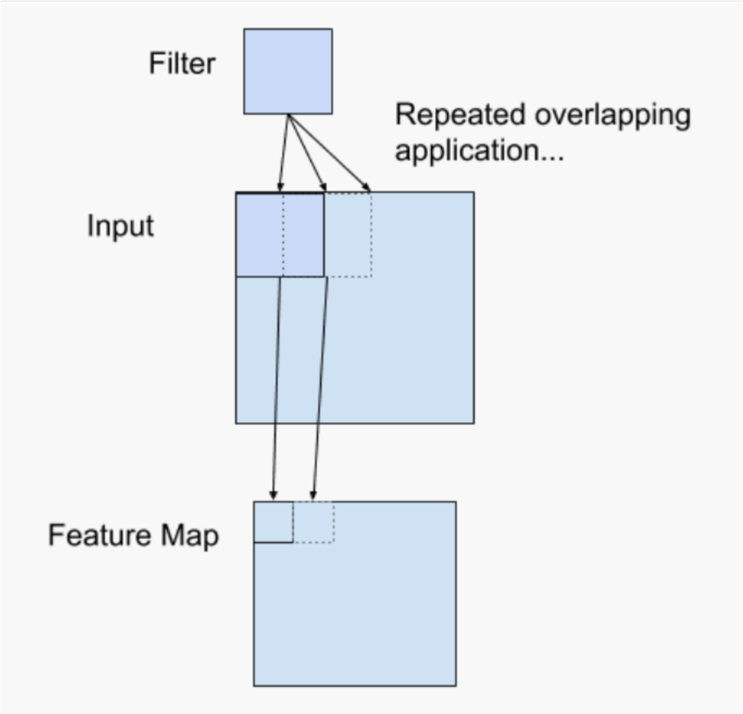
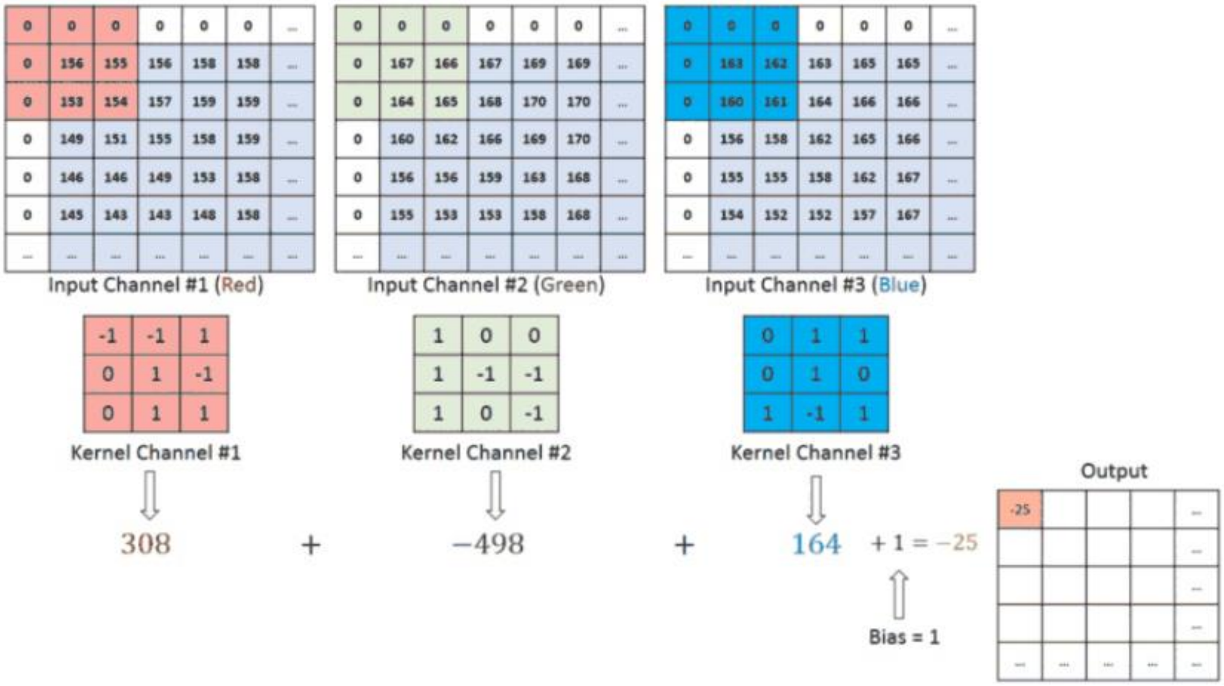


Figure 12: *Repeated Overlapping application of the filter on the Input image (Brownlee, J (2019, April 17). How do Convolutional Layers Work in Deep Learning Neural Networks? Retrieved September 15, 2022, from Machine Learning Mastery. https://machinelearningmastery.com/convolutional-layers-for-deep-learning-neural-networks/)*

From the above image, we can see that the filter, which is usually smaller than the input image, is repeatedly applied to the input image in strides. The strides are user-defined, and we could jump (stride) as many pixels as we want. A key aspect to record her is the lower the stride length, the bigger the feature map would be.



Input Channel #1 (Red)

0	0	0	0	0	0	...
0	156	155	156	158	158	...
0	153	154	157	159	159	...
0	149	151	155	158	159	...
0	146	146	149	153	158	...
0	145	143	143	148	158	...
...	...	...	...	...	...	...

Input Channel #2 (Green)

0	0	0	0	0	0	...
0	167	166	167	169	169	...
0	164	165	168	170	170	...
0	160	162	166	169	170	...
0	156	156	159	163	168	...
0	155	153	153	158	168	...
...	...	...	...	...	...	...

Input Channel #3 (Blue)

0	0	0	0	0	0	...
0	163	162	163	165	165	...
0	160	161	164	166	166	...
0	156	158	162	165	166	...
0	155	155	158	162	167	...
0	154	152	152	157	167	...
...	...	...	...	...	...	...

Kernel Channel #1

-1	-1	1
0	1	-1
0	1	1

Kernel Channel #2

1	0	0
1	-1	-1
1	0	-1

Kernel Channel #3

0	1	1
0	1	0
1	-1	1

$$308 + -498 + 164 + 1 = -25$$

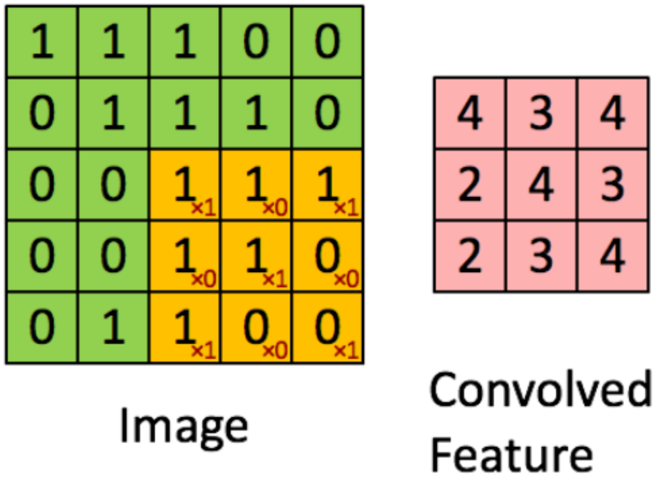
Bias = 1

Output

-25				...
				...
				...
				...
...	...	...	...	...

Figure 13: Kernel operation on input with depth > 1 (Saha, S. (2018, December 17). A comprehensive guide to convolutional neural networks - the eli5 way. Retrieved September 15, 2022, from Medium. https://towardsdatascience.com/acomprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53)

It is possible that some of the image data could be lost during the striding process, particularly from the left and bottom edges. Zero padding helps to overcome this problem by adding as many rows or columns of zeroes as needed to fit the filter size, thereby producing the output image in the exact dimensions of the input image.



1	1	1	0	0
0	1	1	1	0
0	0	$1_{\times1}$	$1_{\times0}$	$1_{\times1}$
0	0	$1_{\times0}$	$1_{\times1}$	$0_{\times0}$
0	1	$1_{\times1}$	$0_{\times0}$	$0_{\times1}$

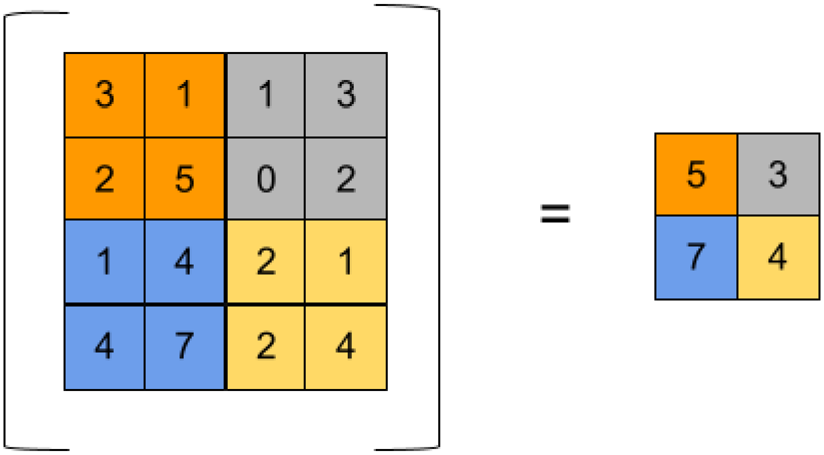
4	3	4
2	4	3
2	3	4

Figure 14: *3x3x1 kernel operation on 5x5x1 image (Saha, S. (2018, December 17). A comprehensive guide to convolutional neural networks - the eli5 way. Retrieved September 15, 2022, from Medium. https://towardsdatascience.com/acomprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53)*

In the above image, we can see that a 3x3 filter is applied to a 5x5 image to extract the convolved feature.

B. Pooling Layer -

Pooling layers are present to extract specific characteristics from the image data and reduce the image dimensionality. Unlike strided-convolutions, pooling is not a strided operation. The two most common pooling types are Max pooling and Average pooling. As the name denotes, we select the biggest pixel from the given area for max pooling while we consider the total average of the pixels in the given area for average pooling. Researchers use Max pooling over Average pooling frequently as it eliminates noise in the data along with reducing the dimensionality as well.

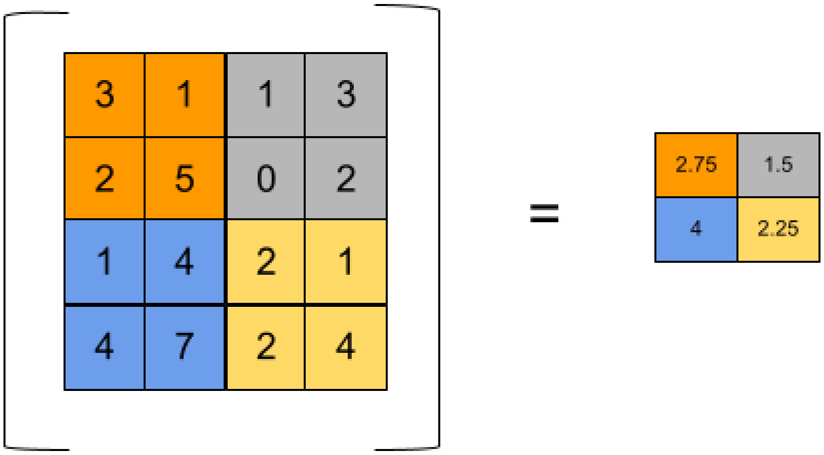


3	1	1	3
2	5	0	2
1	4	2	1
4	7	2	4

=

5	3
7	4

Figure 15: *Max pooling on input feature map ((2021, August 29). Explain Pooling Layers: Max pooling, Average Pooling, Global Average Pooling and Global Max Pooling. Retrieved September 15th 2022, from Knowledge Transfer. https://androidkt.com/explain-pooling-layers-max-pooling-average-pooling-global-average-pooling-and-global-max-pooling/)*



3	1	1	3
2	5	0	2
1	4	2	1
4	7	2	4

=

2.75	1.5
4	2.25

Figure 16: *Average pooling on input feature map ((2021, August 29). Explain Pooling Layers: Max pooling, Average Pooling, Global Average Pooling, and Global Max Pooling. Retrieved September 15th, 2022, from Knowledge Transfer. https://androidkt.com/explain-pooling-layers-max-pooling-average-pooling-global-average-pooling-and-global-max-pooling/)*

Apart from Max pooling and Average pooling, we have Global Average Pooling and Global Max Pooling. Global Average pooling is used to replace the Flatten layer. A Flatten layer sits joining the final CNN layer and the softmax regression layer in a CNN model. The layer “flattens” out the convolutional layer, treats them as feature extractors and helps to convert the data onto a classifiable form. The problem with this layer is that it is a Fully connected layer, and it is prone to overfit. We could replace this layer with a Global Average pooling layer, as the pooling layer can generate feature maps for every set of classifications.

