Face Emotion Recognition Using Fusion Models

By

S. M. Atiqur Rahman (IT-16052)
Session: 2015-2016

Kawshik Mahato (IT-15057)
Session: 2014-2015/15-16

Supervised by
Dr. Sajjad Waheed
Professor

A thesis is submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science (Engg.) in Information and Communication Technology

Department of Information and Communication Technology
MAWLANA BHASHANI SCIENCE AND TECHNOLOGY UNIVERSITY
Santosh, Tangail-1902, Bangladesh
November, 2021

DECLARATION

This project focuses on the **"Face Emotion Recognition Using Fusion Models"** has been carried out by S. M. Atiqur Rahman and Kawshik Mahato in the Department of Information and Communication Technology, Mawlana Bhashani Science and Technology University, Santosh, Tangail-1902, Bangladesh. We evidence that the thesis work or any part of this work has not submitted anywhere for the award of any degree or diploma. The information have allowed for this document accurate and valid to best of our cognition.

S. M. Atiqur Rahman
Department of ICT, MBSTU
Santosh,Tangail-1902, Dhaka, Bangladesh

.....
Signature of Candidate

Kawshik Mahato
Department of ICT, MBSTU
Santosh,Tangail-1902, Dhaka, Bangladesh

.....
Signature of Candidate

Dr. Sajjad Waheed
Professor
Department of ICT, MBSTU
Santosh,Tangail-1902, Dhaka, Bangladesh

.....
Signature of Supervisor

APPROVAL

The thesis titled **"Face Emotion Recognition Using Fusion Models"** submitted by S. M. Atiqur Rahman (IT-16052) and Kawshik Mahato (IT-15057) has been found as satisfactory and accepted as fulfillment of the requirements for the degree of Bachelor of Science (Engg.) in department of Information and Communication Technology, Mawlana Bhashani Science and Technology University, Tangail -1902, Bangladesh.

Dr. Muhammad Shahin Uddin
Professor
Department of ICT, MBSTU
Santosh,Tangail-1902, Dhaka, Bangladesh

.....
Signature of Chairman

Dr. Sajjad Waheed
Professor
Department of ICT, MBSTU
Santosh,Tangail-1902, Dhaka, Bangladesh

.....
Signature of Member
(Internal)

Mr. S. M. Shamim
Lecturer
Department of ICT, MBSTU
Santosh,Tangail-1902, Dhaka, Bangladesh

.....
Signature of Member
(Internal)

Dr. Rafiqul Islam
Professor
Department of CSE, DUET
Gazipur, Bangladesh

.....
Signature of Member
(External)

Acknowledgements

All praise and thanks should go to the Almighty Allah, who gave us the determination, talent and intelligence we needed to finish and turn in our thesis work well and without any major problems. Alhamdulillah, we'd like to say thank you to our honorable supervisor, Dr. Sajjad Waheed, Professor, Department of ICT, who helped us finishing the project at different points and gave us valuable guidance, good suggestions and advice, constant encouragement and trust over the time it took to write this report in a very short time. We will always remember how clear, quick, wise and straightforward he was.

Finally, we would like to acknowledge all other honorable teachers and friends who have encouraged us, helped us and provided mental support over the course of completing our project work.

Abstract

Human psychology suggests that seeing a person's facial expressions is the most effective method for determining how they are feeling. It is now possible, with the help of computer vision and Facial Expression Recognition to analyze non-verbal human behavior as well as the interaction between humans and computers. The field of human-computer interaction, which includes clinical treatment and behavioral description, places a substantial emphasis on FER. Due to the substantial intra-class variability, it continues to be a significant difficulty for researchers, even with the advancement of technology. Solving FER problem hand-crafted features such as SIFT, HOG and LBP have historically been used. When the conditions are carefully managed, it performs quite well. However, the processes involved are quite complicated and putting them into practice is a significant obstacle. While employing more difficult dataset that contain more image variants and incomplete faces, automatic FER does not perform very well. Pose variations, occlusion, identity bias and the heterogeneity of human faces are the primary challenges facial recognition technology faces when used in real life. In recent years, a number of different efforts had already been made to conquer these obstacles yet, it looks as though there is still a significant amount of room which needs to be improved. In this work, a strategy based on convolutional neural networks that would be able to focus on face to get better results was suggested. FER-2013 was used as the dataset and some of the automatic features that are available in Convolutional Neural Networks like ResNet18 and VGG19 were also used. These results were utilized to demonstrate that an Ensemble model consisting of ResNet18 and VGG19 can achieve higher accuracy. People's faces often show a mix of emotions and it is hoped that this work will help to understand this more.

Keywords: Facial Expression Recognition, Convolutional Neural Network, Facial Expression Dataset, Occlusion, Pose Variations, Identity Bias, Heterogeneity of human faces.

Contents

List of Figures

List of Tables

List of Abbreviations

FER	Face Emotion Recognition
SIFT	Scale Invariant Feature Transform
HOG	Histogram of Oriented Gradients
LBP	Local Binary Pattern
CNN	Convolutional Neural Network
ResNet	Residual Neural Network
VGG	Visual Geometry Group
ASD	Autism Spectrum Disorders
JAFFE	Japanese Female Facial Expression
ACN	Attentional Convolutional Network
NMF	Non-negative Matrix Factorization
GPU	Graphics Processing Unit
DSD	Dense Spare Dense
SVM	Support Vector Machine
AdaBoost	Adaptive Boosting
CK+	Extended Cohn-Kanade
FERG	Facial Expression Research Group
LFW	Labeled Faces in the Wild
ReLu	Rectified Linear Unit
SGD	Stochastic Gradient Descent
API	Application Programming Interface
VSI	Visible Light Intensity

Chapter 1

Introduction

1.1 Overview

Facial expression is one of the most powerful, natural and universal signals for human beings to convey their emotional states and intentions. Emotions are an inevitable portion of any inter-personal communication. They can be expressed in numerous ways that might not be visible to the human eye. Therefore, the proper instruments are required to identify these feelings. Facial emotion recognition is crucial to human-computer interactions [1] and has a wide range of uses including in the sectors of public security, online gaming, distance learning and assistant medicine [2–4], diagnostics for ASD in children [5] and urban sound perception [6].

Technology that uses computer hardware as a medium to realize human-computer interaction is known as human-computer interaction technology. Artificial intelligence and pattern recognition have advanced quickly in recent years in the realm of human-computer interface technology [7], [8]. The main focus of this work is the construction of FER using the automatic features of a Convolutional Neural Network.

The use of artificial intelligence have made it possible for humans and computers to communicate in increasingly natural ways in recent years. This trend is expected to continue. As a result, giving the study of technology that can recognize facial expressions a strong push would be of significant benefit to the growth of both the society and the nation. The relationship between humans and computers is referred to as "human-computer interaction". The FER is an essential component of the human-computer interface. The FER is a technology that uses a computer as a helper and combines it with certain algorithms to figure out what

a person is really feeling by looking at their face. It categorizes the aspects of human face emotions according to human emotional expressions [9, 10]. These prototypical facial expressions are anger, disgust, fear, happy, sad and surprise. Hate was subsequently added as one of the basic emotions [11]. Recently, advanced research on neuroscience and psychology argued that the model of six basic emotions is culture-specific and not universal [12].

These days, it is employed in the health care industry. Feelings are essential to understand a person's actions and mental condition since emotions provide a window into their world. One of the most important aspects that contributes significantly to improve levels of personal satisfaction is a healthy emotional state. On the other hand, having poor social or psychological well-being might be the result of having poor emotional health. The ability to recognize or identify a patient's feelings through the utilization of online health care data has the potential to provide incredibly beneficial and important insight into the emotional states of the patient. For example, in the process of generating more accurate medications for patients and in the practice of providing treatment to autistic children based on the psychological changes they display [13].

Emotions are by far crucial in the education field, as their impact almost rivals that of the environment or the language being in use to transmit the information. In fact, they have an essential role in cognitive processes responsible for learning and assimilating new information [14]. FER technology uses different emotions and eye tracking to inform of students attention level and assist teachers in adjusting content or teaching methods and detecting learning disabilities. FER systems are now being used as innovative tool for improving students performance and learning approaches specially in distance education which is also known distance learning.

Facial Expression Recognition plays a vital role in intelligent security systems for real-time surveillance. FER is a form of biometric authentication that focuses on recognizing a person's face based on one or more physical or behavioral traits and how they look when they are feeling something. The system records a person's emotions and facial expressions with a camera and sends the footage to be preprocessed before it can be used to figure out if the emotion or expression recorded was meant to hurt someone [15].

These days, it's also applied in the field of improving relations with the e-commerce consumers [16], human social and physiologiacl interacion detection [17].

As the amount of communication between people and computers has grown, so has the need to use FER a lot in order to keep up with the fast growth of digitalization. In the next few years, FER will be used in every part of our daily life, especially in public safety, interactive gaming, digital advertising, distance education and medical assistants. [18].

The use of FER technology in real life requires a significant amount of time [19]. Only photographs that had been shot in controlled conditions are suitable for use with traditional FER modeling techniques. They have the same face position and the lighting is the same. However, the accuracy suffers when dealing with wild photos [20]. These photographs are either drawn at random from various social media platforms or taken from shots that people take on a consistent basis in everyday life. These pictures do not have the same face position, nor do they have the same illumination. The dataset that is based on such samples becomes rather complex as a result of the vast number of variants and at times, partial faces that are contained within such images. As a consequence, the models that are utilized begin to lose their accuracy and produce incorrect classifications based on the input.

Because of this, it has become very important to build models that can do tasks quickly while still being very accurate [21]. Right now, researchers are trying to build such a FER system.

1.2 Motivation

Convolutional neural networks (CNN) architecture had been employed for classification in a few important works that had been done in this field of technology. However none of those focused only on the automatic aspects of the system. CNN and image edge computing had been used successfully by several researchers with numerous datasets [22]. There had been reports of people working with the Attentional Convolutional Network (ACN) on multiple datasets [23].

Additional elements made the system far more complex and it was extremely difficult to apply when it comes to finding solutions to problems that occur in real life. The head position, lighting and partial face depicted in real-life difficulties might vary significantly from situation to situation. This dilemma had given motivation to run an experiment with a CNN that uses exclusively automated features. Some researches had been done using a single dataset [24], but most of them had less number of emotion categories or less data samples. This

dilemma had inspired to work with a dataset that contains maximum emotion categories and large number of data samples. In this work, FER-2013 dataset had been used, because it is a large data set that includes 7 different types of emotional categories [25], [26].

1.3 Problem Definition

Existing models of face emotion recognition have a great deal of room for improvement. The primary issues with FER models were:

- Overfitting and expression unrelated variations. Overfitting arises when training dataset did not include enough information. The majority of the datasets that were employed in previously developed FER models were of a very limited size. However, for deep learning approaches, a data set of considerable size was necessary. Expression Variations unrelated to one another could happen due to occlusion, as well as head position variation. When dealing with a real-time scenario, occlusion and head position variation are both required.

- The vast majority of the samples used in datasets had been collected from laboratories; however, due to the need for additional data and the construction of more intricate datasets, modern-day samples were procured at random from a variety of online social media platforms. In other cases, the primary information-gathering components of the FER pictures, such as the mouth and eyes, are either completely or partially obscured by the covering (e.g. wearing sunglasses, optical glasses or covering the face with hair, mask). As a result wrong information had been extracted.

- The accuracy level experiences a significant drop if random photos were processed using FER models. Such photos were now being collected at random from the internet (such as Social Media and TV Shows) and such photos feature a wide variety of head poses, types of human features and partial and full faces, as well as lighting variations. When dealing with something of this nature the system gets more complicated than it needs to be.