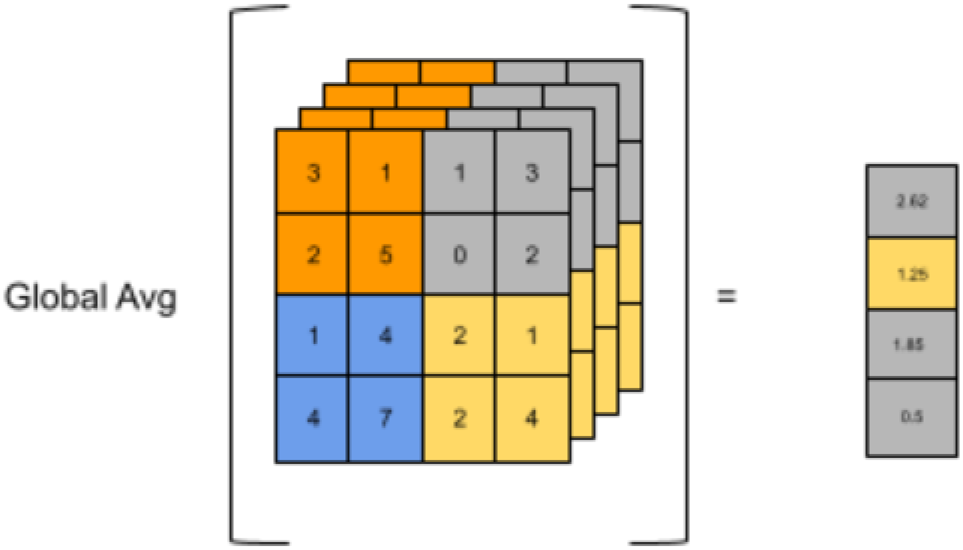


Figure 17: *Average pooling on input feature map ((2021, August 29). Explain Pooling Layers: Max pooling, Average Pooling, Global Average Pooling, and Global Max Pooling. Retrieved September 15th, 2022, from Knowledge Transfer. https://androidkt.com/explain-pooling-layers-max-pooling-average-pooling-global-average-pooling-and-global-max-pooling/)*

Global max pooling works similarly to Max pooling, but it takes the maximum over the time dimension. It could be used to replace the Flatten and Dense layers in a neural network.

C. ResNet Architecture -

ResNet (Residual Network) architecture is a type of Convolutional Neural Network. It is, thus far, one of the most efficient deep learning neural networks over the last few years. It is a common practice to include many layers in neural networks. The depth of the model depends on the intricacy of the input dataset. If there are more features to be detected by the neural network, it would require more layers. But simply stacking the layers one after the other would not work in neural networks, as there exists a problem of vanishing gradient. This problem arises due to back-propagation, where we tune the weights of the network from the output layer to the input layer in a backward fashion. Hence, repeated calculations make the gradient smaller in each step, and it would eventually get saturated and might degrade the performance of the model. As a consequence of the vanishing gradient problem, the error rate of the model would also increase as the layer count rises.

To overcome this problem, ResNet architecture contains identity blocks or residual blocks which are "Identity shortcut connections" which skip a certain number of layers.

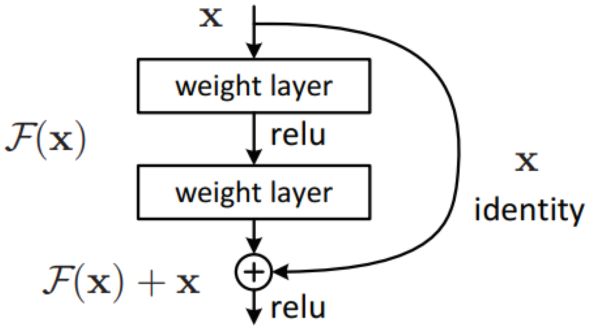

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 18: *Skip connection of the ResNet architecture (Seb. (2022, February 1). An Introduction to Residual Skip Connections and ResNets. Retrieved September 28, 2022, from Programmathically. https://programmathically.com/an-introduction-to-residual-skip-connections-and-resnets/)*

These skip connections make sure that when a set of layers degrades the performance of the model, the entire set of layers is skipped using normalization. In the case where the additional layers are useful, the performance of the model would increase. Therefore, the presence of the skip connection would either increase the performance or skip the connection altogether and would not decrease the performance of the model. Hence, this type of architecture helps to train very deep architectures without facing the vanishing gradient problem.

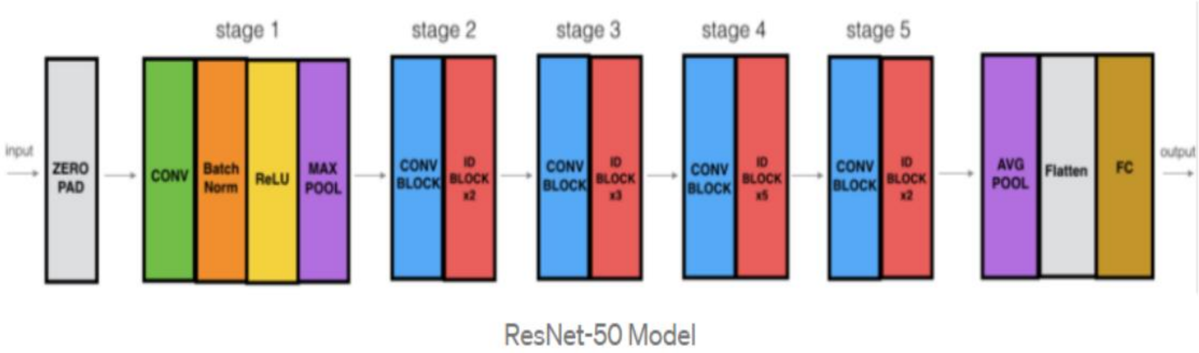


Figure 19: *Resnet-50 architecture (Dwivedi, P. (2019, March 27). Understanding and Coding a ResNet in Keras - Towards Data Science. Retrieved September 28, 2022, from Medium. https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras 446d7ff84d33#:%7E:text=The%20ResNet%2D50%20model%20consists,over%2023%20million%20trainable%20 parameters.&text=Our%20ResNet%2D50%20gets%20to,in%2025%20epochs%20of%20training.)*

There are 5 established architectures of ResNet -

(a) ResNet 18

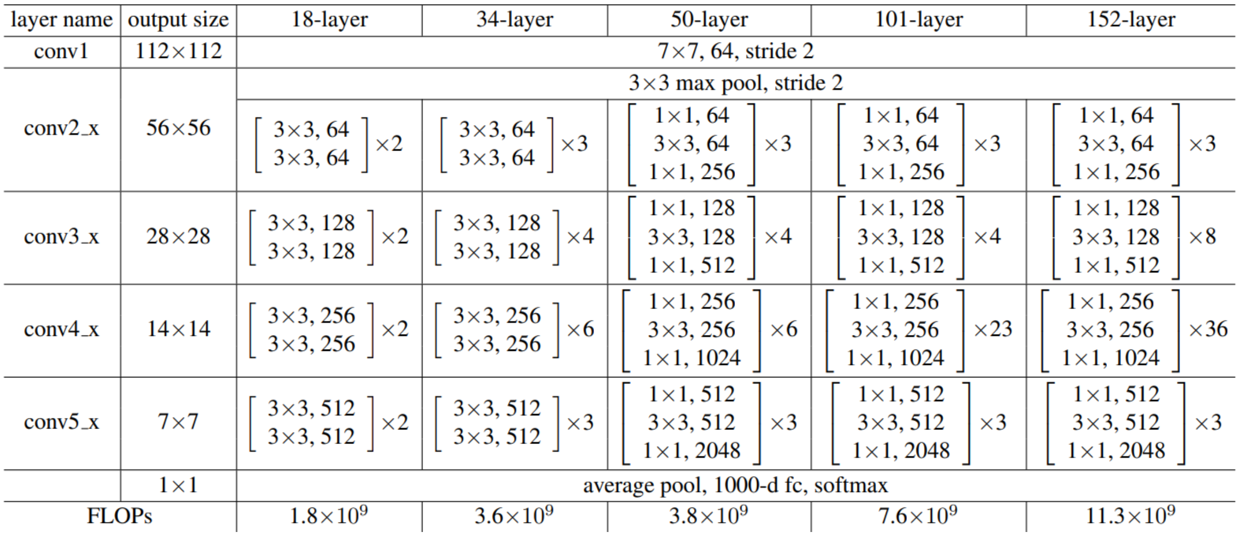
(b) ResNet 34

(c) ResNet 50

(d) ResNet 101

(e) ResNet 152

We are going to be using ResNet-50 architecture in this project. Here are the architectural comparisons between the different ResNet models.



layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 20: *Different ResNet architectures (Hassan, M. U. (2019, January 23). ResNet (34, 50, 101): Residual CNNs for Image Classification Tasks. Retrieved September 29, 2022, from Neurohive. https://neurohive.io/en/popular-networks/resnet/)*

D. SeNet Architecture -

Senet is a mechanism that was introduced in Convolutional Neural Networks to improve the performance of the neural networks by introducing a Squeeze and Excitation Network. The primary reason to add this component to the neural network is to improve the existing attention mechanism in the existing CNN architectures. The Squeeze and Excitation Network focuses on the channels of attention mechanism and is an attempt to rectify each of these channels to produce a better depiction of them, by improving their characteristics. The presence of this module improves the interdependencies between the channels. Similar to ResNet architectures, the SeNet

[37] architectures either improve the performance of the model or ignore the irrelevant features, therefore do not diminish the performance of the network.

The primary motivation for this network comes from the fact that different layers in a CNN architecture recognize different features in the images, such as the upper layers identifying complete objects while the lower layers detect features like edges. Eventually, all these detected features are mapped together through information transfer via channels in each layer.

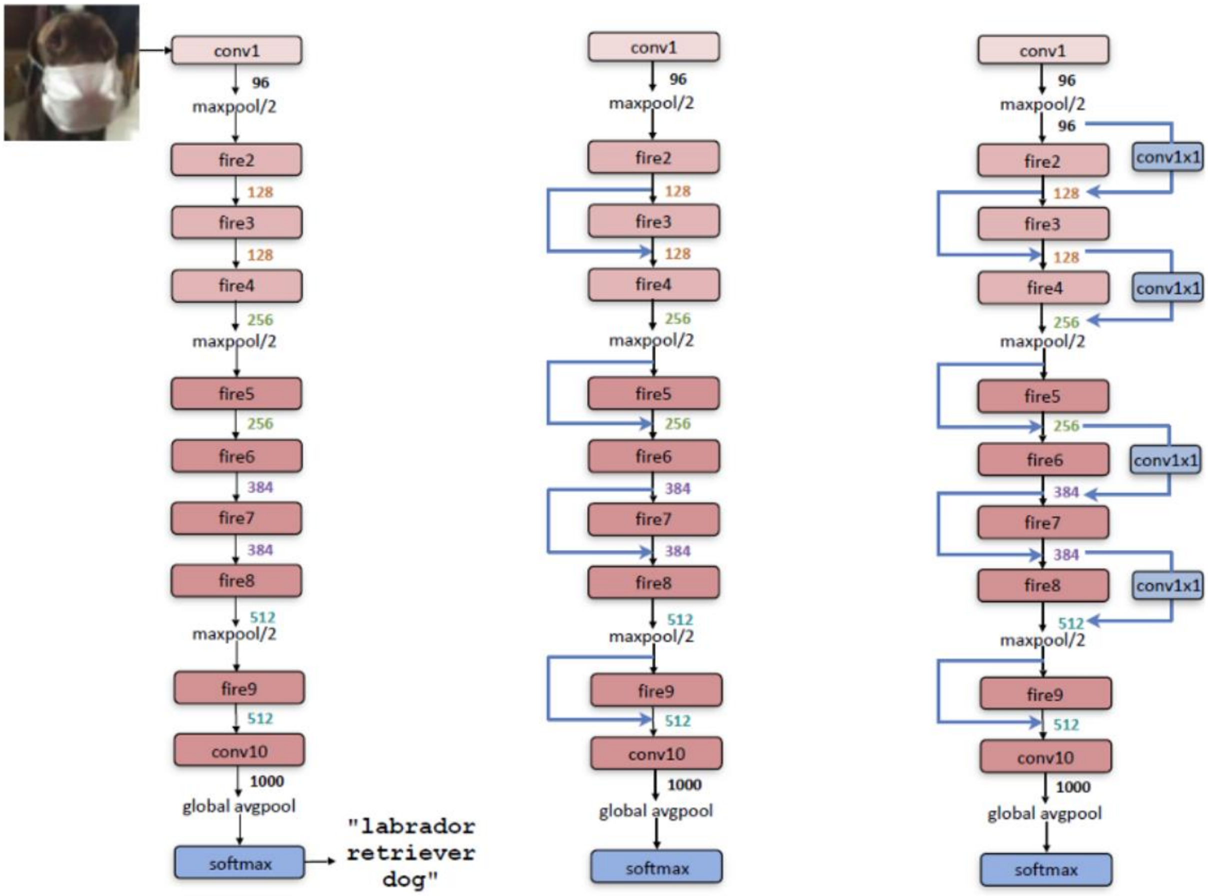


Figure 21: *SqueezeNet architecture (Tsang, S. (2019, April 22). Review: SqueezeNet (Image Classification) - Towards Data Science. Retrieved September 29, 2022, from Medium. https://towardsdatascience.com/review-squeezenet-image-classification-e7414825581a)*

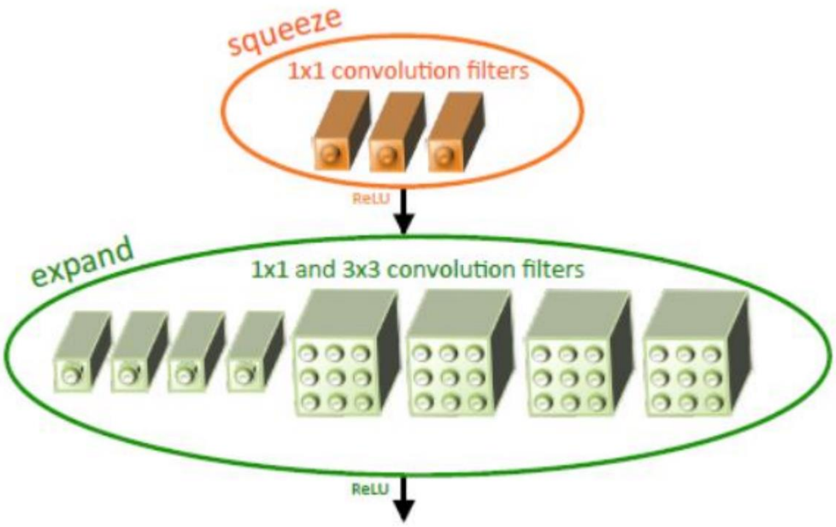


Figure 22: *Fire module in SqueezeNet – Sqeeuze+expand (Tsang, S. (2019, April 22). Review: SqueezeNet (Image Classification) - Towards Data Science. Retrieved September 29, 2022, from Medium. https://towardsdatascience.com/review-squeezenet-imageclassification-e7414825581a)*

The Squeeze and Excitation component in the SeNet consists of 3 operations.

1) Squeeze
2) Excitation
3) Scaling

1. Squeeze -

As the term suggests, the squeeze operation "squeezes" or, in other words, extracts information from each of the channels present in the network. Convolution is a mechanism where there are hundreds of channels, and each channel operates. Hence, each channel would be seeing a smaller part of the image and there must exist a mechanism that reduces the parameter count and decreases the complication of the model. Though in newer CNNs, Global Max Pooling is utilized, in SeNet Global Average Pooling is preferred.

The result of a convolutional layer is a 4D tensor of size (B x H x W x C) where:

- B is the count of data in the batch
- H is the height
- W is the width
- C is the channel count

After performing Global Average Pooling, the size of the 4D tensor reduces to (B x 1 x 1 x C) from (B x H x W x C).

2. *Excitation -*

Now that the dimensions of the 4D tensor are minimized, we can use a fully connected Multi-Layer perceptron having 3 layers, in which the middle layer exists for decreasing the feature count by using a reduction factor **r**, to generate the weights and to adapt every channel that exists in the given feature map.

Input to the MLP is a 4D tensor, having a size (B x 1 x 1 x C) hence there is a C number of neurons present. The neuron count gets reduced by a matter of C / r, where r is the reduction factor. Subsequently, in the output, the neurons count increases back to the original count (C).

3. *Scaling* -

Output from the Excitation layer is passed to an activation function, which converts the excited tensors into numeric values within 0 and 1. The output of the sigmoid function is then multiplied by the input tensor, where the elements closer to 0 are the channels that are less important and can be reduced, while the values that are close to 1 are more important.

This helps as when we multiply the input tensor with a value that is close to 0, we get a reduced pixel value, which is smaller when compared to the input tensor, while the product of the input tensor to the value that is closer to 1 will be better when compared to the previous case. Hence, the network reduces irrelevant information while preserving useful information.

VIII. PROPOSED MODELS

In this research, our key focus is to improve the efficiency of the deep learning models when they are trained on the FER datasets. In general, the training of CNNs on FER datasets has some challenges. One major problem is that the models overfit frequently, as some images are labeled incorrectly in the dataset. As a result, higher amounts of mislabeled data could prevent the model from converging in the early stages of the process.

The primary design of the model included the addition of CNN layers to the pre-trained model. Upon adding them, we freeze the model, leaving out the Batch Normalization layers in the model, and train for several epochs. After the training, the entire model is unfrozen and re-trained for several epochs. This helps to adjust the weights of the network and improves the accuracy of the model.

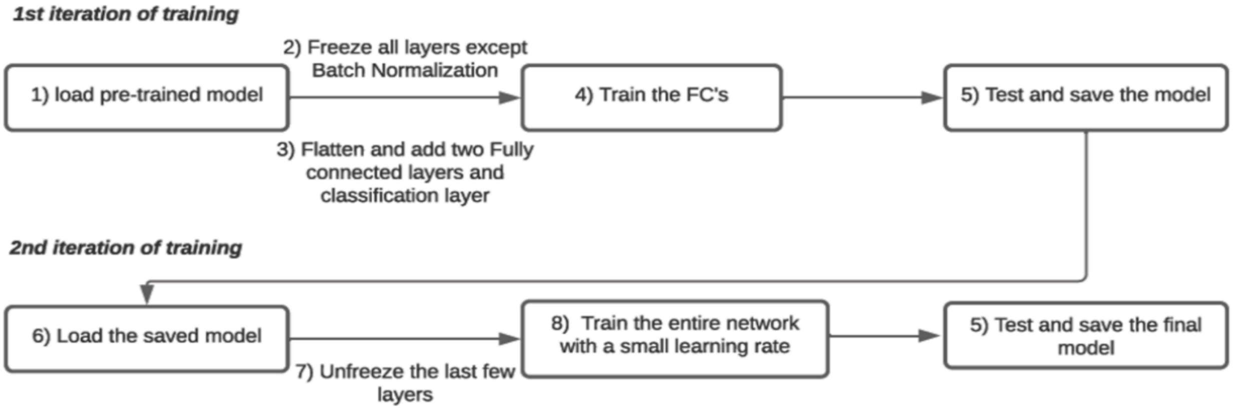

1st iteration of training
1) load pre-trained model
2) Freeze all layers except Batch Normalization
3) Flatten and add two Fully connected layers and classification layer
4) Train the FC's
5) Test and save the model
2nd iteration of training
6) Load the saved model
7) Unfreeze the last few layers
8)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

Figure 23: Transfer learning approach pipeline [51]

Apart from the CNN layers, a set of Dropout layers, a flattened layer, and a final softmax layer are also added to the model. Apart from the model itself, useful packages such as

ModelCheckpoint, EarlyStopping, and ReduceLROnPlateau are used to keep track of the accuracy during the training process. Model Checkpoint is used to save a copy of the model whenever a metric improves. For example, whenever the validation accuracy improves, we could save a checkpoint. Early Stopping monitors the given metric and stops the training early if the monitored metric does not improve over a defined set of epochs. For example, whenever the accuracy of the validation data stays the same for over 10 epochs, the model stops training. ReduceLROnPlateau reduces the learning rate of the training process whenever a metric stops improving. Usually, this happens on a scale of 2 to 10 times. The combination of EarlyStopping and ReduceLROnPlateau makes sure that the model does not continue training forever without seeing an improvement in accuracy.

The addition of two Convolutional layers (and flatten layer, Activation layers, and Dropout layers along with a final softmax layer) could also add to the overfitting problem. Experiments were performed on a different set of filters for the two layers. These values change for all of these models as they depend on the final layer in the pre-trained network. For example, in a ResNet-50 model, the final layer has 2048 filters.

Our goal is to create a better model with not just a higher accuracy, but also a much more optimal model.

A. ResNet - 50 -

The ResNet 50 [36] model was initially trained on 240x240 images. As the images in FER 2013 dataset are of the size 48x48, we resize them. We had to take additional care to not reduce the input dimensions lesser than 197 as the model breaks for any dimensions lesser than that. So, we re-sized the input image dimensions to 197x197 and reduced the input dimensions for the ResNet model as well. We freeze the model, leaving out the Batch Normalization layers so that they get adjusted to this type of dataset. We also added a couple of Convolutional layers containing 1024 neurons and 512 neurons respectively, along with dropout layers and a Softmax layer at the end.

Concerning optimizers, we used Stochastic Gradient Descent with a learning rate of 0.01 and a decay of 0.0001. The model was trained on the data from ImageDataGenerator for 100 epochs with a batch size of 128, and we achieved 67.4 % validation accuracy.

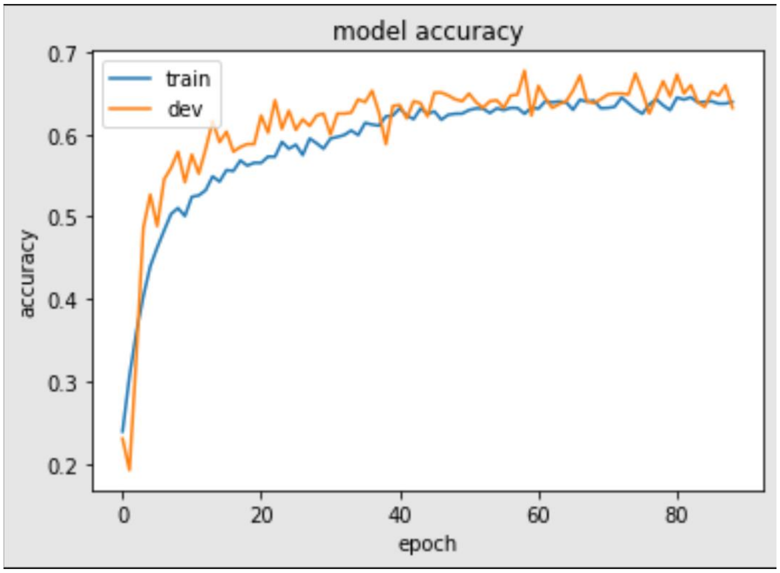


Figure 24: Accuracy Graph of Frozen ResNet - 50 network

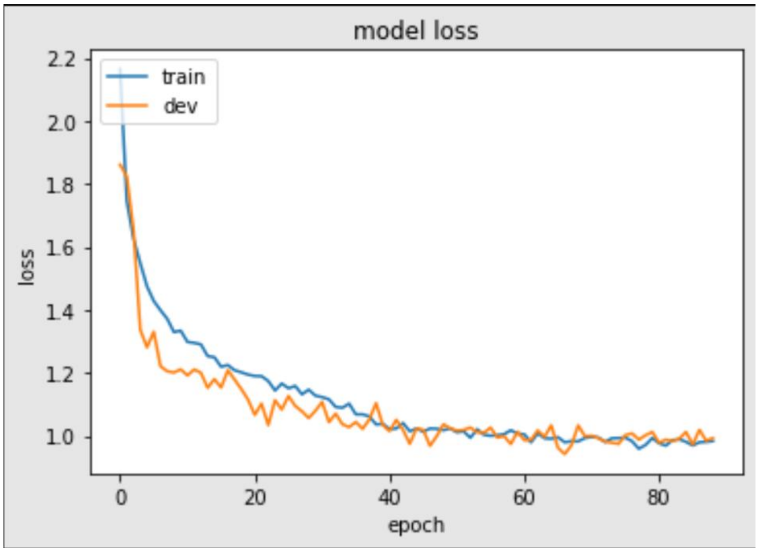


Figure 25: *Loss Graph of Frozen ResNet 50 network*

Once, we relax the network and retrained them for 100 epochs using the same batch size, the accuracy of the new model touched 74.08%.

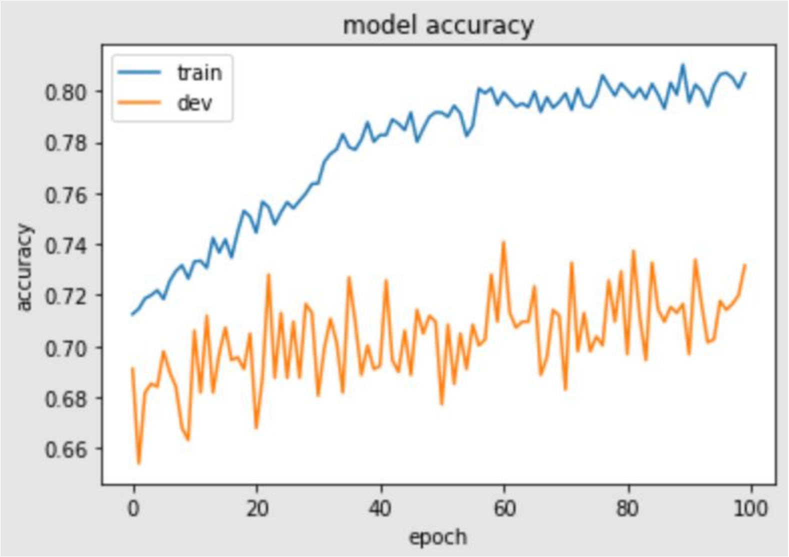


Figure 26: *Accuracy Graph of unfrozen ResNet - 50 network*

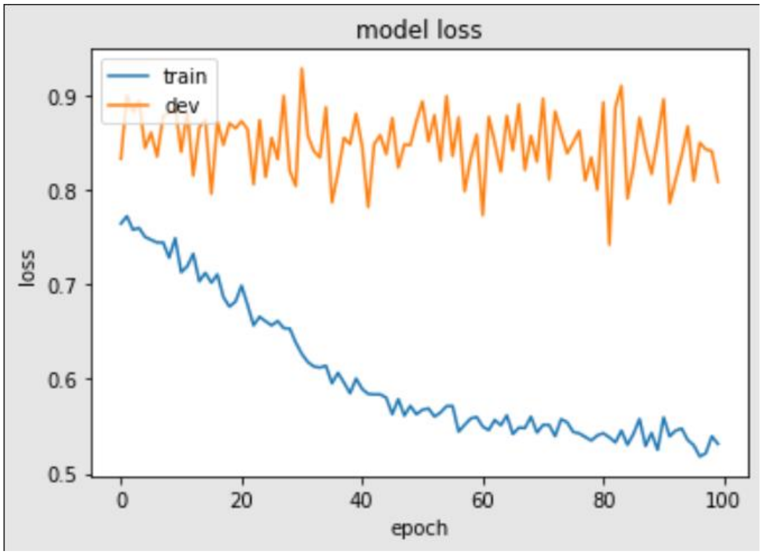


Figure 27: *Loss Graph of unfrozen ResNet - 50 network*

When we included the image-preprocessing techniques in this model, the Histogram Equalization method started overfitting, and we achieved a Validation accuracy of less than 65%. Though the performance did not get better when we used UnSharp Mask, it was fairly better than the model with Histogram Equalization.