1.4 Objectives

The main objective of this research was to build a Face Emotion Recognition system using only automatic features of CNN. A large dataset with maximum emo-

tion categories was used and no added samples were used to increase the accuracy of the system.

1.5 Outline of the book

The Literature review of Face Emotion Recognition has been presented in chapter 2. In chapter 3 the proposed methodology and the system architecture for our recognition model has been described. In the following chapter 4, discussion has been made about the dataset that was utilized for this work as well as the methodology behind the experiment. In chapter 5, discussion has been made about the outcomes that were achieved, how they were examined and comparison with the earlier researches. Chapter 6 is where the article has come to a close and brought up several potential avenues for additional improvement.

Chapter 2

Literature Review

2.1 Early Works

Within the realms of computer vision and machine learning, a variety of facial expression recognition systems had been investigated to discover how expression information might be encoded using facial representation. It was commonly believed that facial expressions were the most potent and natural signals that humans used to communicate their emotional states to one another. Early in the 20th century, Ekman and Friesen et al. [27] identified the fundamental six emotions based on research into many cultures. The Affect Model, which was based on these basic emotions, limited in its capacity to represent the complexity and nuance of our daily affective displays [28]. Other emotion description models, such as the facial action coding system [29] and the continuous mode, which used affect dimensions, considered to represent a wider range of emotions, but the Affect Model was limited in its ability to do so.

The categorical model that represented emotions was still the most common perspective for FER due to the pioneering investigations that it conducted in addition to the straightforward and instinctive definition of facial expressions that it proposed. The vast majority of classic FER methods rely on either manually constructed features or shallow learning techniques such as local binary patterns (LBP) used by Shan et al. [30], Zhao et al. [31] used LBP on three orthogonal planes (LBP-TOP), non-negative matrix factorization (NMF) was used by Zhi et al. [32] and Zhong et al. [33] used sparse learning. At the beginning of 2013, the method of FER started going through several transitions. The collection of reasonably sample training data from difficult real-world scenarios via emotion recognition competitions implicitly promotes the transition of FER from lab-controlled to in the wild settings [34].

In addition, due to the dramatically increased processing capabilities of chips (such as GPUs) and well-designed network architecture, researchers in a variety of fields had started shifting their focus to deep learning methods, which had achieved higher levels of accuracy and surpassed their previous results by a significant margin [35]. This shift to deep learning methods had enabled researchers to achieve better results [36]. In a similar vein, given the more effective training data of facial expressions, deep learning algorithms had increasingly been adopted to manage the tough elements for emotion recognition in the wild. This was due to the fact that facial expressions were more expressive than ever before [37].

2.2 Related Works

In the early stages of face emotion recognition, a number of studies had been published on various methods developed using handcrafted models with the assistance of controlled data sets. These methods had been used to create various models. This method was difficult to follow and yielded inaccurate results when applied to the resolution of problems that occur in real life. However, the amount of work that was performed based solely on automatic characteristics and an uncontrolled data set was not significant. Since 2013, numerous researchers from different countries all over the world had published significant works based on their research.

In 2019, Mariana et al. [38] developed a Face Emotion Recognition System that utilized CNN models in addition to some handcrafted models for the system's construction which referred using three CNN models which were VGG-face, VGG-f and VGG-13. To solve the problem of overfitting the dense-spare-dense method (DSD) was used. FER-2013, FER+ and Affectnet were used as data sets.

The FER 2013 data set contained 28,709 training images, 3,589 validation (public test) images and another 3589 (private) test images. All images were of 48-by-48 pixels with 7 classes of emotion [39]. The FER+ data set contained 25045 training images, 3191 validation images and another 3137 test images with 7 classes of emotion [39]. The AffectNet data set contained 287651 training images and 4000 validation images, which are manually annotated. 7 way classifier contained 3500 validation images and 8 way classifier contained 4000 validation images [39].

This goal was to achieve the highest level of accuracy possible through the utilization of an ensemble model that was produced by combining the results of all

three CNN models, used both in local and global forms of learning respectively.

Global learning was used to find the vector of weights and the bias term that define the hyperplane which maximally separates the feature vectors of the training examples belonging to the two classes. To extend the linear SVM classifier to multi-class facial expression recognition problem. Local learning methods was used to locally adjust the performance of the training system to the properties of the training set, in each area of the input space.

In order to implement VGG-face and VGG-f, the input images were scaled to 224-by-224 pixels, while input of VGG-13 images were scaled to 64-by-64 pixels. In order to partition images into spatial pyramid representation, The following results were obtained:

Table 2.1: Ablation results on the FER 2013 [40], the FER+ [41] and the AffectNet [42] data sets. Results are reported with and without data augmentation (*).

Model	FER	FER (*)	FER+	FER+ (*)	AN 8-way	AN 8-way(*)	AN 7-way	AN 7-way(*)
Pre-trained VGG-face	65,65%	65.78%	81.54%	81.73%	49.28%	50.08%	54.14%	54.94%
Fine-tuned VGG-face	71.50%	72.11%	84.35%	84.79%	58.77%	58.93%	62.54%	62.66%
Fine-tuned VGG-f	69.38%	70.30%	85.72%	86.01%	55.85%	56.03%	60.40%	60.51%
VGG-13	66.31%	66.51%	84.38%	84.41%	40.50%	41.75%	44.60%	44.57%
Ensemble + Global SVM	73.34%	73.25%	86.68%	86.96%	59.20%	59.30%	63.20%	62.91%
Ensemble + Local SVM	74.92%	75.42%	87.76%	87.25%	59.45%	59.58%	62.94%	63.31%

N.B. "AN" denotes AffectNet dataset

From above table it was noted that the local SVM generally attains better performance than the global SVM, in all but one case, proving that the idea of using local learning was indeed useful. Overall, the results demonstrate that proposed method based on local SVM and a combination of deep and handcrafted features, achieves top performance on all three data sets: FER- 2013, FERC and AffectNet.

A method that is more straightforward than the conventional FER models was suggested by Hongli et al. [43]. Hongli had used the CNN model and image edge computing. This study began by standardizing the image and the convolution process was then used to extract the edge of each layer of the image. In order to maintain the edge structure information of the texture image, the information that was extracted from the edges was superimposed on each of the feature images. After that, the dimensional reduction of the extracted implicit features were processed by using the maximum pooling method. The maximum pooling method was utilized in order to cut down on the dimensions of the extracted implicit features, which in turn cut down on the amount of time needed to train the CNN model.

In the first step of the process, a Haar classifier was utilized to identify hu-

mans, followed by an integral graph method and the AdaBoost algorithm. After that, pictures of a particular dimensions were generated by using normalizing the initial images. A method called the Histogram Equalization method was used to equalize the gray levels of the images. In order to extract image edge information, eight different directions of the Kirsch edge operator were used. In the final step, a softmax classifier was utilized in order to classify and recognize the expression that was taken from the test sample image. A combined dataset of FER-2013 and LFW was used for this particular piece of research. Through this work researchers were able to achieve 88.56% success with fewer iterations, making the process 1.5 times faster overall.

Shervin et al. [44] in 2019 approched a deep learning method that was based on the Attentional Convolutional Network model. This model was able to zero in on significant aspects of the face and is capable of identifying the underlying emotion conveyed by facial expressions. Convolutional neural networks with fewer than 10 layers were used in this work, along with attention (which was trained from scratch). Due to the fact that FER systems only need to pay attention to the particular regions in order to get a sense of the underlying emotion, attention mechanism was added on the basis of this fact. Adam was the optimizer that was used in order to get the loss function optimized. As datasets, FER-2013, CK+, FERG and JAFFE were used.

The FER 2013 data set contained 28,709 training images, 3,589 validation (public test) images and another 3589 (private) test images. All images were of 48-by-48 pixels with 7 classes of emotion, the entire 28,709 images were used in the training set to train the model, validated on 3.5k validation images and reported the model accuracy on the 3,589 images in the test set [44]. The Extended Cohn-Kanade (CK+) dataset contained 593 video sequences from a total of 123 different subjects. Each video shows a facial shift from the neutral expression to a targeted peak expression, recorded at 30 FPS with a resolution of either 640-by-490 or 640-by-480 pixels. Out of these videos, 327 were labelled with one of seven expression classes. In this work 70% of the images were used as training, 10% for validation and 20% for testing form CK+ dataset [44].

For FERG dataset, 34k images were used for training, 14k for validation and 7k for testing. For each facial expression, randomly 1k images were selected for testing [44]. To use JAFFE 120 images were used for training, 23 images for validation and 70 images for test [44].

A visualization technique was also utilized, the sole purpose of which was to concentrate on the significant facial regions in order to identify a variety of

emotions based on the output of the classifier. The proposed model yielded the following results:

Table 2.2: Accuracy of the proposed method [44].

Dataset	Accuracy Rate
FER-2013	70.02%
FERG	99.3%
JAFFE	92.8%
CK+	98%

Yousif et al. [45] suggested using a single network as a model in order to achieve higher levels of precision. In this study numerous optimization strategies were utilized in addition to the VGGNet architecture for the CNN network. For the purpose of this study, only the FER-2013 dataset was used without any additional training data.

There was a considerable amount of data enhancement done in the training section. The process of augmenting an image involved rescaling it by up to twenty percent of its original size, shifting it horizontally and vertically by up to twenty percent and rotating it by up to ten degrees. Each2 of the methods was implemented in a random fashion and with a chance of fifty percent. After this, each of the images in the data set was cropped to a size of 40-by-40, and then with a probability of 50%, random portions of each of the crops were removed. The final step in the process was to normalize each crop by dividing each pixel by 255.

All of the images were experimented through 300 iterations in an effort to minimize the cross-entropy loss. The conclusion drawn from this model was 73.28%.

Chapter 3
Methodology and System Architecture

3.1 Convolutional Neural Network

Convolutional neural networks, often known as CNNs, are a type of neural network that can have one or more convolutional layers. CNNs is a class of Artifical Neural Network (ANN) most commonly applied to analyze visual imagery [46]. CNNs are also known as Shift Invariant or Space Invariant Artificial Neural Networks (SIANN), based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation-equivariant responses known as feature maps [47]. Counter-intuitively, most convolutional neural networks are not invariant to translation, due to the down sampling operation applied to the input [48].

CNNs networks are mostly utilized for the processing of images, as well as for classification [49], segmentation and other forms of data that are auto-correlated. CNNs are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "full connectivity" of these networks make them prone to overfitting data. Typical ways of regularization, or preventing overfitting, include: penalizing parameters during training (such as weight decay) or trimming connectivity (skipped connections, dropout, etc.) CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble patterns of increasing complexity using smaller and simpler patterns embossed in their filters. Therefore, on a scale of connectivity and complexity, CNNs are on the lower extreme [50].

Utilizing it helps to make the most of the two-dimensional structural photographs. In general, a CNN model is made up of some convolutional layers (in addition to pooling layers), which is actually a hierarchical feed-forward model,

followed by some fully connected layers in which all of the neurons are connected to each other. The final layer of a CNN model is also fully connected layer.

Convolutional networks may include local or global pooling layers along with traditional convolutional layers. Pooling layers reduce the dimensions of data by combining the outputs of neuron clusters at one layer into a single neuron in the next layer. Local pooling combines small clusters, tiling sizes such as 2-by-2 are commonly used. Global pooling acts on all the neurons of the feature map [51]. There are two common types of pooling in popular use: max and average. Max pooling uses the maximum value of each local cluster of neurons in the feature map, [52] while average pooling takes the average value.

Fully connected layers connect every neuron in one layer to every neuron in another layer. It is the same as a traditional multilayer perceptron neural network (MLP). The flattened matrix goes through a fully connected layer to classify the images.

The input of the model is a two-dimensional image and the training of the neurons is determined by the values that are given. In the final layer, which is called the output layer, the output is subjected to processing before being sorted into several categories. The pooling layers offer an excellent means of absorbing form fluctuations when employed in this capacity. In addition to this, the CNN model has a reduced number of parameters in comparison to a fully connected architecture that has the same amount of hidden layers. It is also easy to train using a gradient-based approach, which helps to directly minimize error rates and optimize overall performance. Both of these benefits contribute to the ease with which it may be learned [53].

3.2 Proposed Methods

Since their first applications in a variety of subfields of artificial intelligence or machine learning, numerous methods had been created specifically for use in convolutional neural networks. These methods could be used to perform a variety of tasks. An in-depth study of a selection of the most crucial methods was the most effective way to begin navigating the vast array of accessible approaches. The activation function that had been used, the shape and depth of the convolutional layers, the presence of sampling layers, the regularization technique and other factors were the most important considerations. The type of neuron that had been used as activation function can be thought of as the most important factor. mainly focused on the depth of the convolutional network in order to

achieve higher accuracy. Because of this, in this work, the ResNet18 (18 layers deep) and VGG19 (19 layers deep) model had been used as CNN networks. then the output had been taken of these models and trained it in Ensemble model, which was the combined model of ResNet18 and VGG19. Through the utilization of Ensemble model, the accuracy of predictions were intended to be improved.

3.2.1 Activation Function

Two separate convolutional neural networks had been combined into the model that have been proposed. Resnet18 has seventeen convolutional layers, followed by one fully connected layer, whereas VGG19 has sixteen convolutional layers, followed by three fully connected dense layers. Rectified linear neurons were utilized for the activation procedure throughout the convolutional layer of ResNet18 [54]. When given the input x, the output of a rectified linear neuron will always be:

$$f(x) = max(0, x)$$

This function was also referred to as the ReLU function or the Rectified Linear Unit function. An easy way to begin learning about the ReLU function was to understand that it is a piece wise linear function that will output the input directly if it is positive and will output zero if the input is negative or zero [55].

$$f(x) = \begin{cases} 1; \text{ if } x=0 \\ 0; \text{ if } x \neq 0 \end{cases}$$

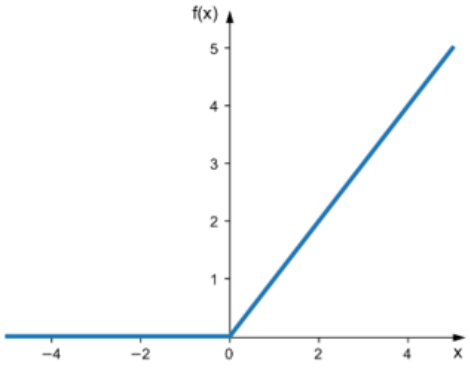

Figure 3.1: ReLu Function [55].

For vgg19 model Softmax function was used to normalize output of the network used. Soft max function, also known as Softargmax or normalized expo-

nential function is a generalization of the logistic function multiple dimensions. It had been used in multinomial logistic regression and often used as the last activation of a neural network to normalize the output of a network to a probability distribution over predicted output classes, based on Luce's choice axiom [56].

$$f_i(\vec{a}) = \frac{\epsilon^{a_i}}{\sum_k \epsilon^{a_k}}$$

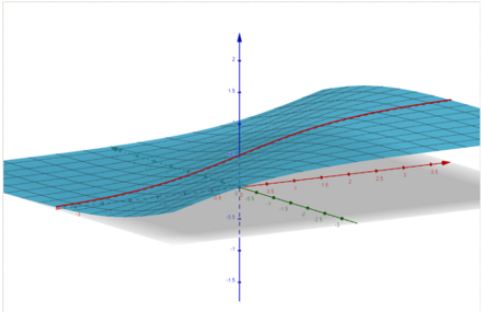

Figure 3.2: Soft-max function [57].

The soft-max algorithm is primarily a mathematical function that, when applied, transforms a vector of numbers into a vector of probabilities, with the probabilities of each value being proportional to the relative scale of each value in the vector [57].

3.2.2 Overfitting and Regularization

The issue of a neural network becoming too "Overfit" could be remedied through the implementation of a technique known as "Regularization."

In neural networks, one of the most significant issues is overfitting. This is especially true in contemporary networks, which typically have a very large number of parameters to work with. This occurs when the network begins to merely memorize the training set, rather than understanding characters well enough to generalize their behavior to the test set.

The dropout regularization had been utilized for this model. Dropout, a completely different strategy, in which the network itself had been directly modified. It entails deleting some of the hidden neurons in the network in a random and temporary manner, all the while maintaining the integrity of the neurons that were responsible for input and output [58].

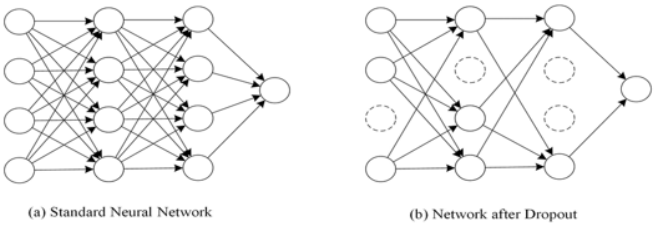

Figure 3.3: Dropout.

When dropout had been used in various sets of neurons, it was almost like training multiple neural networks at once. As a result, the dropout procedure was similar to taking the average of the results obtained from an extremely diverse range of networks. Because the various networks would each overfit in their own unique way, it was hoped that the overall effect of dropout will be to reduce the amount of overfitting.

3.2.3 Sub-Sampling

The images can be made to look smoother by using a technique called sub-sampling, which pools together multiple layers of samples. The features that were present in a region of the feature map that had been generated by a convolution layer are summed up by pooling layers.

Additionally, the dimensions of the feature maps could be shrunk with its help. As a result, the number of parameters that needed to be learnt and the amount of computation that had been carried out within the network were both reduced. Pooling layers were typically applied right after the convolutional layers in a neural network. The information contained in the output of the convolutional layer had been simplified when pooling layers were used. The maximum pooling and the average pooling methods both known as typical forms of pooling.

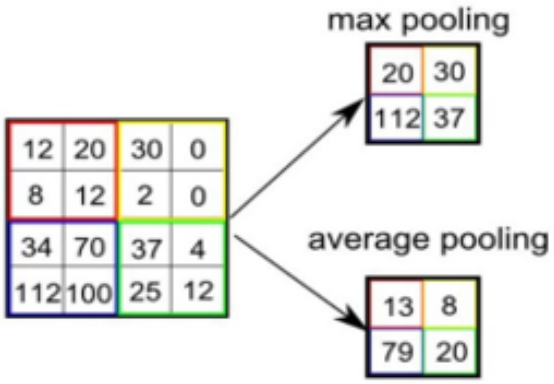

Figure 3.4: Max-pool and Avg-pool [59].

A max-pooling unit with a pool size of 2-by-2 simply outputs the maximum activation that was present in an input region that was 2-by-2 in size from the layer below it. As common knowledge, a convolutional layer would typically consist of multiple feature maps. Max-pooling had been performed independently on each of the feature maps [59]. A big benefit of using max-pooling was that, we got much fewer pooled features.

$$OneCube = k \times k \times n[l-1] + 1$$

Total Parameters=One Cube$\times Filter_{number} = (k \times k \times n \times [l-1] + 1) \times n[l]$

k is the kernel size, n[L] is the number of filters in layer L and n[L-1] is the number of filters in layer L-1 which is also the number of channels of the cube [60].

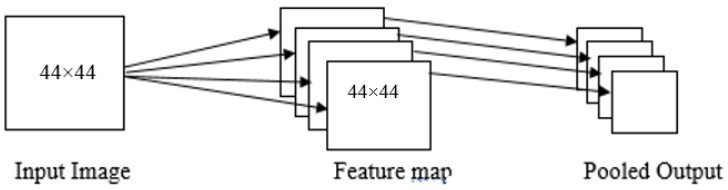

Figure 3.5: Mapping and Pooling.

Which helped to reduce the number of parameters needed in later layers. It also helped in extracting low-level features like edges, points etc. Max-pool selected the brightest pixels from the image which was very useful when the background of the image is dark.

Max pooling had been implemented in both the ResNet18 and VGG19 mod-

els. Due to the fact that certain records in the data set had a black background.

An average-pooling unit with 2-by-2 pool size simply outputs the average activation in a 2-by-2 input region from the previous layer. As is common knowledge, a convolutional layer will typically consist of several different feature maps. A separate application of average-pooling was made to each individual feature map [61].

The image had been smoothed out as a result of the benefits of using the average polling method and as a result, the sharp features might not be identified. Through the enforcement of correspondences between feature maps and categories, it had a convolutional structure that was more closely aligned with its natural state. Therefore, the feature maps could very easily be understood to be confidence maps for the categories.

Average-pooling method was only used in ResNet18 model.

3.3 System Architecture

The CNN models that had been utilized in this work were a combination of untrained ResNet18 and untrained VGG19. These models have 17 and 16 convolutional layers, respectively and a kernel size of 3-by-3 in most cases.

As activation function, Rectified Linear Unit (ReLU) had been used. For ResNet18, the first convolutional layer was followed by a 3-by-3 max-pooling layer and an average-pooling layer was used before the fully connected layer. On the other hand, for VGG19, a max-pooling layer was used after every convolutional layer. After that, no additional dropout was used for each layer; rather, relied solely on the system's built-in automatic dropout functionality to address the overfitting issue.

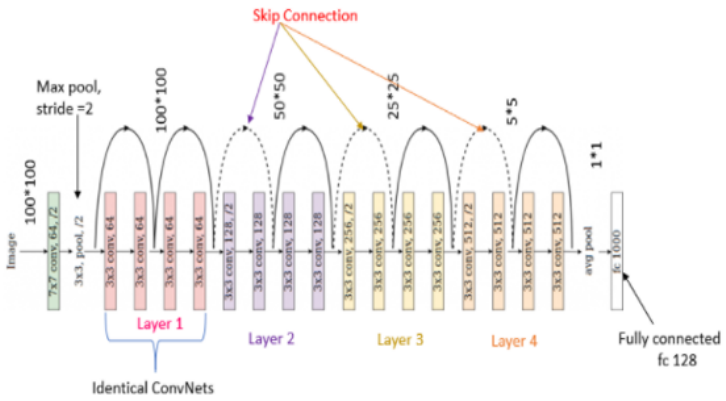

Figure 3.6: Architecture of ResNet18 [62].

When using the ResNet18 model, the input image was initially placed into the conv1 layer, which was a 7-by-7, 64 convolutional layer with a stride of 2. After that, a 3-by-3 max pool layer with stride-2 was applied. This layer's input and output were both of the same size (100-by-100). After that, the output of the previous layer was inserted as the input of the next four modulated layer. Each of these modulated layers contains four convolutional layers of 3-by-3 (Layer-1, 3-by-3, 64; Layer-2, 3-by-3, 128; Layer-3, 3-by-3, 126; Layer-4 3-by-3, 512) and the output image is reduced in each module layer. In the beginning stages of training, ResNet models typically use fewer layers than other models, which helped reduce the impact of vanishing gradients.