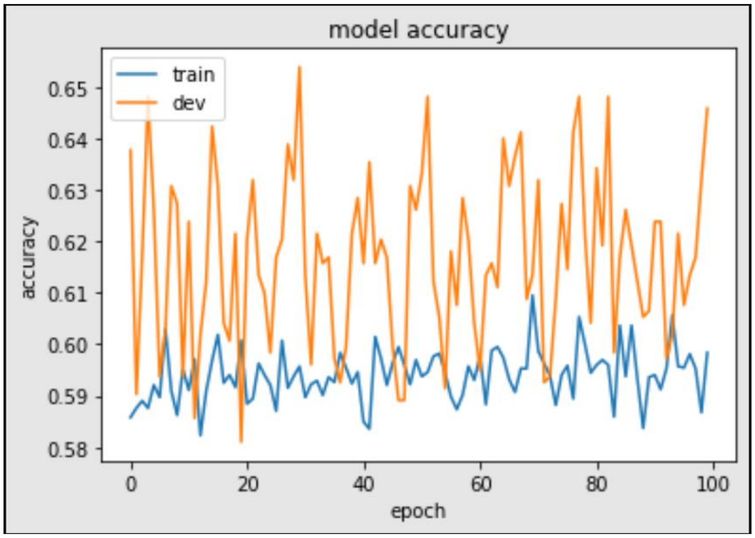


Figure 28: *Accuracy Graph of Frozen ResNet - 50 network with UnSharp Mask*

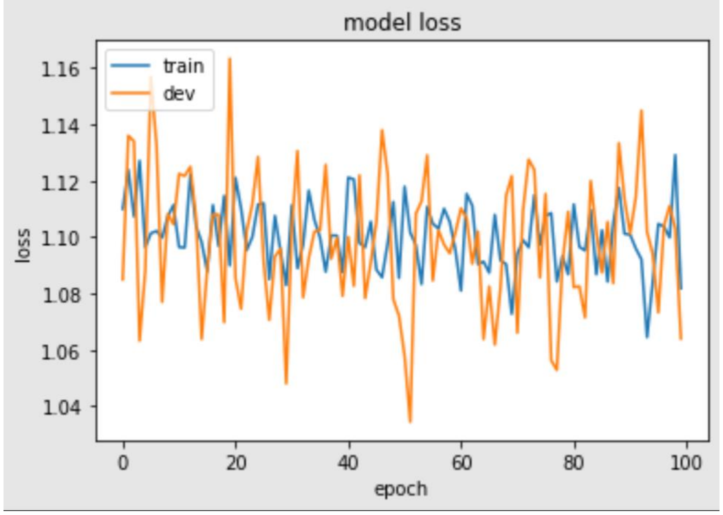


Figure 29: Loss Graph of Frozen ResNet - 50 networks with UnSharp Mask

The frozen ResNet architecture achieved the highest validation accuracy of 65% in this case.

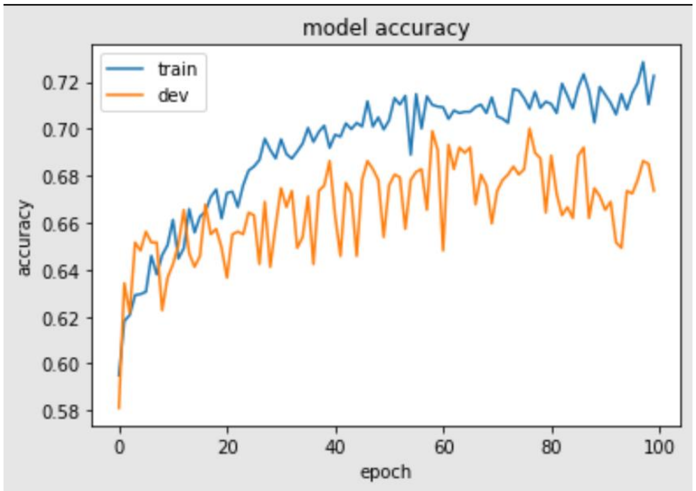


Figure 30: *Accuracy Graph of unfrozen ResNet - 50 network with UnSharp Mask*

We can also note that the validation loss is higher than the ResNet model without using Unsharp Mask. Hence, sticking with the preprocessing techniques that were provided by Keras (Image Data Generator) seemed efficient for this model.

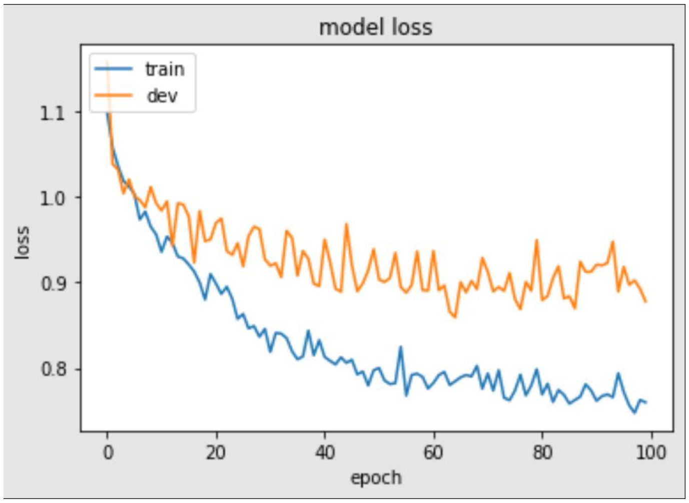


Figure 31: Loss Graph of Frozen ResNet - 50 network with UnSharp Mask

B. SeNet - 50 -

For Senet-50, we followed a method, similar to that was used for Resnet-50. We resized the data to 197x197 to make sure the deep neural network does not break. Initially, we freeze the model leaving out the Batch Normalization layers and newly attached fully connected layers at the end of the network. Along with these, we added Dropout layers to make sure the model does not overfit. Using SGD, the model was trained for 100 epochs for a batch size of 128 which produced a validation accuracy of 66.6 %.

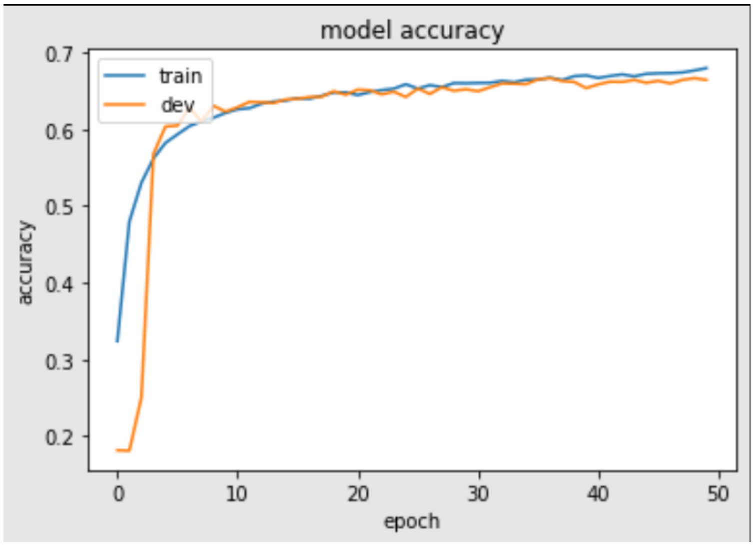


Figure 32: Accuracy Graph of frozen SeNet - 50 network

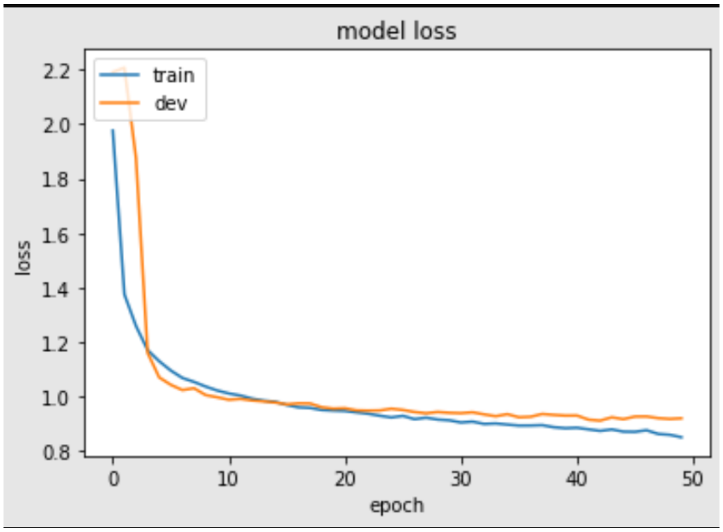


Figure 33: Loss Graph of frozen SeNet - 50 network

When we re-trained the Senet-50 model, we kept the first 200 layers of the Senet model frozen. The reason for this is that the model consists of a lot of parameters, leading to an “Out of memory” exception while training on Google Colab, due to the limitation of resources. The rest of the layers were relaxed and fine-tuned with the previously attached dense layers for 100 epochs and a batch size of 128, with a learning rate of 0.001. The highest validation accuracy that we were able to achieve was 71.38%.

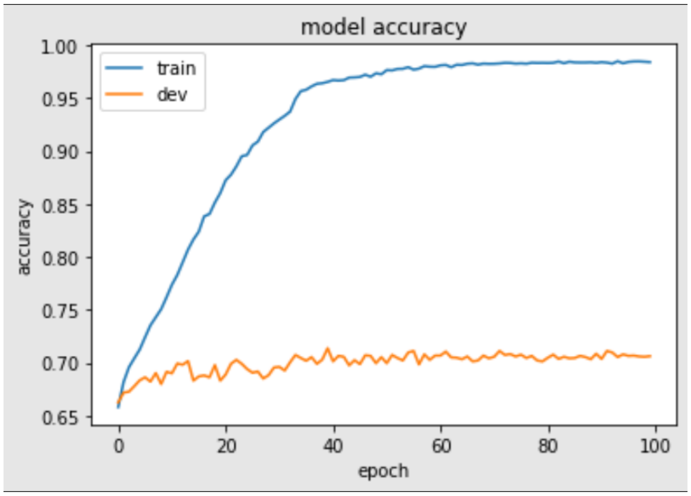


Figure 34: *Accuracy Graph of unfrozen SeNet - 50 network*

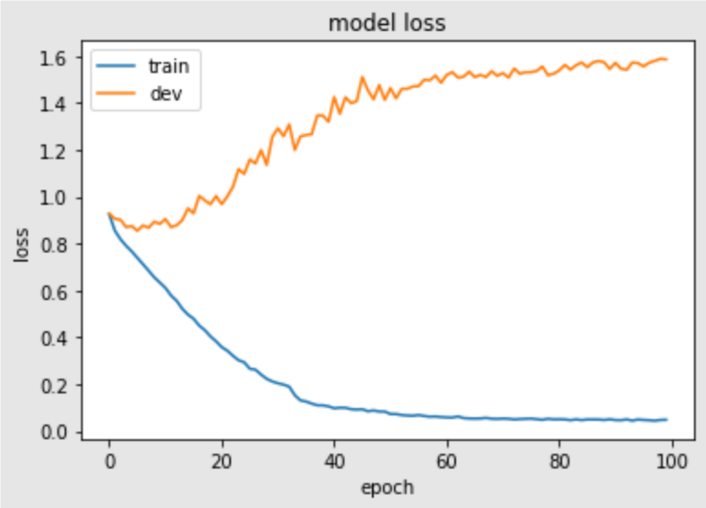


Figure 35: *Loss Graph of frozen SeNet - 50 network*

We can also see that the model started to overfit at a point (the training accuracy by the end of 100 epochs was 98% and the training loss was almost 0.0). We believe that the dense layers, each of them having 1024 and 512 filters play a role in this, as previous research [51] on this model containing two dense layers having 4096 and 1024 filters respectively yielded a much better accuracy of 73.3%.

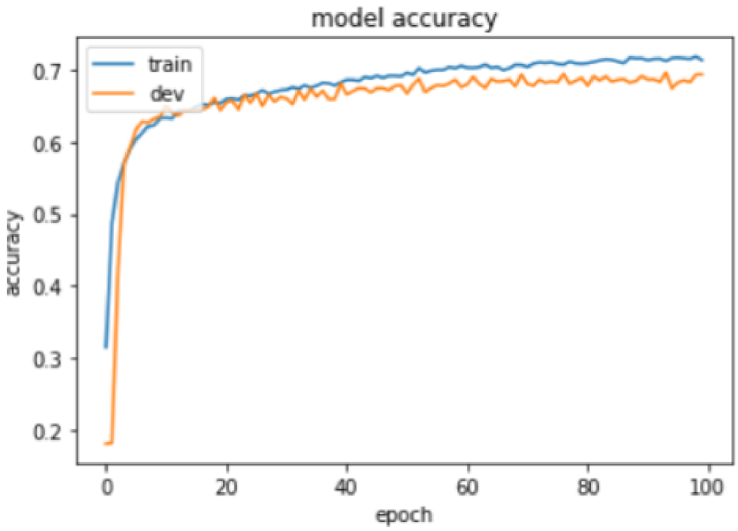


Figure 36: *Senet-50 Accuracy Graph (Vepuri, Ksheeraj Sai, "Improving Facial Emotion Recognition with Image processing and Deep Learning" (2021). Master's Projects. 1030. DOI: https://doi.org/10.31979/etd.3wrz-53ee)*

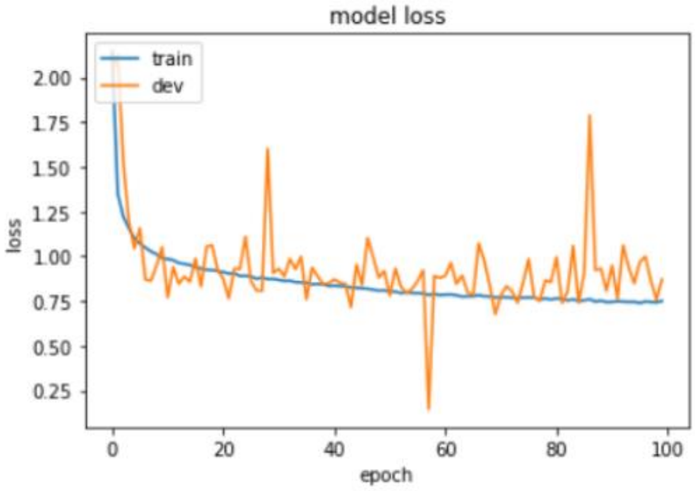


Figure 37: *Senet-50 Loss Graph (Vepuri, Ksheeraj Sai, "Improving Facial Emotion Recognition with Image processing and Deep Learning" (2021). Master's Projects. 1030. DOI: https://doi.org/10.31979/etd.3wrz-53ee)*

The inclusion of pre-processing technique UnSharp mask did not improve the model performance, as the validation failed to improve by over 71.1%.