This was because there were fewer layers for the gradients to propagate through. As it continues to learn the feature space, the network would then gradually restore the layers that it skipped. Figure-3.6 includes notations for connections that could be skipped. The model's output was then inserted into the layer that contains the average pooling and the model's output was then inserted into the layer that contains the fully connected 1000. In this particular model, the kernel sized 7-by-7 and the padding was 3-by-3.

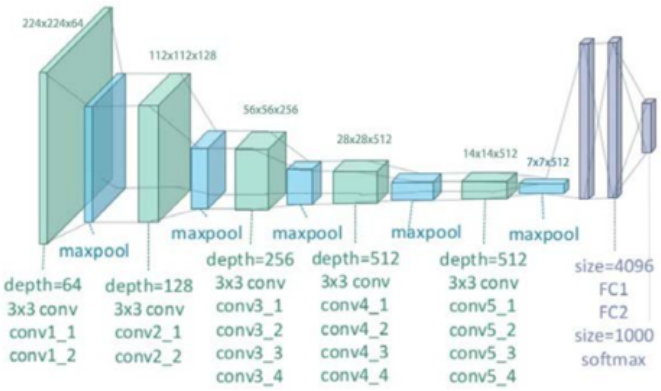

Figure 3.7: Architecture of VGG19 [63].

From Figure-3.7 it was seen, the input image was initially inserted into the conv1 layer of the VGG19 model, which was a 3-by-3 convolutional layer that has a depth of 64. After that, a max pool layer with stride-2 was applied.

The input and output sizes of the first layer were both reduced by fifty percent. Following the completion of the first layer, the output of that layer was used as the input for the second convolutional layer, which had a depth of 128 and then be followed by max-pool. The input image was also shrunk by fifty percent in this step. Its output was fed into the third convolutional layer, known as cov3, which served as its input. It had a depth of 512 and in addition to that, it decreased the image by fifty percent and was then followed by a max-pool. con4 layers were operated in the same manner as conv3 layers.

After that, the output was connected to two fully connected layers with a size of 4096, which was then followed by a fully connected layer with a size of 1000 and a soft-max layer. VGG models, in contrast to ResNet models did not skip any layers during the initial training stages of the model. They progressed through each and every stage. Kernel sized 3-by-3 and a padding sized 1-by-1 were used for this particular model.

For the purpose of Ensemble model. To begin, Ensemble model had been constructed. Following that, the loss function was declared. After that, declaration of optimizer was needed, so that system can be optimized. In the course of this work, stochastic gradient descent (SGD) algorithm was used as optimizer.

After that, this system had begun training. In order to accomplish this, it had begun by setting gradient clear to zero. After that, they turned to forward propagation. After that, calculation of error had been made. Then, in order to lower the overall error rate, the back propagation of error technique had been used. After

that, the parameters had been revised by making appropriate corrections to the error. After that, the value of each round was printed out. On each of the models, these procedures were followed.

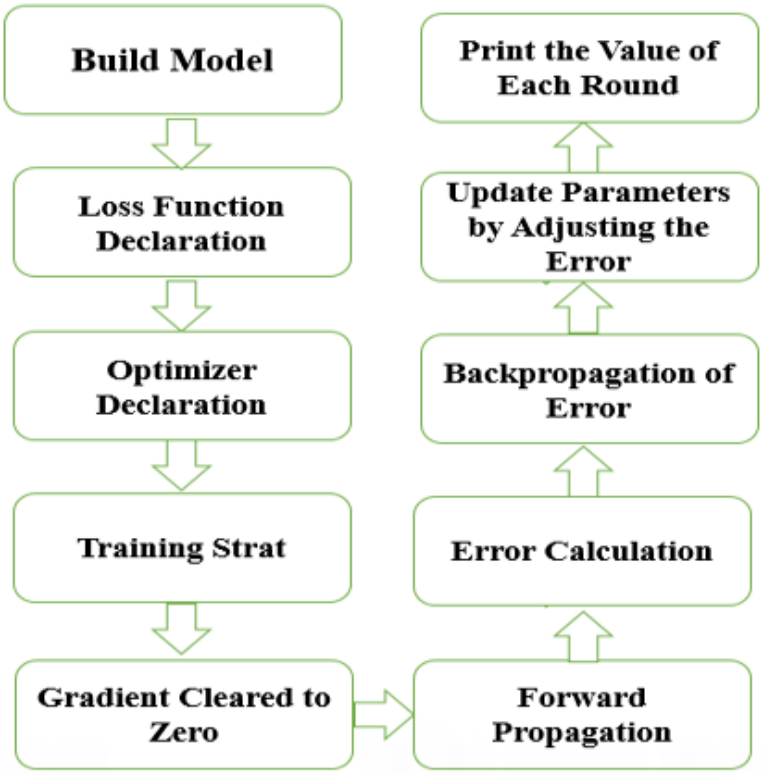

Figure 3.8: Proposed Ensemble model structure.

At long last, the dataset was run with each of the models. When developing Ensemble model, the trained data set that was obtained from the output of ResNet18 and VGG19 was used. The results of Ensemble model were then plotted in a confusion matrix that was 7-by-7. The exact same thing was done for ResNet18 and VGG19 as well.

Chapter 4

Datasets and Experiment

4.1 Datasets

A great number of datasets had been utilized in the early works in FER systems for the purpose of human emotion recognition. Among these, the data sets CK+ [64], MMI [65], Oilu-CASIA [66] , JAFEE [67], FER2013 [68], Multi-PIE [69], EmoNet [70], RAF-DB [71], AffectNet [72], and ExpW [73] were utilized for majority of the time. These data sets were very well liked by the people who work on the FER system and the researchers who used it. However, not all of the data was applicable to the solution of problems in real life. Some of the data sets were gathered under carefully supervised circumstances, which was very different from the circumstances that would be encountered in everyday life. The CK+, MMI and Oulu-CASIA datasets are examples of laboratory-controlled datasets. In each of the samples, a fixed head pose and controlled light source were maintained. These data sets contained extremely small amounts of samples overall in total number. Oulu-CASIA, one of these data sets, had more images than other laboratory-controlled data sets (2,880 image sequences collected from 80 objects), but it was not preferred when it comes to finding solutions to problems that occur in real life.

Figure 4.1: Sample of Laboratory and Real World based images [74].

It was clear from Figure-4.1 that there was a distinction to be made between the controlled environments of the laboratory and the images that typically collected from the real world.

A significant difference could be seen in both the head pose and the VSI difference. Because of this, FER systems would rather not use such data sets to solve problems that occur in real life. The Japanese Female Facial Expression (JAFFE) is a data set that was used to solve the heterogeneity of human faces in various work. This data set is now used in modern FER systems, despite the fact that it consists of a low amount of samples but achieves good accuracy for real-time problems in certain geographical regions. Data sets such as FER2013, Multi-PIE, EmotionNet, RAF-DB, AffectNet and ExpW are automatically collected from the internet.

The Multi-PIE dataset was made up of 755,370 pictures taken by 337 people from 15 different vantage points and 19 different lighting scenarios over the course of four recording sessions. This data did not completely reflect actual life events. In most instances, the Multiview facial expression analysis made use of this dataset. The Real-world Affective face dataset, also known as RAF-DB, is a real-world database containing 29,672 facial images that had been created from the internet uploaded images and feature a wide variety of expressions. There were seven basic emotion labels and eleven compound emotion labels that had been provided for the samples through the use of manually crowd-sourced annotation and reliable estimation. The FER2013, EmotionNet and AffectNet datasets were among those that are regarded as standard data sets for the solution of real-time problems. EmotionNet is a large-scale dataset that was collected randomly from the internet and contains one million facial expression images.

These images included fundamental expressions and ten compound expres-

sions. More than one million images had been gathered from across the internet and stored in AffectNet using a variety of search engines. It is by far the largest dataset that included facial expressions categorized according to two distinct emotion models (the categorial model and the dimensional model) with a total of eight fundamental expressions. FER2013 is an expansive and unrestricted dataset that was compiled automatically by the google image search API using seven fundamental expressions.

The FER2013 served as the basis dataset of this work. This dataset was not particularly large or particularly small; rather, it is of the "standard data" size and was typically utilized for the development of FER systems. There were seven directories that neatly organize the seven fundamental ways that humans express themselves. Additionally, FER20123 offered public test data set as well as a private test data set in addition to a training dataset, which was the benchmark for the solution of real-life problems in comparison to other datasets.

Figure 4.2: Sample of FER2013 dataset images [75].

4.1.1 FER2013

The Google Image Search API was used to automatically collect the data that makes up FER2013, which is a large-scale and unconstrained dataset. The dataset consisted of grayscale pictures of faces that were 48-by-48 pixels. The faces had been automatically registered in such a way that each face is approximately centered and takes up approximately the same amount of space in each image. Figure-4.2 showed sample images of the dataset.

A training data set, a public test dataset and a private test dataset were all included in the FER2013 software package. The training set included 28,709 different examples. The validation dataset of a system was typically composed of public test datasets. Public test dataset containing 3,589 examples. The private test dataset was the same size as the public test dataset and both were utilized as sources of data for assessing the accuracy of a system's predictions. It could be seen that 80% of the total dataset was the training dataset by looking at Figure-4.3. The remaining 20% of the data was split between two datasets: the public dataset and the private dataset. 10% of each set was equally distributed across all of the sets [76].

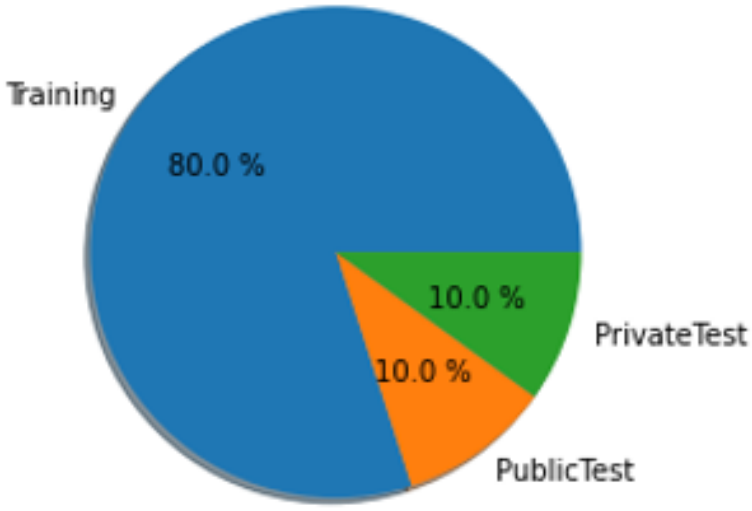

Figure 4.3: Different Sets of FER2013 dataset.

All sets were devidedd into 7 directories. Angry, Disgust, Fear, Happy, Sad, Surprise and Neutral.

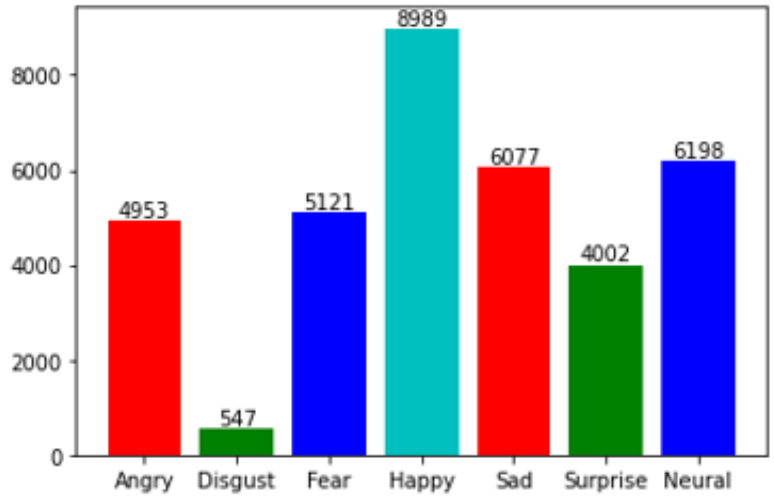

Figure 4.4: Seven directories of FER2013.

Directory "Angry" has 958 files, "Disgust" has 111 files, "Fear" has 1024 files, 1774 files in directory "Happy", 1233 for "Neutral", "Sad" has 1247 files and "Surprise" has 831 files.

4.2 Experiment

In order to use the dataset, a number of processes were required to go through in the experiment. In the beginning, it was required to pre-process the data, then the data was validated. Finally, the program had been run for a number of iterations until the desired level of precision was found based on the experiment. The steps for the entire process are detailed down below.

4.2.1 Data Pre-processing

Data pre-processing is the process of transforming raw data into an understandable format. Because the raw data could not be worked with, this was also an important step in the data mining process [58]. Before applying any machine learning or data mining algorithms, it was important to verify that the data were of sufficient quality. There were a few stages involved in the pre-processing of data. The dataset that was utilized in this work contains images in a variety of different shapes. Therefore, it was necessary to get them ready for the suggested structure [77].

4.2.1.1 Image Conversion

Altered the data type of an image as well as converted between different image types such as RGB, binary, grayscale and indexed images. Binary, indexed, grayscale and truecolor images were some of the formats that were supported by Image Processing Toolbox. Pixels were stored in a variety of different formats depending on the type of image. Initially, a manual program was utilized that was equipped with the `resize()` function from the `cv2` module in order to scale all of the images that were included in the FER2013 dataset down to the same dimensions, which were 40-by-40 pixels. The dataset included images with a resolution of 48-by-48 pixels. Then, we retrieved the photographs of the samples after they had been cropped [78].

4.2.1.2 Image Normalization

Image normalization is a procedure that was frequently utilized in the process of preparing data sets for use with artificial intelligence (AI). During this procedure, a number of images were combined into a single image using the same statistical distribution for the size and pixel values.

On the other hand, a single image could also be normalized within itself in order to achieve the desired results. Each pixel was assigned a value that was determined by dividing it by 255 and this was done so that the pixels could be normalized [78].

This time around, the normalization of the data could be achieved through the utilization of an equation, which follows:

$$Out(i, j) = In(i, j)/255.0$$

4.2.1.3 Image Transformation

Common types of image transformations included transforms. Through the use of Compose, images were able to be linked together. Functional transforms provided fine-grained control over the transformations and most of the different transform classes had a function equivalent. This cames in handy if it was needed to construct a transformation pipeline that was more complicated (e.g. in the case of segmentation tasks).

The vast majority of transformations were compatible with both PIL images and tensor images, but there were a few transformations that were exclusive to either PIL or tensor. Converting to and from PIL images was accomplished with the help of the Conversion Transforms. The transformations that were able to

process tensor images also had the capability of processing tensor image batches. Tensor images were represented by tensors with the shape **(C, H, W)**, where C represented the number of channels and H and W represented the height and width of the image, respectively. A tensor with the shape **(B, C, H, W)** represented a batch of tensor images, where B represented the number of images contained in the batch.

The tensor dtype provides an implicit definition of the expected range of the values that were contained within a tensor image. Tensor images that had a float dtype should have values that fall somewhere in the range `[0, 1]`. Tensor images that had a dtype of integer are expected to have values in the range `[0, MAX DTYPE]`, where MAX DTYPE was the highest value that can be represented by that particular dtype. Randomized transformations would apply the same transformation to all of the images in a given batch, but the transformations they produce varied from call to call. Functional transformations was used if transformations to be reproducible across calls [78].

4.2.1.4 Dataset Reshape

`torchvision.transforms`. Random Crop`(size, padding=None, pad-if-needed = False, fill=0, padding-mode='constant')` size, padding = None, pad-if-needed = False and fill=0. [source]. Performed a haphazard crop on the image that was provided. If non-constant padding was used, the input was expected to have at most two leading dimensions, but if the image was a torch Tensor, it was expected to have the shape `[..., H, W]`, where means an arbitrary number of leading dimensions.

If the image was a torch Tensor, it was expected to have the shape `[..., H, W]`, size (either a sequence or an integer) was the output size that the crop should ideally have. If size was an integer rather than a sequence like **(H, W)**, then a square crop with dimensions size and size was performed.

If the computer was given a sequence that was only one element long, it interpreted as `(size[0], size[0])`, padding (int or sequence, optional) – Padding that can be added on each of the image's borders if desired. Default was none.

If only a single int is given, that value would be used to pad all of the borders. In the event that a sequence of length 2 was supplied, this represents the padding on the left and right, as well as the top and bottom. This was the case whenever a sequence of length 4 is presented. padding for the left, top, right and bottom borders, respectively. Padding for the left border. Note because padding as a single int was not supported in torchscript mode, a sequence was used of length

`1:[padding,]`. Pad if needed (boolean).

If the image was smaller than the size that was desired, it would add padding to the image so that an exception was not raised. Given that cropping occurs after padding, it would appear that padding was carried out at an arbitrary offset. Fill (number or str or tuple) – Pixel fill value for constant fill. Default was 0.

If a tuple had three elements, those elements were filled in as the R, G, and B channels, respectively. Only when the padding-mode was set to constant will this value be utilized. The torch Tensor only accepted numbers as an argument. When it came to values, PIL Image only supports integers, strings and tuples. Padding-mode (STR) referred to the type of padding being used. Should have one of the following properties: constant, edge, reflect or symmetric. Default is constant. pads with a constant value, the value of which was specified with fill edge pads with the last value at the image's edge constant: padded with a value that was constant fill edge If a 5D torch tensor was input, the last 3 dimensions of the tensor was padded rather than the last 2 dimensions.

Reflect: pads with a reflection of the image without repeating the most recent value on the edge. For instance, padding `[1, 2, 3, 4]` with 2 elements on both sides while the mode was set to reflect will produce the result `[3, 2, 1, 2, 3, 4, 3, 2]`. Pads with a reflection of the image that repeats the previous value on the edge constitute symmetry. In symmetric mode, padding `[1, 2, 3, 4]` with 2 elements on both sides resulted in `[2, 1, 1, 2, 3, 4, 4, 3]` [78].

4.2.1.5 Horizontal Flip

The process of rotating an image along a horizontal or vertical axis was referred to as flipping.

In a vertical flip, the object being flipped would be on the horizontal axis and in a horizontal flip, the object being flipped would be on the vertical axis. Randomly rotated the image in the horizontal direction using the probability that was given.

If the image was a torch tensor, then it was anticipated that it had the shape `[..., H, W]`, where, refers to any arbitrary number of leading dimensions. P (a float) represented the probability that the image would be flipped. Parameters Default value was 0.5 [78].

4.2.1.6 Coversion Transforms

Conversion Transforms were a class that was found in the `torchvision.transforms` package with the mode set to none. `[source]` perform a PIL image conver-

sion on a tensor or an ndarray. The torchscript language was not supported by this transformation. A torch that could be converted. Tensor with the shape C x H x W or a numpy ndarray with the shape H x W x C could be converted into a PIL Image while the value range was maintained.

The color space and pixel depth of the input data were specified in the parameters mode `(PIL.Image mode)` (optional). When the mode was None, which was the default setting, the following presumptions were made about the input data: The mode was presumed to be RGBA if the input had a total of four channels. If the input had three channels, then RGB was presumed to be the mode being used.

The mode was presumed to be LA if the input consists of more than one channel. If there was only one channel on the input, the data type would decide what mode to use (i.e int, float, short). Examples employing the JIT and ToPILImage:Tensor transforms [78].

4.2.1.7 PIL Image Conversion

Torchvision.transforms. [original source] ToTensor. Performed the tensor conversion on a PIL Image or a numpy.ndarray. The torchscript language was not supported by this transformation.

Creates a torch from a PIL Image or numpy.ndarray with a height, width, and depth value between 0 and 255. FloatTensor with the shape **(C x H x W)** in the range [0.0, 1.0] if the PIL Image belongs to one of the modes `(L, LA, P, I, F, RGB, YCbCr, RGBA, CMYK, 1)` or if the numpy.ndarray had dtype = np.uint8.

If the PIL Image did not belong to one of the modes, FloatTen in all other circumstances, the tensors were handed back without having their scaling applied. This transformation should not be used when transforming target image masks because the input image was scaled to the range [0.0, 1.0]. Consulted the references for information on how to put the transformations for image masks into practice [78].

4.2.1.8 Image Tupling

`Torchvision.transforms`. TenCrop with size parameters and vertical flip was set to false. The given image was cut into four corners, a central crop and the flipped version of these and then croped the corners again (horizontal flipping is used by default).

If the image was a torch Tensor, then it was anticipated that it will have the shape `[..., H, W]`, where refers to any arbitrary number of leading dimensions.

There might be a discrepancy between the number of inputs and targets that the dataset returns and the number of images that this transform returns. This transform returned a tuple of images. Took a look at the example that follows to see how this should be handled. size (either a sequence or an integer) was the output size that the crop should ideally have.

If size was an integer rather than a sequence like **(H, W)**, then a square crop with dimensions size and size was performed. If the computer was given a sequence that was only one element long, it would be interpreted as `(size[0], size[0])`. `vertical flip (bool) (bool)`. Instead of moving horizontally forward, vertical flipping (img) [source] was implemented, img (either a PIL Image or a Tensor) was the image that is going to be cropped. It returned a tuple containing 10 images. PIL Images and tensor images both were counted as images [78].

4.2.1.9 Generic Transforms

`Torchvision.transforms.Lambda(lambd)`. Performed the transformation using a user-defined lambda. The torchscript language is not supported by this transformation. Parameters lambd (function) is the name of the lambda/function that will be used for the transform [78].

4.2.2 Platform

PyTorch is a tensor library that had been optimized to work specifically for use in deep learning applications running on GPUs and CPUs. The facebook artificial intelligence research team was primarily responsible for developing this open-source machine learning library for the Python programming language.

It was based on the torch library and was used for applications such as computer vision and natural language processing. It was one of the machine learning libraries that sees widespread usage. PyTorch was the foundation for a number of different pieces of deep learning software, including tesla autopilot, uber's pyro, hugging face's transformers, pyTorch lightning and catalyst, to name just a few [79]. PyTorch provided two high-level features:

- Tensor computing (like NumPy) with strong acceleration via graphics processing units (GPU).

- Deep neural networks built on a type-based automatic differentiation system.

4.2.3 Data Partitioning

During the process of loading the master data, data partitioning was a method that physically divided the data into separate sections. By employing this method, a table was cutted up into more manageable sections in accordance with the rules that the user specifies.

It was typically used when there were very large tables, which required a significant amount of time to read the entire data set. Because of this, it enabled system to improve the maintenance, performance or management of the database. This procedure was broken down into two stages. The test-and-train split was the first phase and the validation of data was the second phase.

4.2.3.1 Train Test Split

The train-test split is a method that was used to evaluate the effectiveness of an algorithm for machine learning. It was applied for problems involving classification or regression, in addition to being useful for any supervised learning algorithm.

In this step of the process, a dataset was taken and it was split into two separate subsets. During the earlier phase, the dataset was split into test data and train data using the train `test split()` function. 20% of the data was set aside for determining the final accuracy, while the remaining eighty percent was used for training purposes.

Later on, another iteration of the train `test spilt()` function was carried out in order to generate the validation data, which comprised 20% of the training data. During the training phase, the accuracy of the model was evaluated with the help of the validation set.