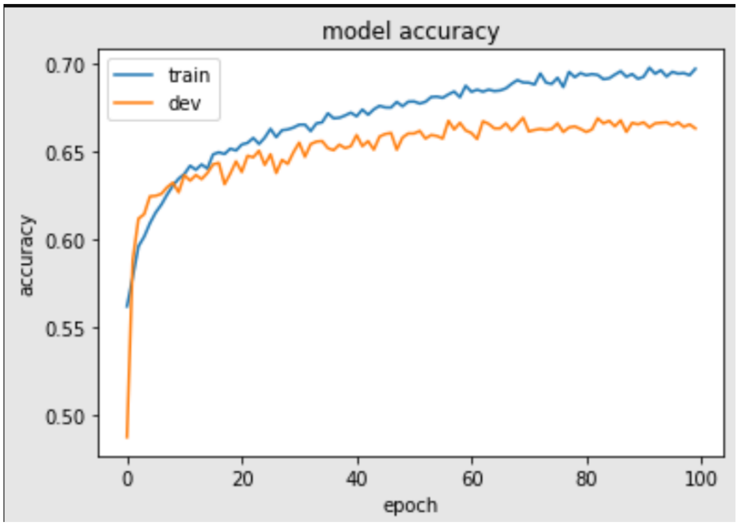


Figure 38: Accuracy Graph of frozen SeNet - 50 network

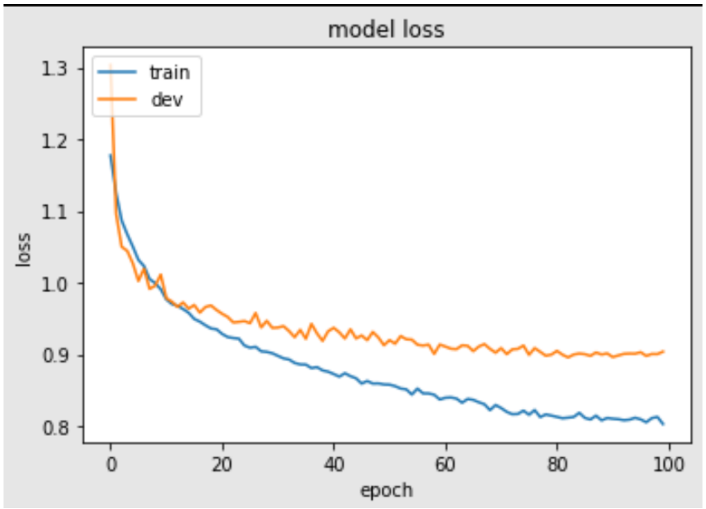


Figure 39: Loss Graph of frozen SeNet - 50 network

Even in this case, the fine-tuned model started to overfit, with the training accuracy touching 98% and the training loss touching 0.001.

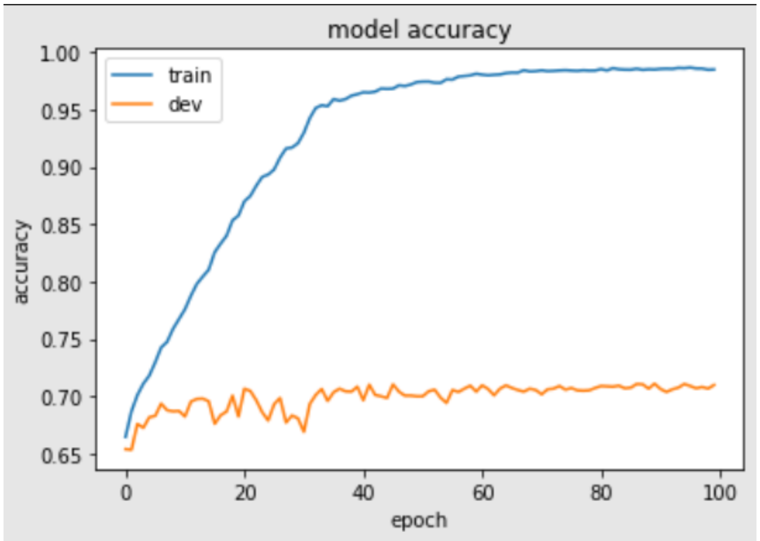


Figure 40: Accuracy Graph of unfrozen SeNet - 50 network

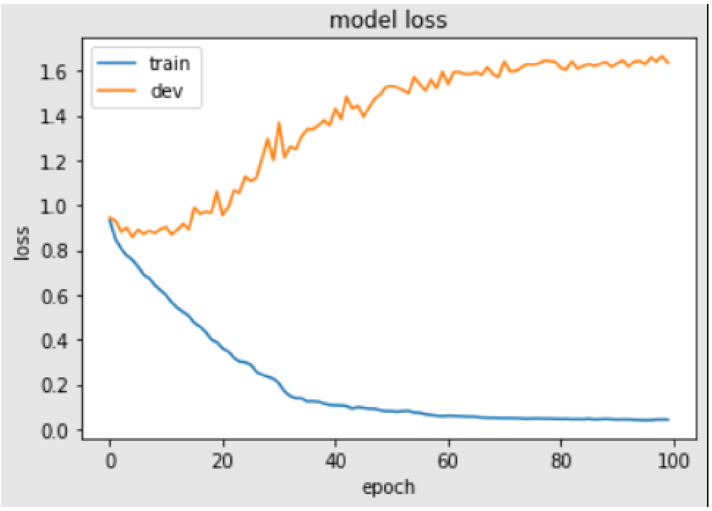


Figure 41: Loss Graph of unfrozen SeNet - 50 network

IX. RESULTS

Out of the 2 models that we used with and without pre-processing methods, Resnet-50 architecture achieved the highest validation accuracy of 74.1%. Out of the experiments that were performed on the 2 models, Resnet-50 and SeNet-50, with and without Image Sharpening, we did not work more on Resnet-50 with the UnSharp mask as the model started overfitting right from the start. Application of preprocessing yielded better results and improved the model's working in many of the previous experiments [30] [10] [11] [51] but in our case, the model converged when using Image Sharpening. The accuracies that we achieved for Resnet-50 with and without Image Sharpening were 74.1 % and 71.2% respectively, and the accuracies that we achieved for Senet-50 with and without Image Sharpening were 71.3% and 71.1% respectively. These outperformed some of the results from past research which used pre-trained architectures [28] [25] [51].

Figures 21 and 22 represent the training and validation accuracy of the Resnet-50 model. We can also see that most of the pre-trained models had a similar performance, with the accuracies remaining in the range of 69% to 72%. Our Resnet-50 model employed a similar architecture to that of Vepuri's research [51], except for using two dense layers of 1024 and 512 neurons respectively, and outperformed his model by 0.3%. But at the same time, the Resnet-50 started to overfit when using Image Sharpening which did not happen in his case. In addition to that, his Senet-50 models outperformed the models used in our research.

Table 2: *Summary of test accuracies*

Model	**Preprocessing**	**Test Accuracy (%)**
Resnet-50	Data Augmentation	**74.08**
Resnet-50	Data Augmentation + Unsharp Mask	72.1
Resnet-50	Data Augmentation + Histogram Equalization	72.3
Senet-50	Data Augmentation	71.3
Senet-50	Data Augmentation + Unsharp Mask	71.1

Table 3: Accuracies of the past researches

Previous Studies	**Model**	**Test Accuracy**
Vepuri	Resnet-50	**73.8**
Khanzada et al.	Resnet-50	73.2
	Senet-50	70.0
Pramerdorfer et al	Resnet-50	72.4

X. GENERATE ADVERSARIAL NETWORKS

Generative Adversarial Networks [44] are a type of modeling technique that is performed via deep learning architectures such as Convolutional Neural Networks. Similar to them, this technique involves identifying and learning patterns from the input data, hence this technique is unsupervised learning. At the same time, the Discriminator performs classification and hence, this is interpreted cleverly as a supervised learning problem. The architecture of GANs involves two CNN models - a Generator and a Discriminator. The primary job of the former is to develop a noise randomly and send it to the latter. The Discriminator compares this generated noise with the original input image and does a similarity test. If the image is different from the input image, the Discriminator would reject it, and the Generator would update the image, and generate new images. This process would go on until the latter is fooled by the former into accepting the fake noise as the original image.

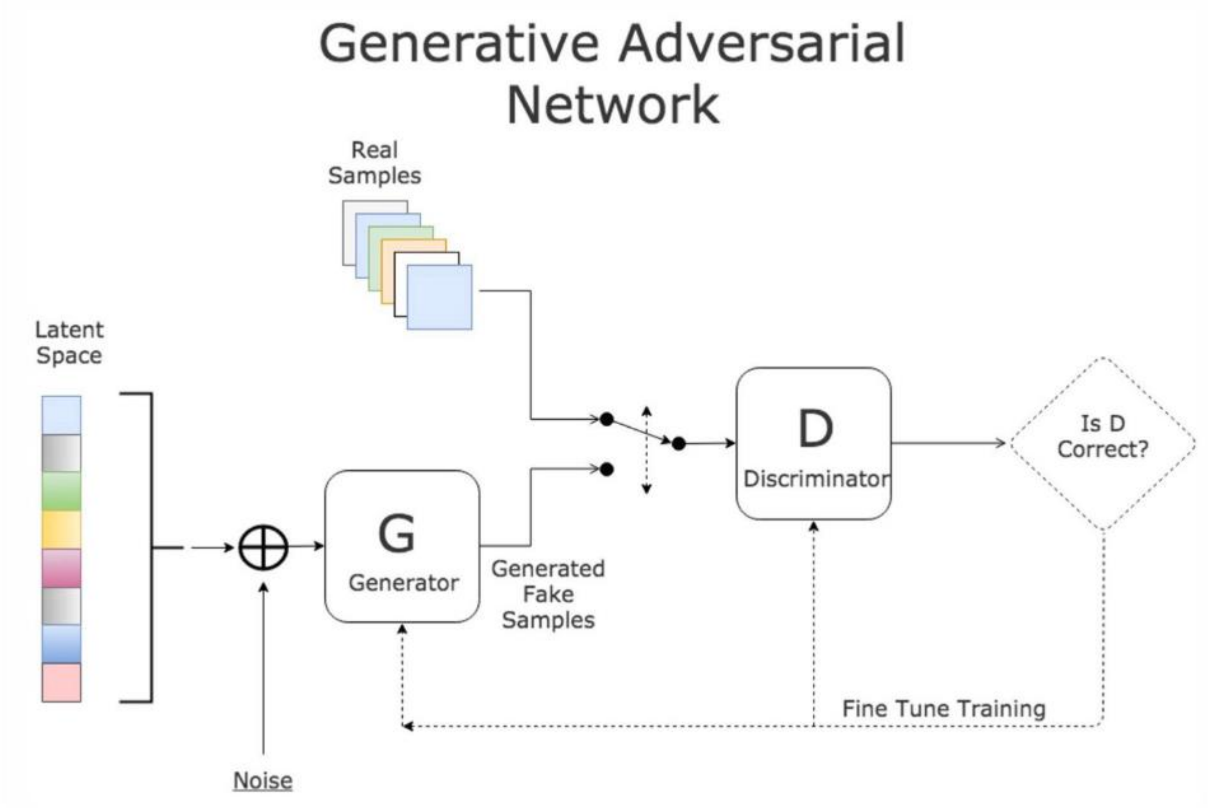


Figure 42: *Global concept of GANs (Mosquera, D. G. (2018, February 1). GANs from Scratch 1: A deep introduction. With code in PyTorch and TensorFlow. Retrieved October 6, 2022, from https://medium.com/ai-society/gans-from-scratch-1-a-deep-introduction-with-code-in-pytorch-and-tensorflow-cb03cdcdba0f)*

Upon training, the quality of the images gradually improves, and we achieve good results in the end. Similar to a CNN model, the GANs iterations converge at a point and the generator simply stops producing better images. Here is an example of a GAN model producing images using the MNIST dataset.