4.2.3.2 Validation Data

Before using, importing or otherwise processing data in any way, it was important to perform a process known as data validation, which involved checking the correctness and quality of the source data.

Depending on the requirements placed by the destination or the goals that it seeked to accomplish, various types of validation may be carried out. The process of validating data is one type of data cleaning. Following the test-train split, FER2013 was able to comprehend 28,709 image examples splitted between one public test set containing 3,589 and one private test set containing 3,589.

4.2.4 Program Execution

It was not a given that increased performance will result from running a program on a multicore platform, even if that platform was homogeneous. The most important thing was to allow each of the cores to contribute to the overall computation for the longest amount of time possible.

For problem settings that were anything but trivial and especially for heterogeneous platforms such as ones made of CPU and GPU cores, this called for some form of load balancing, which could be either explicit or implicit workload shifts between the cores, in order to eliminate idle times to the greatest extent possible. Specifically, this called for explicit or implicit workload shifts between the cores.

Because of the extensive nature of used dataset, it was decided to conduct system training in batches. A batch size of 128 was utilized, which indicated that 128 samples were used for each iteration of the process. Last but not least, Each test run was given a duration of various epochs (20,30 and 300), which enabled the computer program to carry out all those iterations across each training dataset.

Chapter 5

Result Analysis

5.1 Accuracy Analysis

Following a number of phases of experimentation aimed at determining the best possible outcome, the Ensemble model that was suggested for this research was finally completed. The dataset had been put through its paces using all three models. In order to locate an accuracy level that was satisfactory, the system was tried every possible combination. The results will be presented in the following sections in a clear and concise manner.

5.1.1 Accuracy Distribution on FER2013 Using ResNet18 Model

To begin, untrained ResNet18 model was applied to the analysis of the FER2013 dataset. This allowed to evaluate the precision of this model structure. After finishing the construction of Ensemble model, the dataset training begun. This allowed to gradually improve the accuracy of Ensemble model. In order to evaluate the performance of a single model using varying degrees of precision, we constructed a confusion matrix with 7-by-7 cells (First, with 20 epochs then 30 epochs and with 300 epochs).

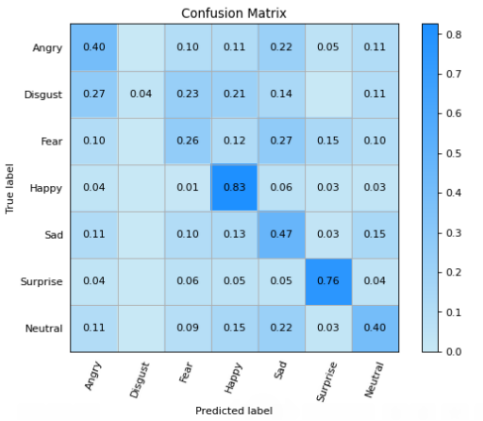

Figure 5.1: Confusion matrix for ResNet18 with 20 epochs.

Figure 5.1 shows that after 20 epochs, accuracy for the emotion angry was 40%, accuracy for the emotion disgust was 4%, accuracy for the emotion fear was 26%, accuracy for the emotion happy was 83%, accuracy for the emotion sad was 47%, accuracy for the emotion surprised was 76%, and accuracy for neutral was 40%. With this model, system was only able to classify five of the seven emotions after 20 epochs of data collection. An overall accuracy of **53.60%** was achieved, with incorrect classifications of the emotions disgust and fear.

Once more, the ResNet18 model was executed with 30 iterations. As shown in Figure 5.2, accuracy for identifying emotion angry was 35%. The accuracy for the emotion of disgust was 16%, while the accuracy for fear was 27%.

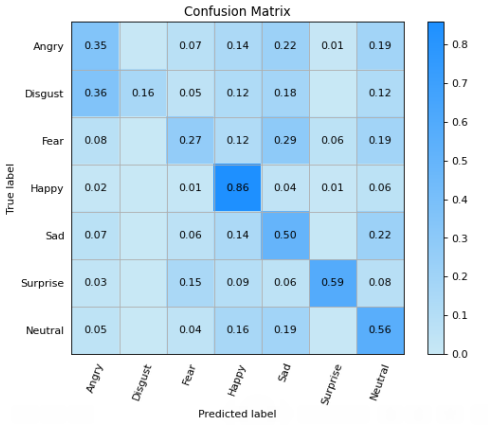

Figure 5.2: Confusion matrix for ResNet18 with 30 epochs.

Happy received 86% of the accuracy, sad received 50%, surprise received 59% and neutral received 56%. System was successful in classifying five of the seven emotions with this model after running it for 30 epochs. With a total accuracy

of **55.11%**, the emotions Disgust and Fear were not accurately identified as their respective types.

The ResNet18 model was then executed for a total of 300 iterations. Figure 5.3 shows that the accuracy for the emotion Angry was 51%, accuracy for the emotion Disgust was 55%, accuracy for the emotion Fear was 46%, accuracy for the emotion Happy was 85%, accuracy for the emotion Sad was 55%, accuracy for the emotion Surprise was 80%, and accuracy for the emotion Neutral was 62%. System was successful in classifying seven of the seven emotions with this model after running it for 300 epochs. An overall accuracy of **64.61%** was achieved, with correct classifications of the emotions disgust and fear.

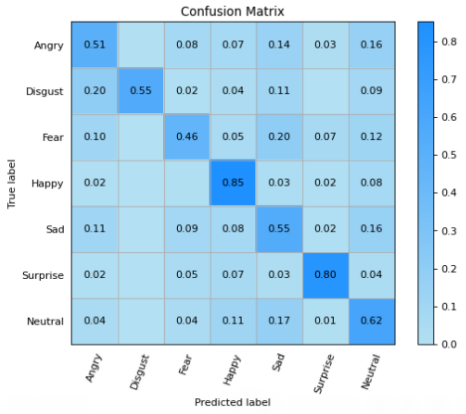

Figure 5.3: Confusion matrix for ResNet18 with 300 epochs.

Table 5.1: Dataset accuracy using ResNet18 model

Emotions	Resnet18 with Different Epochs			Final Emotion Classified		
	20 Epochs	30 Epochs	300 Epochs	20 Epochs	30 Epochs	300 Epochs
Angry	40%	35%	51%	Angry	Angry	Angry
Disgust	**4%**	**16%**	55%	**Angry**	**Angry**	Disgust
Fear	**26%**	**27%**	46%	**Sad**	**Sad**	Fear
Happy	83%	86%	85%	Happy	Happy	Happy
Sad	47%	50%	55%	Sad	Sad	Sad
Surprise	76%	59%	80%	Surprise	Surprise	Surprise
Neutral	40%	56%	62%	Neutral	Neutral	Neutral

N.B. Bold Letter means the wrong emotion was picked up.

- For **20** iterations emotion detection rate was **71.42%** and accuracy percentage was **53.60%**.
- For **30** iterations emotion detection rate was **85.71%** and accuracy percent-

age was **55.11%**.

- For **300** iterations emotion detection rate was **100%** and accuracy percentage was **64.61%**.

5.1.2 Accuracy Distribution on FER2013 Using VGG19 Model

Following, FER2013 dataset was run through an untrained VGG19 model so that the accuracy of this model structure could be evaluated. Following the completion of this model, the dataset training was began. A 7-by-7 confusion matrix was collected for different accuracy levels in order to compare the accuracy of the same model (First, with 20 epochs then 30 epochs and with 300 epochs).

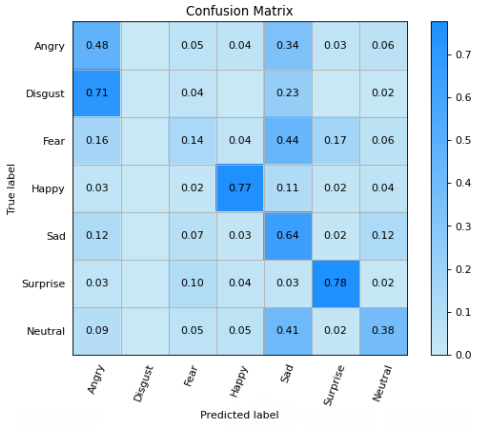

Figure 5.4: Confusion matrix for VGG19 with 20 epochs.

At 20 epochs, we observed from Figure 5.4 that an accuracy of 48% for the emotion angry, an accuracy of NULL for the emotion disgust, 14% for the emotion fear, 77% for the emotion happy, 64% for the emotion sad, 78% for the emotion surprise and 38% for the emotion neutral. With this model, system was only successful in classifying four of the seven emotions after running it for 20 epochs. Overall, the classification of the emotions disgust, fear and neutral was incorrect, resulting in a percentage of accuracy of **54.22%**.

Once more, the VGG19 model was executed with 30 iterations. Figure 5.5 revealed that system achieved an accuracy of 40% for the emotion of anger, NULL for the emotion of disgust, 16% for the emotion of fear, 84% for the emotion of happiness, 72% for the emotion of sadness, 69% for the emotion of surprise and 57% for the emotion of neutrality. System was successful in classifying five of the seven emotions with this model after running it for 30 epochs. An overall

accuracy of **59.09%** was achieved, with incorrect classifications of the emotions disgust and fear respectively.

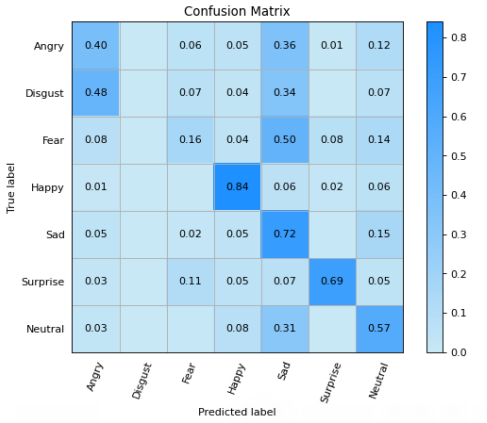

Figure 5.5: Confusion matrix for VGG19 with 30 epochs.

The next step involved running the VGG19 model through 300 iterations. According to the data presented in Figure 5.6, system's level of accuracy for the emotion of anger was 55%, while level of accuracy for the emotion of disgust was 62%, level of accuracy for the emotion of fear was 53%, level of accuracy for the emotion of happiness was 85%, level of accuracy for the emotion of sadness was 64%, level of accuracy for the emotion of surprise was 82% and level of accuracy for the emotion of neutrality was 66%. After testing this model for a total of 300 epochs, system was able to successfully categorize seven of the seven feelings that are possible and successful in categorizing everything correctly, leading to an accuracy score of **68.90%** overall.

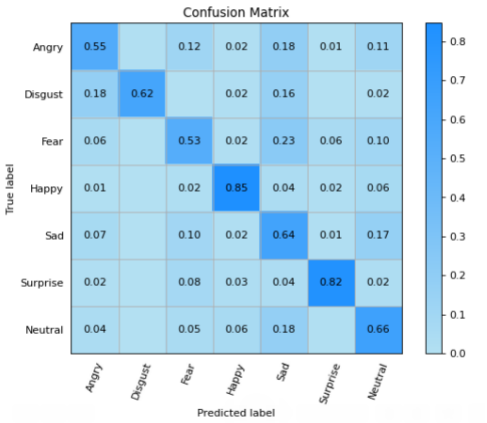

Figure 5.6: Confusion matrix for VGG19 with 300 epochs.

Table 5.2: Dataset accuracy using VGG19 model

Emotions	vgg19 with Different Epochs			Final Emotion Classified		
	20 Epochs	30 Epochs	300 Epochs	20 Epochs	30 Epochs	300 Epochs
Angry	48%	40%	55%	Angry	Angry	Angry
Disgust	**NULL**	**NULL**	62%	**Angry**	**Angry**	Disgust
Fear	**14%**	**16%**	53%	**Sad**	**Sad**	Fear
Happy	77%	84%	85%	Happy	Happy	Happy
Sad	64%	72%	64%	Sad	Sad	Sad
Surprise	78%	69%	82%	Surprise	Surprise	Surprise
Neutral	**38%**	57%	66%	**Sad**	Neutral	Neutral

N.B. Bold Letter means the wrong emotion was picked up.

- For **20** iterations emotion detection rate was **57.14%** and accuracy percentage was **54.22%**.
- For **30** iterations emotion detection rate was **71.42%** and accuracy percentage was **59.09%**.
- For **300** iterations emotion detection rate was **100%** and accuracy percentage was **68.90%**.

5.1.3 Accuracy Distribution on FER2013 Using Ensemble Model

Ultimately, FER2013 dataset had been ran in an untrained Ensemble model so that the accuracy of this model structure could be evaluated. Following the completion of this model, the dataset training had begun. A 7-by-7 confusion matrix was collected for different accuracy levels in order to compare the accuracy of the same model (First, with 20 epochs then 30 epochs and with 300 epochs).

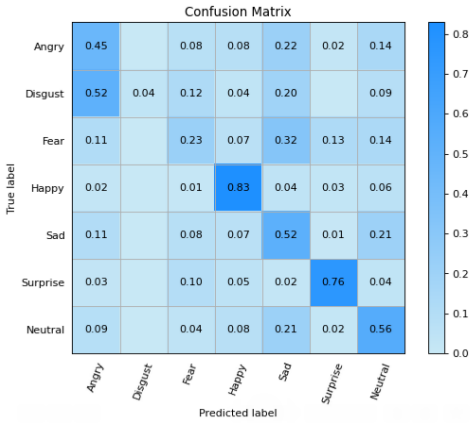

Figure 5.7: Confusion matrix for Ensemble with 20 epochs.

Figure 5.7 showed that after 20 epochs, accuracy for the emotion angry was 45%, accuracy for the emotion disgust was 4%, accuracy for the emotion fear was 23%, accuracy for the emotion happy was 83%, accuracy for the emotion sad is 52%, accuracy for the emotion surprised was 76%, and accuracy for the emotion neutral was 56%. With this model, system was only able to categorize five of the seven emotions after 20 epochs of data collection. The overall accuracy was **56.48%** and the emotions Fear and Disgust were misclassified as their respective types.

Once more, the Ensemble model was run with 30 epochs of data. Figure 5.8 reveals that system achieved an accuracy of 51% for the emotion of anger, 12% for the emotion of disgust, 30% for the emotion of fear, 84% for the emotion of happiness, 56% for the emotion of sadness, 76% for the emotion of surprise, and 58% for the emotion of neutrality. System was successful in classifying six of the seven emotions with this model after running it for 30 epochs. The overall accuracy was just 60.76% and the emotion Disgust was incorrectly categorized as a result.

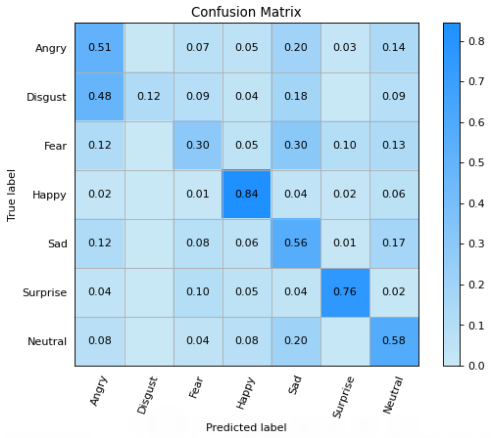

Figure 5.8: Confusion matrix for Ensemble with 30 epochs.

The Ensemble model was then applied to an additional 300 epochs of data. Figure 5.9 demonstrates that system was successful in identifying emotions with an accuracy of 60% for the feeling of anger, 59% for the feeling of disgust, 54% for the feeling of fear, 86% for the feeling of happiness, 61% for the feeling of sadness, 83% for the feeling of surprise and 66% for the feeling of neutrality. After testing this model for a total of 300 epochs, system was able to successfully categorize seven of the seven feelings that are possible. A total accuracy of just 69.54% was achieved and as a direct consequence, the feeling of disgust was correctly identified.

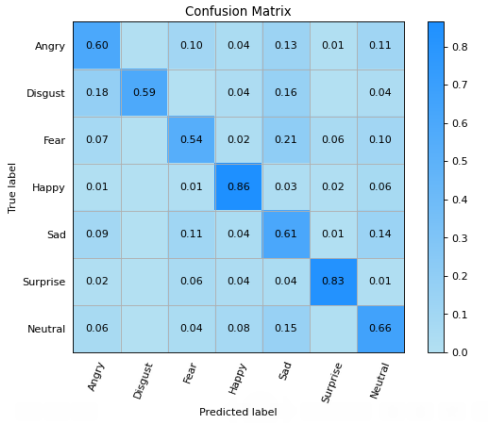

Figure 5.9: Confusion matrix for Ensemble with 300 epochs.

Table 5.3: Dataset accuracy using Ensemble model

Emotions	Ensemble Model with Different Epochs			Final Emotion Classified		
	20 Epochs	30 Epochs	300 Epochs	20 Epochs	30 Epochs	300 Epochs
Angry	45%	51%	60%	Angry	Angry	Angry
Disgust	**4%**	**12%**	59%	**Angry**	**Angry**	Disgust
Fear	**23%**	30%	54%	**Sad**	Fear	Fear
Happy	83%	84%	86%	Happy	Happy	Happy
Sad	52%	56%	61%	Sad	Sad	Sad
Surprise	76%	76%	83%	Surprise	Surprise	Surprise
Neutral	56%	58%	66%	Neutral	Neutral	Neutral

N.B. Bold Letter means the wrong emotion was picked up.

- For **20** iterations emotion detection rate was **71.42%** and accuracy percentage was **57.48%**.
- For **30** iterations emotion detection rate was **85.71%** and accuracy percentage was **60.76%**.
- For **300** iterations emotion detection rate was **100%** and accuracy percentage was **69.54%**.

Table 5.4: Comparison of Accuracy with different models

Emotions	Resnet18		VGG19		Ensemble Model	
	20 Epochs	30 Epochs	20 Epochs	30 Epochs	20 Epochs	30 Epochs
Angry	40%	35%	48%	40%	45%	51%
Disgust	**4%**	**16%**	**NULL**	**NULL**	**4%**	**12%**
Fear	**26%**	**27%**	**14%**	**16%**	**23%**	30%
Happy	83%	86%	77%	84%	83%	84%
Sad	47%	50%	64%	72%	52%	56%
Surprise	76%	59%	78%	69%	76%	76%
Neutral	40%	56%	**38%**	57%	56%	58%

N.B. Bold Letter means the wrong emotion was picked up.

- For **20** iterations maximum emotion detection rate was **71.42%** detected using **Resnet18** and **Ensemble model**. Maximum accuracy percentage was **57.48%** deteted using **Ensemble model**.

- For **30** iterations maximum emotion detection rate was **85.71%** detected using **Resnet18** and **Ensemble model**. Maximum accuracy percentage was **60.76%** deteted using **Ensemble model**.

Table 5.5: Comparison of Accuracy with different models

Emotions	Resnet18		VGG19		Ensemble Model	
	30 Epochs	300 Epochs	30 Epochs	300 Epochs	30 Epochs	300 Epochs
Angry	35%	51%	40%	55%	51%	60%
Disgust	**16%**	55%	**NULL**	62%	**12%**	59%
Fear	**27%**	46%	**16%**	53%	30%	54%
Happy	86%	85%	84%	85%	84%	86%
Sad	50%	55%	72%	64%	56%	61%
Surprise	59%	80%	69%	82%	76%	83%
Neutral	56%	62%	57%	66%	58%	66%

N.B. Bold Letter means the wrong emotion was picked up.

- For **30** iterations maximum emotion detection rate was **85.71%** detected using **Resnet18** and **Ensemble model**. Maximum accuracy percentage was **60.76%** deteted using **Ensemble model**.

- For **300** iterations maximum emotion detection rate was **100%** detected for all models. Maximum accuracy percentage was **69.54%** deteted using **Ensemble model**.

5.2 Loss Analysis

There was at least one category of emotions generating low levels of accuracy across all three models had been observed. However, in addition to that, there were also other problems, which, regardless of the model, caused a drop in performance. Few such scenarios were found after investigating the samples, which were predicted wrong after the test runs.

5.2.1 Faulty Images

The presence of invalid images in the dataset, which cannot be correctly categorized into any of the available options, was the issue that had come across the most frequently. The samples were almost never placed in the appropriate category of feelings. Some of those examples could be seen in the first set of samples that are presented in Figure5.10.

Figure 5.10: Contradicted predictions in FER2013.

5.2.2 Loss Distribution

All of these incident contributed to an overall increase in the number of failed tests. Despite this, the test loss for each of the three models was not particularly substantial. At such a low rate of epochs, they could be determined to be acceptable when compared to the early researches that were mentioned. Figure 5.11 presents a comparison of the suggested accuracy and loss rate of Ensemble model for each epoch and it is shown below-

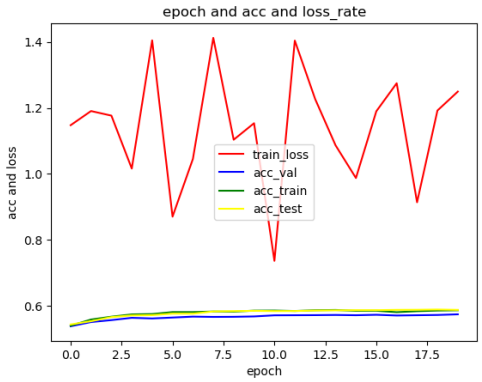

Figure 5.11: Accuracy and loss-rate of FER2013 in Ensemble model.

Figure 5.12 presents a comparison of the suggested accuracy and loss rate of ResNet18 model for each epoch.

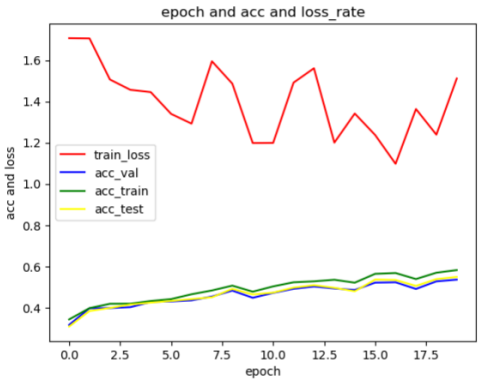

Figure 5.12: Accuracy and loss-rate of FER2013 in Resnet18 model.

Figure 5.13 presents a comparison of the suggested accuracy and loss rate of VGG19 model for each epoch.

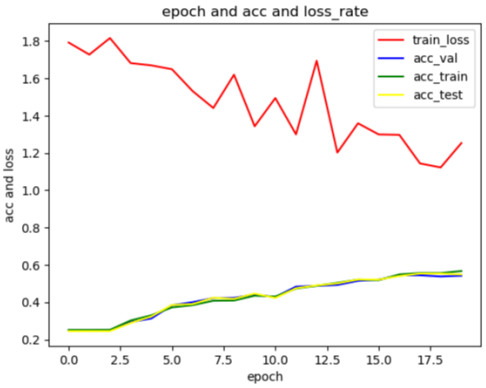

Figure 5.13: Accuracy and loss-rate of FER2013 in VGG19 model.