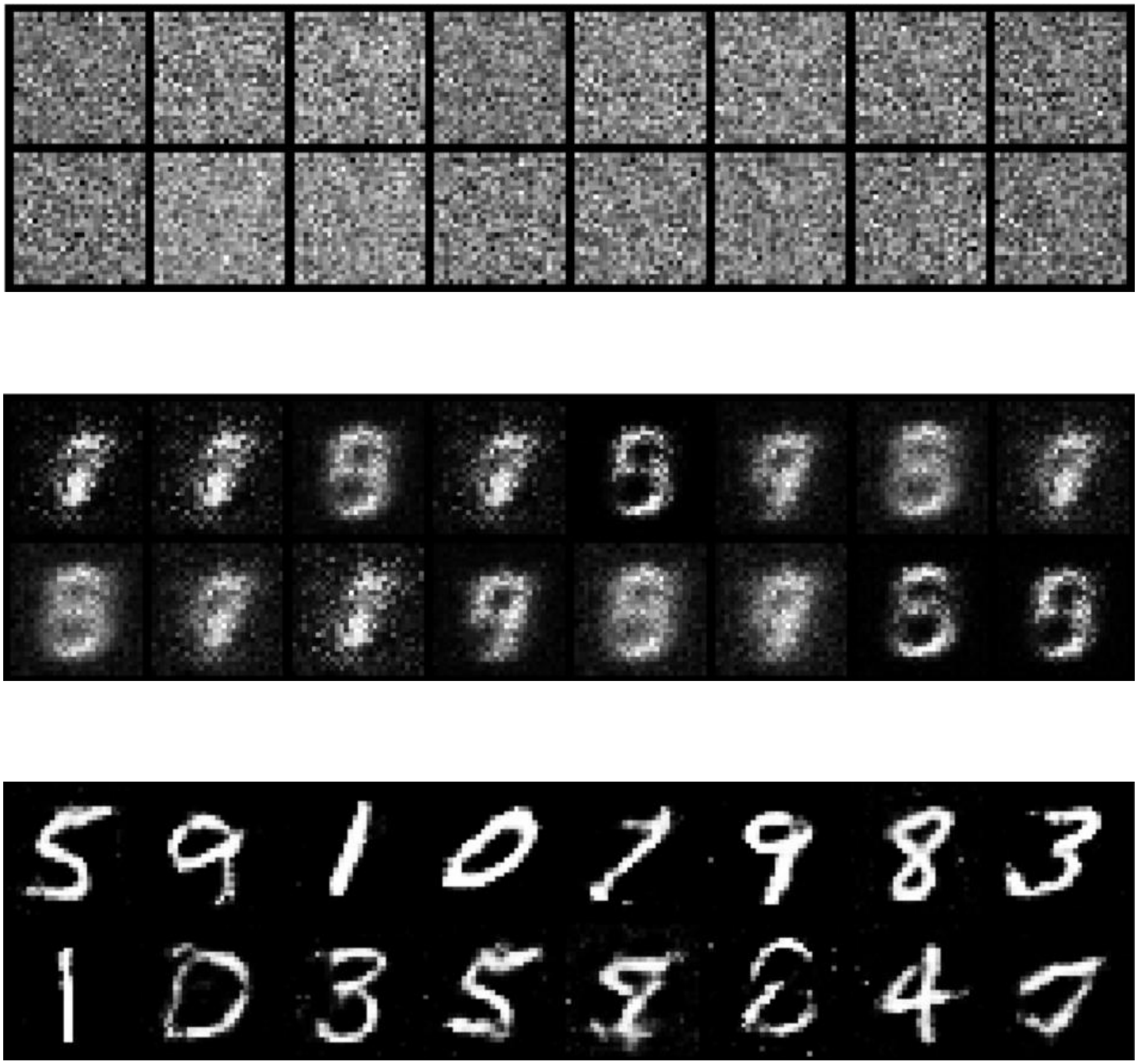


Figure 43: *Training of GANs on MNIST (Mosquera, D. G. (2018, February 1). GANs from Scratch 1: A deep introduction. With code in PyTorch and TensorFlow. Retrieved October 6, 2022, from https://medium.com/ai-society/gans-from-scratch-1-a-deep-introduction-with-code-in-pytorch-and-tensorflow-cb03cdcdba0f)*

Apart from the basic GAN model, various models can generate images that are different from the original image, such as StyleGAN - which is used to produce larger images of higher quality by progressive training, CycleGAN- which is used to alter the images, such as generating an image with a winter landscape by using a summer landscape image as the input, changing the emotion of the input image from sad to happy, etc.

Despite GANs being a trendsetter in this decade, GAN architecture has a lot of problems. Firstly, a Convolutional Neural Network architecture itself is a complex structure, and GANs use two of them for processing. GANs are prone to instability due to vanishing gradients [13]. This is caused by the Discriminator, which is near perfect, diminishes the loss function, and provides little feedback to the Generator. Hence, the Generator would not learn properly or would take a long time to learn from the feedback given by the Discriminator. Another problem that could arise is Mode Collapse [13]. This happens when the generator mixes and maps diverse inputs to the same output. Hence, the generator is not able to produce a diverse set of outputs.

In our research, we utilize the basic GAN architecture and try to use a deep learning model such as ResNet in place of a simple Discriminator model, to check how the model improves or decreases the efficiency of the GAN architecture. To begin with, we created a basic architecture of GANs. The generator component consists of a 7-layer CNN architecture and the Discriminator consists of a 5-layer CNN architecture. We used the same FER 2013 dataset for this research for the sake of simplicity. One important thing to note in these architectures is that they do not have a Pooling layer. This is because a pooling layer reduces the input dimension before sending them to the next layer. Hence, it makes sense to use them when we are extracting key features from the image to perform tasks like classification. But in this case, we are trying to reproduce the exact image that we provide as input, so we would not be needing a Pooling layer and would instead stick to strided-convolutions. Strided-convolutions have an additional advantage over pooling, as convolutions contribute to the learning process of the neural network.

A. Adversarial Networks and The Math Behind GANs-

While the generator generates the images, the task of the discriminator in the GAN architecture is to classify the newly created images as real or fake, which is simply a Binary classification problem, and the discriminator's loss function is Binary Cross Entropy. The discriminator is trained to bloat the chance of any real data being predicted as a real image and minimize the chance of any fake image being predicted as the original image. And the generator's primary task is to deceive its adversary, hence its weights are adjusted for magnifying the chance that the fake data are predicted as real. The Discriminator tries to reduce the loss function, while the other attempts to maximize it during training. Essentially, the two models play a minimax game.

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

Figure 44: GANs value function V(G, D)

B. Initial Training on Basic Architecture-

We initially trained the basic GAN architecture on the FER 2013 dataset. Once the dataset was loaded, the model considered a 6x6 grid containing 36 images from the dataset and trained on them. We developed two CNN models for the generator and discriminator, respectively, and trained the model for 10,000 iterations initially.

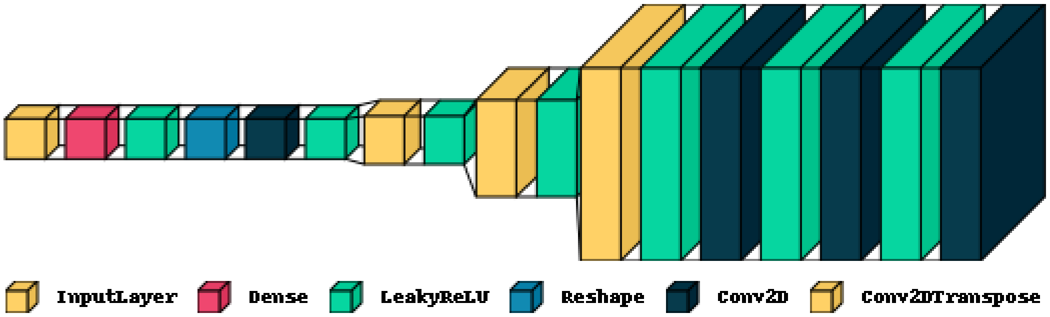


Figure 45: Generator of the Base Model

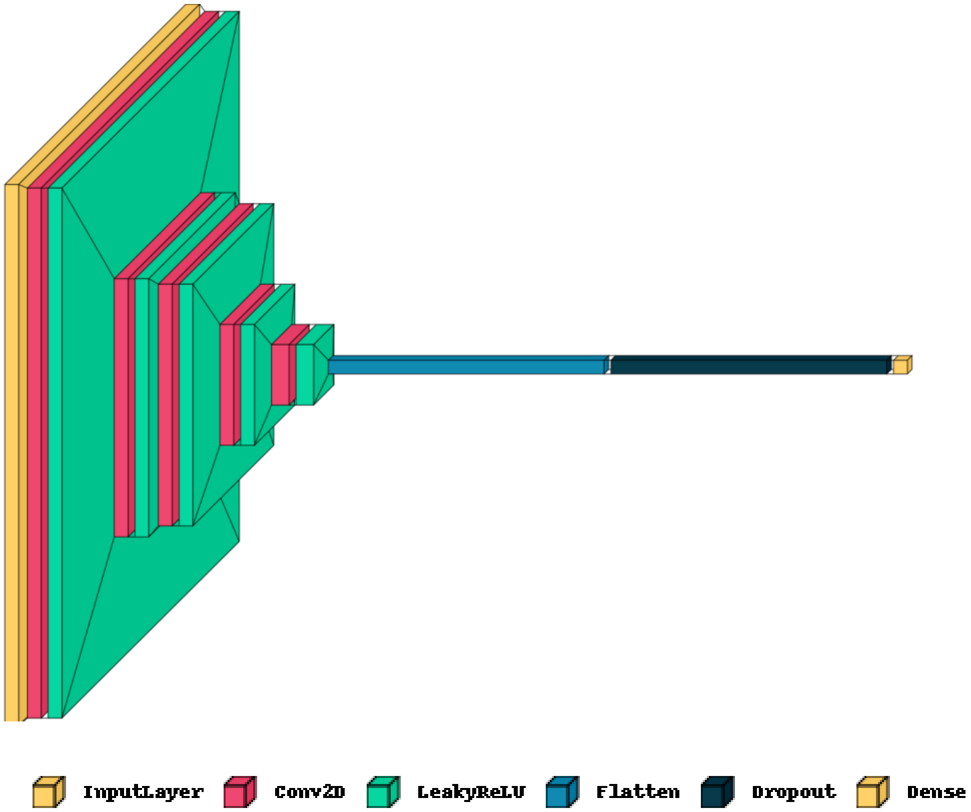


Figure 46: Discriminator of the Base Model

One key aspect to note here is that we are not using epochs. An epoch is when an entire dataset goes through the model, where many iterations on the dataset consist of a single epoch. For example, if we have a dataset of 6,000 images in batches of 500, then we take 6 iterations to complete an epoch [14]. Initially, the model trained for 10,000 iterations, but we later increased the number of epochs to 15,000 as the model did not converge at 10,000 iterations.

Upon training for 15,000 iterations, the model started to converge and produced good results. Training the model for an additional 10,000 to 15,000 iterations could train the generator well and the model could produce images similar to the original images.

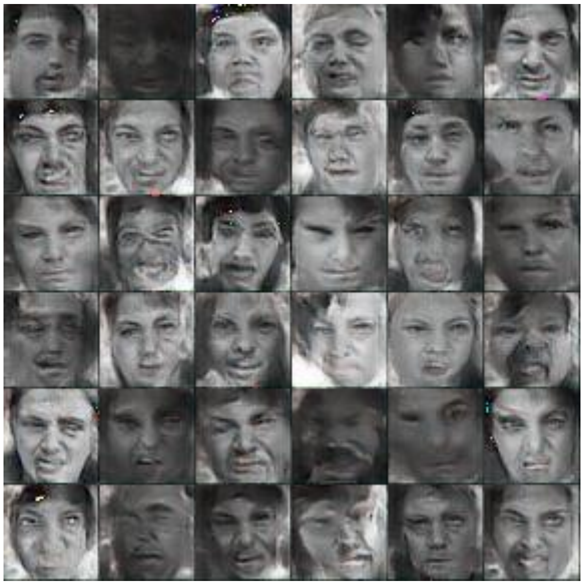


Figure 47: Results from GAN Base Model

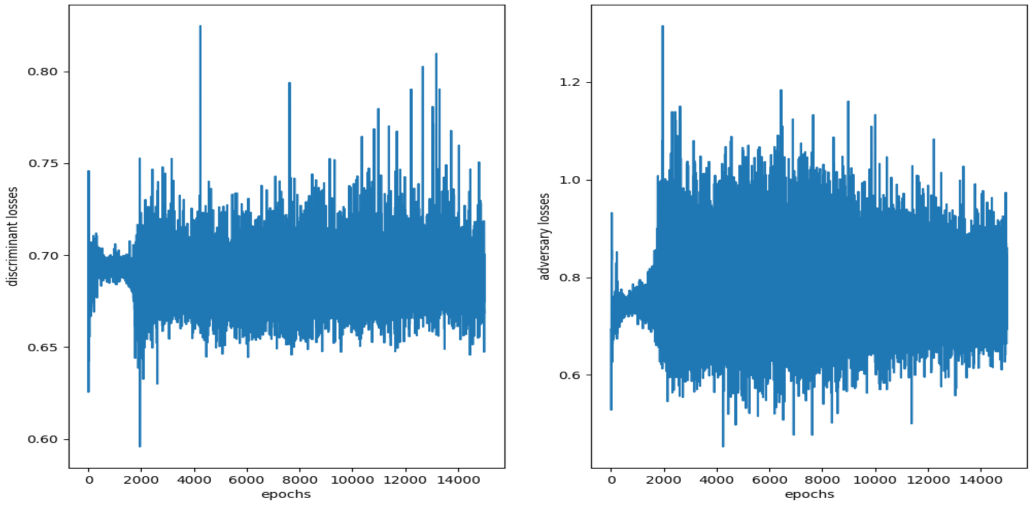
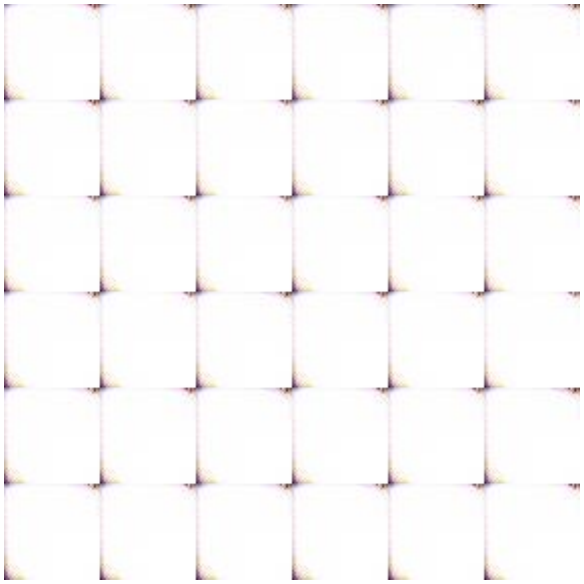


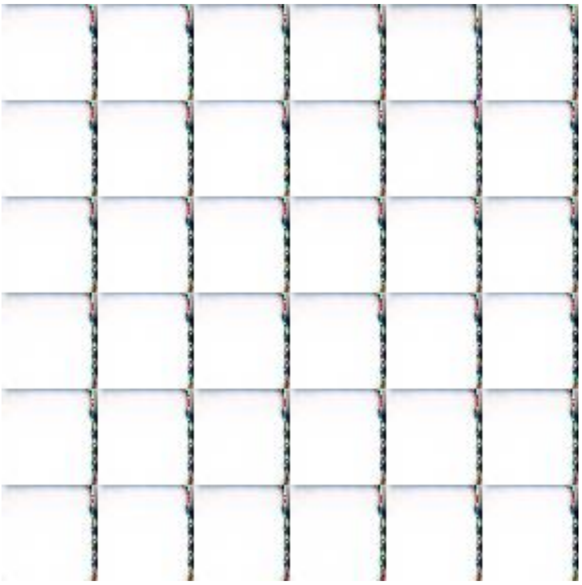
Figure 48: Base Model Loss Graph

We can see from the loss graph that the adversary loss steadily decreases on average through the entire training period.

C. Training using Resnet-50 Discriminator-

Keeping the same generator, we replaced the Discriminator with a Resnet-50 architecture. Due to a lack of enough computational resources, we were unable to train the model for over 10,000 iterations. We used a batch size of 64 images and trained on a single emotion of the FER 2013 dataset. Note that, GANs are not an emotion classification problem, so the use of different emotions does not contribute to the training of the model. As a result, the model was not able to converge and the images that were generated were nowhere like the original images.





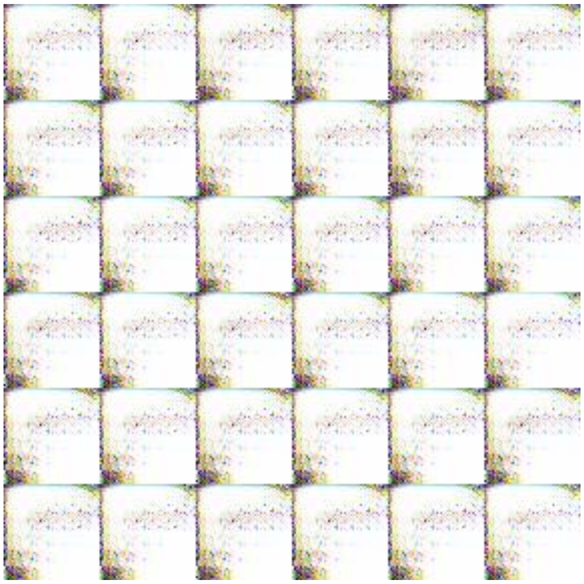


Figure 49: Results from GAN with Resnet-50 Discriminator

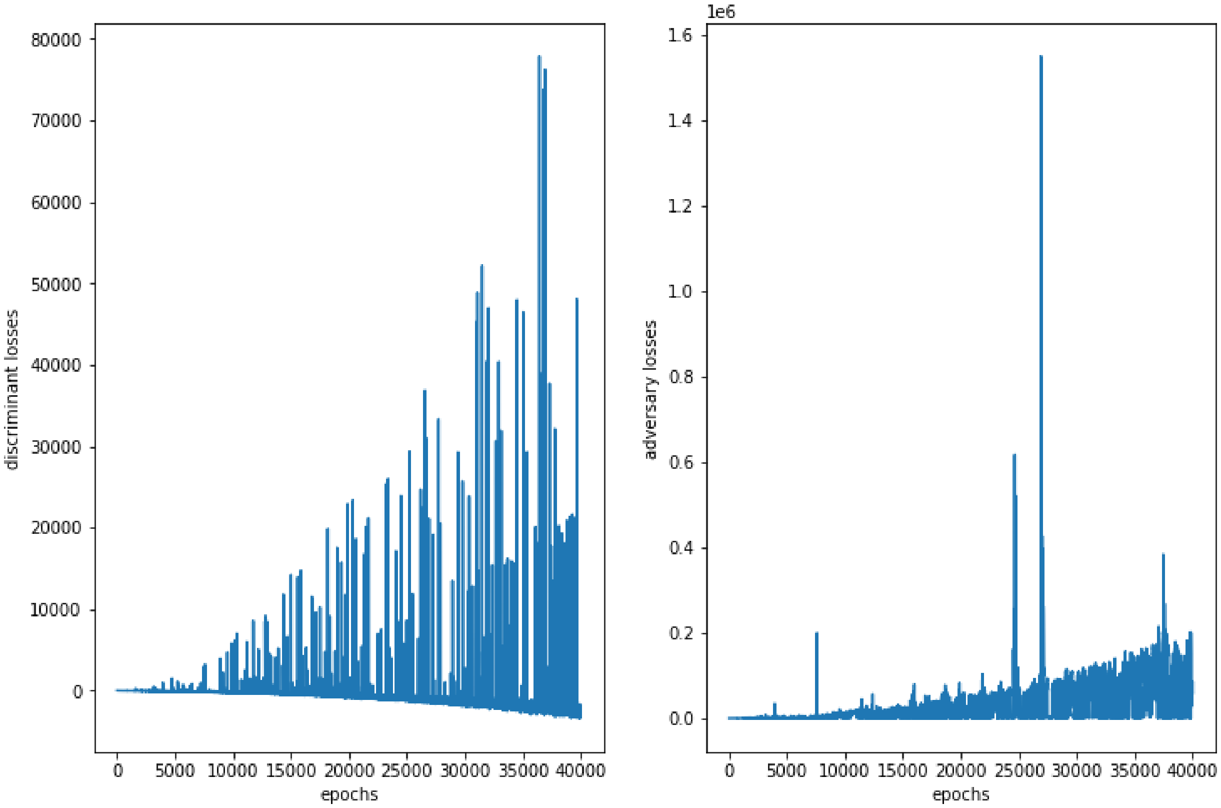


Figure 50: Results from GAN with Resnet-50 Discriminator

From the above results, we were able to identify that the network produced a high amount of loss for both the discriminator and the adversary. We attempted to improve this situation by making use of Spectral Normalization [52], which is a technique that helps to stabilize the training of the discriminator. We initially tested it out on the base model but the base model produced a higher loss rate when compared to the initial case. Hence, we decided not to make use of Spectral Normalization for Resnet-50.

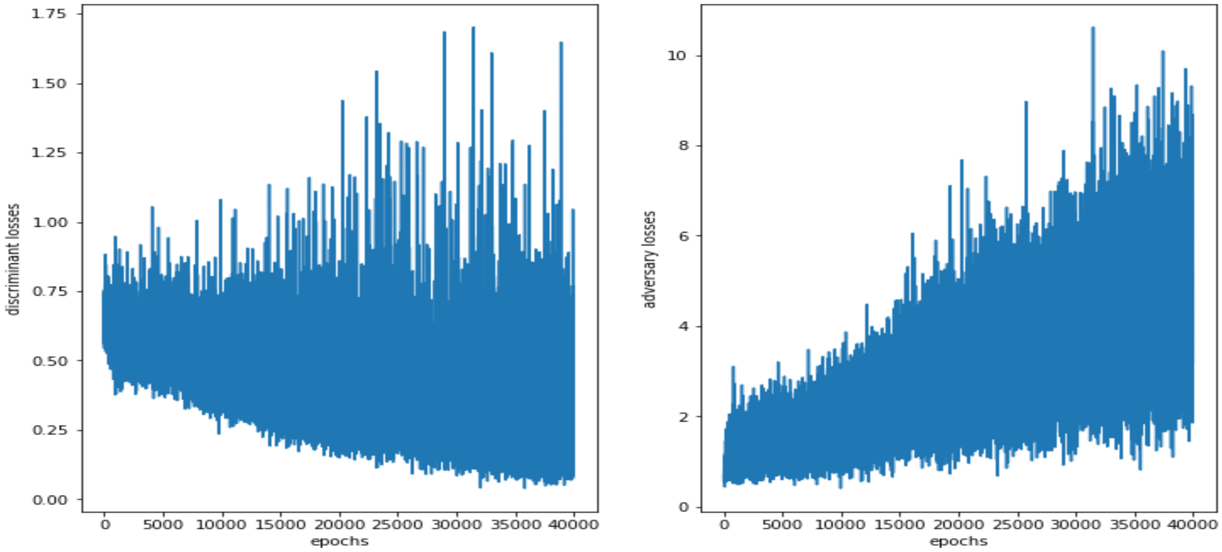


Figure 51: Base Model with Spectral Normalization Loss Graph

We believe the problem might have arisen due to these reasons:

1. The model is not balanced. We have a small Generator and a huge Discriminator. Using a balanced architecture can solve the problem.
2. The Resnet-50 architecture that we used was pre-trained on an imagenet dataset. The imagenet dataset consists of a lot of general images, such as images of animals, and

vehicles. Using a Resnet-50 model that was pre-trained on FER datasets such as FER2013+, CelebA may improve the results.

3. We were unable to train the model beyond a point due to a lack of computational power.

D. Using CycleGAN as the Generator-

To check whether a balanced model could solve this problem, we used a Conditional CycleGAN architecture from [CycleGAN paper]. CycleGAN transfers the patterns and styles from one image to another [15]. While other types of GANs focus on re-constructing the images, CycleGAN focuses on transferring the styles from the original data to the target. Similar to the basic GANs model, we use backpropagation to mutate the generator, to fix the shortcomings identified by the discriminator.

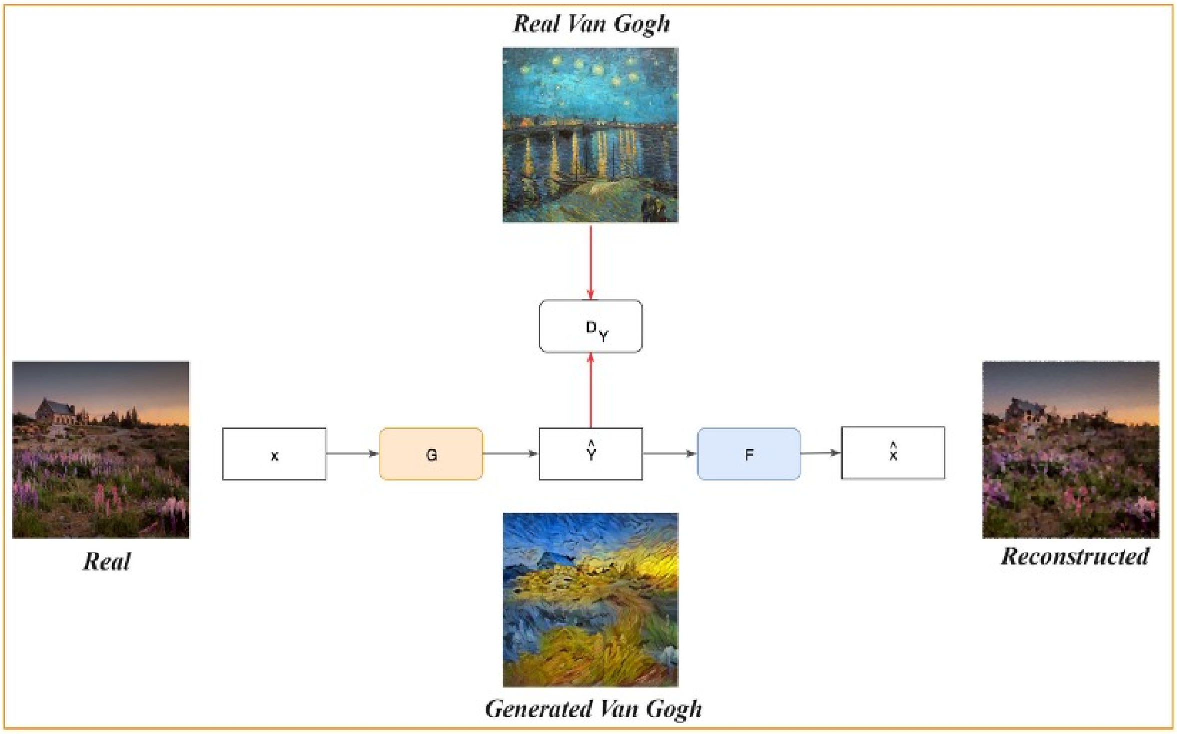


Figure 52: CycleGANs (Hui J. (2018, June 14). GAN - CycleGAN (Playing magic with pictures). Retrieved October 10, 2022, from https://jonathan-hui.medium.com/gan-cyclegan-6a50e7600d7)

Conditional CycleGANs are a variation of the CycleGANs where we “condition” the model to learn a depiction between a domain and a set of features, in this case, emotions. Hence, an image with emotion x gets changed to emotion y.

For this purpose, the Generator is split into two - an encoder and a decoder. The encoder is responsible for encoding the image into its hidden representation, and the decoder is responsible for performing the image-to-image translation. In other words, the decoder simply applies the emotion to the latent image. This kind of architecture for the Generator is known as a U-NET [16] [17] generator.

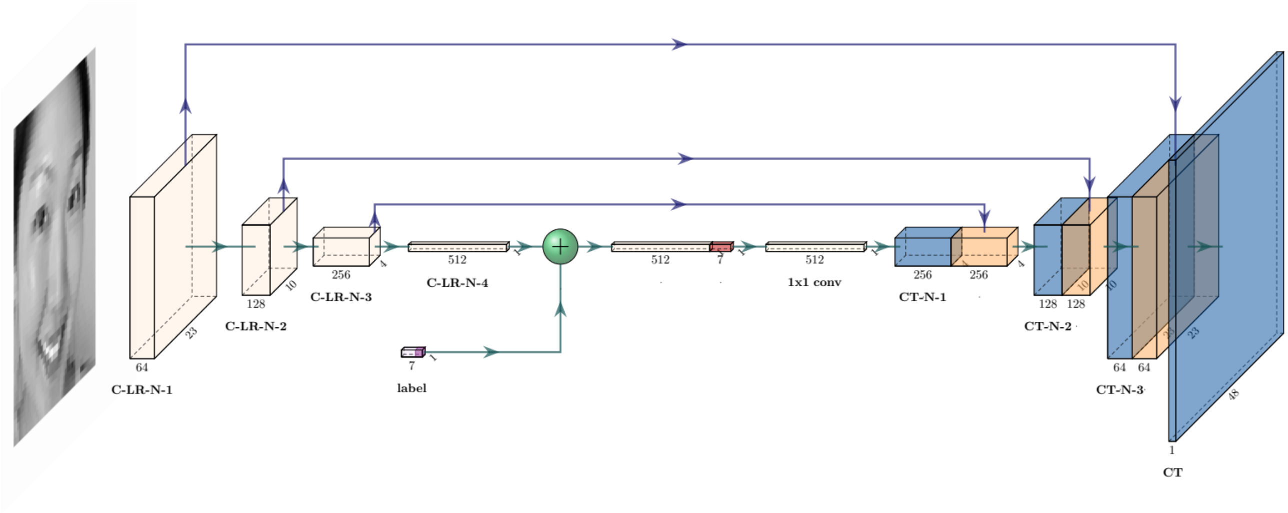


Figure 53: U-NET Generator (Tesei G. (2019, June 4).Generating Realistic Facial Expressions through Conditional Cycle-Consistent Generative Adversarial Networks (CCycleGAN). Retrieved October 10, 2022, from Open Review. https://openreview.net/forum?id=HJg6j3-oeB)

Another key realization of this CycleGAN model is that the discriminator is modeled as a two-task learning function to overcome the difficulties faced during back-propagation. To achieve this, a part of the network is common between the two tasks, with sigmoid and softmax being used as the activation functions.

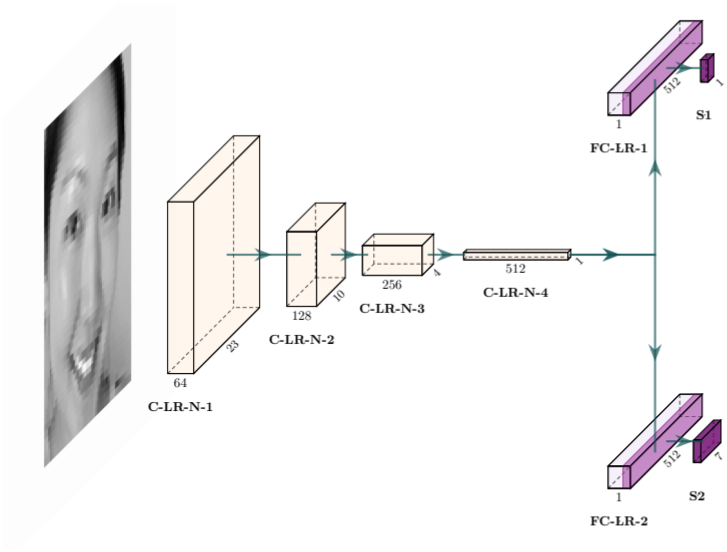
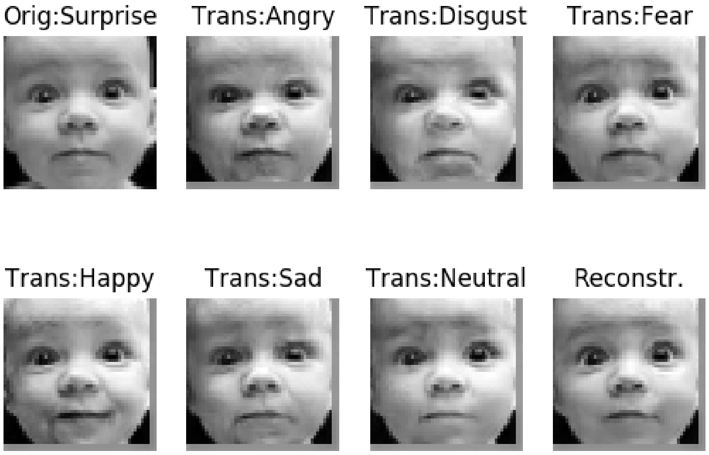


Figure 54: Discriminator (Tesei G. (2019, June 4).Generating Realistic Facial Expressions through Conditional Cycle-Consistent Generative Adversarial Networks (CCycleGAN). Retrieved October 10, 2022, from Open Review. https://openreview.net/forum?id=HJg6j3-oeB)

Upon training the model for 170 epochs, with a batch size of 64 and an Adam learning rate of 0.002, these were the results obtained by the author.



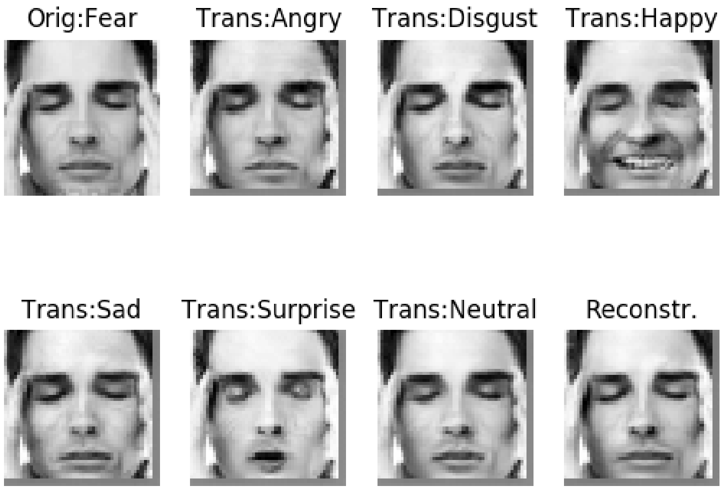


Figure 55: CycleGANs (Tesei G. (2019, November 21). Conditional Cycle-Consistent Generative Adversarial Networks. Retrieved October 10, 2022, from https://github.com/gtesei/ccyclegan)

When we started using the CycleGAN model with a Resnet-50 discriminator, we faced a lot of compatibility issues with the CNN models. As much as we attempted to fix the problem, we could not get a running model in the due time. Any changes to the discriminator would mean that we would be editing the Resnet-50 itself, which is not what we are trying to accomplish here. Hence, we believe that creating a deep CNN model which would be perfectly compatible with this CycleGAN generator could help accomplish our goal.

XI. CONCLUSION AND FUTURE WORK

Over the course of this research, we looked at the use of Image data augmentation and Image Sharpening techniques to preprocess the input dataset and boost the performance of the CNN models during training. These techniques have improved the performance when compared to the previously published models by Wang et al. [29] and Vepuri [51]. When compared to previous studies, we achieved higher test accuracies when using the pre-trained models, without making use of extra data and without freezing the entire network. We were able to create a GAN architecture for working on FER 2013 dataset and we made use of ResNet – 50 as the discriminator. Then we attempted to use the CCycleGAN model as the Generator to create a balance.

Future work on this research includes improving the GAN architecture by trying to use ResNet-50 models pre-trained on Facial data, improving the architecture by making it a bit more sophisticated, training the model for many epochs, and making use of a sophisticated generator when making use of Resnet-50 as the discriminator. Concerning the data preprocessing, we can attempt to reduce the noise and can make use of auxiliary data to improve the performance of our model further. Better FER detection techniques can immensely help autistic children, can improve the performance of robots, and can ensure the safety of drivers by monitoring their attention while driving.

REFERENCES

[1] S. Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), 2017, pp. 1-6, doi: 10.1109/ICEngTechnol.2017.8308186.

[2] R. Xin, J. Zhang and Y. Shao, "Complex network classification with convolutional neural network," in Tsinghua Science and Technology, vol. 25, no. 4, pp. 447-457, Aug. 2020, doi: 10.26599/TST.2019.9010055.

[3] A. Bhandari (2020, August 16). Image Augmentation on the fly using Keras ImageDataGenearator. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/08/image-augmentation-on-the-fly-using-keras-imagedatagenerator/

[4] Histogram Equalization. UCI Interdisciplinary Computational and Applied Mathematics Program, 2010, pp. 1, 2.

[5] S. Sudhakar (2017, July 9). Histogram Equalization. Medium. https://towardsdatascience.com/histogram-equalization-5d1013626e64

[6] C. D. Gürkaynak and N. Arica, "A case study on transfer learning in convolutional neural networks," 2018 26th Signal Processing and Communications Applications Conference (SIU), 2018, pp. 1-4, doi: 10.1109/SIU.2018.8404642.

[7] D. Sarkar (2018, November 14). A comprehensive Hands-on Guide to Transfer Learning with Real-World Applications in Deep Learning. Medium. https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a

[8] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90

[9] J. Hu, L. Shen and G. Sun, "Squeeze-and-Excitation Networks," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7132-7141, doi: 10.1109/CVPR.2018.00745.

[10] H. Nie, "Face Expression Classification using Squeeze-Excitation based VGG16 Network," 2022 2nd International Conference on Consumer Electronics and Computer Engineering (ICCECE), 2022, pp. 482-485, doi: 10.1109/ICCECE54139.2022.9712817.

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

[12] Z. Liu, M. Tong, X. Liu, Z. Du and W. Chen, "Research on Extended Image Data Set Based on Deep Convolution Generative Adversarial Network," 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 2020, pp. 47-50, doi: 10.1109/ITNEC48623.2020.9085221.

[13] L. Gonog and Y. Zhou, "A Review: Generative Adversarial Networks," 2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2019, pp. 505-510, doi: 10.1109/ICIEA.2019.8833686.

[14] M. Wiatrak, S. V. Albrecht and A. Nystrom, “Stabilizing Generative Adversarial Networks: A Survey” 2020

[15] S. Sharma (2017, September 23). Epochs vs Batch Sizes vs Iterations. Medium. https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9

[16] J. Hui (2018, June 14). GAN – CycleGAN (Playing magic with pictures). Medium. https://jonathan-hui.medium.com/gan-cyclegan-6a50e7600d7

[17] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Lecture Notes in Computer Science, pages 234–241. Springer International Publishing, 2015. 1, 4.2

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[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] 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.

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

[26] 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.

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

[28] 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

[29] 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.

[30] 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

[31] 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

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

[33] 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.

[34] 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.

[35] 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),

[36] 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-keras446d7ff84d33#:%7E:text=The%20ResNet%2D50%20model%20consists,ov er%2023%20million%20trainable%20parameters.&text=Our%20ResNet%2 D50%20gets%20to,in%2025%20epochs%20of%20training.

[37] Tsang, S. (2019, April 22). Review: SqueezeNet (Image Classification) - Towards Data Science. Medium. https://towardsdatascience.com/reviewsqueezenet-image-classification-e7414825581

[38] 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

[39] 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-intensorflow-part-2-b77237256df0

[40] 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.

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

[42] 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.

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

[44] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio.: Generative Adversarial Networks. arXiv: 1406.2661, 2014.

[45] G. Tesei. (2019). Generating Realistic Facial Expressions through Conditional Cycle-Consistent Generative Adversarial Networks (CCycleGAN). (accessed

[46] H. Jeong, J. Yu and W. Lee, "Poster Abstract: A Semi-Supervised Approach for Network Intrusion Detection Using Generative Adversarial Networks," IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2021, pp. 1-2, doi: 10.1109/INFOCOMWKSHPS51825.2021.9484569.

[47] Prabhat, Nishant and D. Kumar Vishwakarma, "Comparative Analysis of Deep Convolutional Generative Adversarial Network and Conditional Generative Adversarial Network using Hand Written Digits," *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS),* 2020, pp. 1072-1075, doi: 10.1109/ICICCS48265.2020.9121178.

[48] 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

[49] 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.

[50] 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.

[51] Vepuri, Ksheeraj Sai, “Improving Facial Emotion Recognition with Image processing and Deep Learning” (2021). *Master's Projects*.1030. doi: https://doi.org/10.31979/etd.3wrz-53ee https://scholarworks.sjsu.edu/etd_projects/1030

[52] Y. Horiuchi, S. Iizuka, E. Simo-Serra and H. Ishikawa, "Spectral Normalization and Relativistic Adversarial Training for Conditional Pose Generation with Self-Attention," 2019 16th International Conference on Machine Vision Applications (MVA), 2019, pp. 1-5, doi: 10.23919/MVA.2019.8758013.

[53] Seb. (2022, February 1). An Introduction to Residual Skip Connections and ResNets. Programmathically. https://programmathically.com/an-introduction-to-residual-skip-connections-and-resnets/

[54] M. U. Hassan. (2019, January 23). ResNet (34, 50, 101): Residual CNNs for Image Classification Tasks. Neurohive. https://neurohive.io/en/popular-networks/resnet/