Chapter 6

Conclusion

6.1 Achievements

Convolutional neural networks were shown to be the best way to recognize human emotions at the moment. However, ever since that moment, the researchers working in this particular field have had their ability to address problems based in the actual world severely limited to the point that they can only consider applying laboratory-controlled datasets in their investigations. The fact that only a very limited number of researchers worked on real-world condition datasets while at the same time the characteristics they used were additional features in some way is unsatisfactory.

The detection of any type of emotion across the dataset was much improved as a result of this research and this improvement was achieved without the utilization of any new manufactured features or training data. Discussion had taken place over each and every one of the several categories of human feelings, all of which, when taken together, provide the broadest possible spectrum of instances. In general, the method that was discussed in this work presents a strategy that, in comparison to previous automated models, had delivered better degree of improvement in the recognition of human emotions. This proves that the work had been conducted is efficient and effective.

A single dataset was worked on by using several models, all of which shared the same attributes and epoch rate with one another. The goal was to develop and build a system that would function perfectly well without the need for any extra data to be incorporated into it in any manner, shape, or form. The system was developed successfully integrating models from ResNet18 and VGG19 and successfully run the data set using the Ensemble model. Both of these accomplishments were possible because of hard work.

At 20 epochs, the Ensemble model was able to reach an accuracy of **57.48%**, at 30 epochs, it was able to achieve **60.76%** and at 300 epochs it achieved **69.54%**. For the FER2013 dataset, the state-of-the-art accuracy without using extra training data and handcrafted features is **70.02%** [80] at 300 epochs, whereas the accuracy using extra training data is **73.28%** [81]. Both of these figures are based on not using handcrafted features. At both the low and the high epochs, system was successful in achieving the greatest degree of precision that was theoretically attainable [82].

Over the course of twenty, thirty, and three hundred epochs, system was able to correctly identify five, six and all seven feelings respectively. System could only identify disgust with a moderate level of accuracy because there weren't enough samples in the data set for this directory file. As a result, a low degree of precision was only able to be detected. This model had a degree of accuracy that was noticeably superior to that of the conventional CNN models. As a result, it was possible to assume that the following are some of the things that had been successfully accomplished:

- This was the first research study that had successfully developed a Ensemble model by combining ResNet18 and VGG19 and the dataset that was used has the maximum number of character classes that are currently available.

- Unlike most previous study that used only Automatic Features, it was able to identify human emotion with a level of precision that is superior. As it was capable of recognizing a wider range of human feelings.

- When contrasted with the ResNet18 and VGG19 models, the results showed that system was able to achieve a higher level of overall accuracy as well as more accurate detection of emotion classification classes.

6.2 Comparison with the existing methods

The vast majority of the work that had been done on the FER system has entailed the utilization of handcrafted features in some capacity. In recent years, software engineers had been working toward the objective of establishing a FER system that does not include any handcrafted features in order to obtain a universal standard. This goal was intended to be accomplished through attaining a universal standard.

The greatest achievable accuracy for the FER2013 data sets had been measured up to this time to be **70.02%** after 300 iterations. This was achieved without the use of any additional training data or created features. Without using any supplemental training data or manually crafted features, system was able to get FER2013's accuracy up to **69.54%** after 300 iterations.

This was accomplished without the use of any handcrafted features. This was also quite close to the highest possible accuracy for the same number of epochs per second. As a result of its efforts, proposed model had reached the state of the art, which it accomplished by accurately detecting seven different emotions with a high degree of precision. A rate of accuracy that was greater than fifty percent was achieved in the identification of all of the feelings, and the achieved rate of accuracy for each feeling was remarkably close to that of the most sophisticated model.

6.3 Limitation

The first difficulty that came upon was getting access to the dataset that was being used. When it comes to the resolution of real-world issues by researchers, the vast majority of the datasets that were considered to be industry standards for FER systems were either not freely available or require a significant amount of effort to access. This was a problem because both of these circumstances make it difficult to access the data. Because of this, a single dataset was used in order to achieve the purpose of this work, which was relevant to our daily life.

There was still a requirement to make enhancements to the data set and certain folders do not have an adequate quantity of data. Due to the absence of such an emotion, it was challenging to classify and as a consequence, the accuracy of a model diminishes, which had a detrimental effect on the total accuracy rate.

6.4 Future Works

Only one dataset that was based on the real world was used in the creation of this autonomous FER model. There was still a significant amount of room to expand upon this work. The future effort ought to center on these three primary concerns.

- In this work, emotion classification had been solved; nevertheless, the solution to the problem of emotion regression is required in real-world systems, along with the answer to the problem of emotion classification.

- Building a new data collection with more sample variants and object numbers needs to be done in order to solve the problem of heterogeneity in human faces. It is now very necessary to tackle real-life problems in order to go forward. Researchers are working hard to find a solution to this issue, but so far they have not identified the best course of action.

- The primary focus of the dataset was on the fundamental feelings that humans experience. Compound emotions are something that humans are capable of producing in addition to their basic emotions. If FER is to be utilized for security purposes, it is imperative that the model be trained with as many complex emotions as is practically possible.

References

[1] R. Cowie, E. Douglas-Cowie, N. Tsapatsoulis, G. Votsis, S. Kollias, W. Fellenz, and J. G. Taylor, "Emotion recognition in human-computer interaction," *IEEE Signal Processing Magazine*, vol. 18, no. 1, pp. 32–80, 2001. doi: 10.1109/79.911197

[2] R. M. Mehmood, R. Du, and H. J. Lee, "Optimal feature selection and deep learning ensembles method for emotion recognition from human brain eeg sensors," *IEEE Access*, vol. 5, pp. 14797–14806, 2017. doi: 10.1109/ACCESS.2017.2724555

[3] T. Song, W. Zheng, C. Lu, Y. Zong, X. Zhang, and Z. Cui, "Mped: A multimodal physiological emotion database for discrete emotion recognition," *IEEE Access*, vol. 7, pp. 12177–12191, 2019. doi: 10.1109/ACCESS.2019.2891579

[4] E. Batbaatar, M. Li, and K. H. Ryu, "Semantic-emotion neural network for emotion recognition from text," *IEEE Access*, vol. 7, pp. 111866–111878, 2019. doi: 10.1109/ACCESS.2019.2934529

[5] M. Leo, P. Carcagnì, C. Distante, P. Spagnolo, P. L. Mazzeo, A. C. Rosato, S. Petrocchi, C. Pellegrino, A. Levante, F. De Lumè, *et al.*, "Computational assessment of facial expression production in asd children," *Sensors*, vol. 18, no. 11, p. 3993, 2018. doi: 10.3390/s18113993

[6] Q. Meng, X. Hu, J. Kang, and Y. Wu, "On the effectiveness of facial expression recognition for evaluation of urban sound perception," *Science of The Total Environment*, vol. 710, p. 135484, 2020.

[7] Y. Zhang, Y. Li, B. Xie, X. Li, and J. Zhu, "Pupil localization algorithm combining convex area voting and model constraint," *Pattern Recognition and Image Analysis*, vol. 27, no. 4, pp. 846–854, 2017. doi: 10.1134/S1054661817040216/

[8] H. Meng, N. Bianchi-Berthouze, Y. Deng, J. Cheng, and J. P. Cosmas, "Time-delay neural network for continuous emotional dimension prediction from facial expression sequences," *IEEE Transactions on Cybernetics*, vol. 46, no. 4, pp. 916–929, 2015. doi: 10.1109/TCYB.2015.2418092

[9] F. Xu and J. Zhang, "Facial microexpression recognition: a survey," *Zidonghua Xuebao/Acta Automatica Sinica*, vol. 43, no. 3, pp. 333–348, 2017. doi: 10.1109/TAFFC.2022.3205170

[10] M. S. Özerdem and H. Polat, "Emotion recognition based on eeg features in movie clips with channel selection," *Brain Informatics*, vol. 4, no. 4, pp. 241–252, 2017. doi: 10.1007/s40708-017-0069-3/

[11] Y.-I. Tian, T. Kanade, and J. F. Cohn, "Recognizing action units for facial expression analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 97–115, 2001. doi: 10.1109/34.908962

[12] R. E. Jack, O. G. Garrod, H. Yu, R. Caldara, and P. G. Schyns, "Facial expressions of emotion are not culturally universal," *Proceedings of the National Academy of Sciences*, vol. 109, no. 19, pp. 7241–7244, 2012. doi: 10.1073/pnas.1200155109/

[13] X. Yu, S. Zhang, Z. Yan, F. Yang, J. Huang, N. E. Dunbar, M. L. Jensen, J. K. Burgoon, and D. N. Metaxas, "Is interactional dissynchrony a clue to deception? insights from automated analysis of nonverbal visual cues," *IEEE Transactions on Cybernetics*, vol. 45, no. 3, pp. 492–506, 2014. doi: 10.1109/TCYB.2014.2329673

[14] M. Bouhlal, K. Aarika, R. A. Abdelouahid, S. Elfilali, and E. Benlahmar, "Emotions recognition as innovative tool for improving students' performance and learning approaches," *Procedia Computer Science*, vol. 175, pp. 597–602, 2020. doi: 10.1016/j.procs.2020.07.086

[15] A. A. M. Al-Modwahi, O. Sebetela, L. N. Batleng, B. Parhizkar, and A. H. Lashkari, "Facial expression recognition intelligent security system for real time surveillance," in *Proc. of World Congress in Computer Science, Computer Engineering and Applied Computing*, 2012. www.researchgate.net/publication/236987503 [accessed: Jan. 10, 2023].

[16] F. Ren and C. Quan, "Linguistic-based emotion analysis and recognition for measuring consumer satisfaction: an application of affective computing,"

Information Technology and Management, vol. 13, no. 4, pp. 321–332, 2012. doi: 10.1007/s10799-012-0138-5

[17] A. B. Nassif, I. Shahin, I. Attili, M. Azzeh, and K. Shaalan, "Speech recognition using deep neural networks: A systematic review," *IEEE Access*, vol. 7, pp. 19143–19165, 2019. doi: 10.1109/ACCESS.2019.2896880

[18] W. B. Group, *World development report 2016: Digital dividends*. World Bank Publications, 2016. https://documents1.worldbank.org/curated/en/896971468194972881/pdf/102725-PUB-Replacement-PUBLIC.pdf [accessed: Jan. 10, 2023].

[19] Y. Khaireddin and Z. Chen, "Facial emotion recognition: State of the art performance on fer2013. arxiv 2021," *ArXiv Preprint arXiv:2105.03588*. doi: 10.48550/arXiv.2105.03588

[20] A. Mollahosseini, B. Hasani, M. J. Salvador, H. Abdollahi, D. Chan, and M. H. Mahoor, "Facial expression recognition from world wild web," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 58–65, 2016. doi: 10.1109/CVPRW.2016.188

[21] S. E. Kahou, X. Bouthillier, P. Lamblin, C. Gulcehre, V. Michalski, K. Konda, S. Jean, P. Froumenty, Y. Dauphin, N. Boulanger-Lewandowski, *et al.*, "Emonets: Multimodal deep learning approaches for emotion recognition in video," *Journal on Multimodal User Interfaces*, vol. 10, no. 2, pp. 99–111, 2016. doi: 10.1007/s12193-015-0195-2

[22] M. Z. Khan, S. Harous, S. U. Hassan, M. U. G. Khan, R. Iqbal, and S. Mumtaz, "Deep unified model for face recognition based on convolution neural network and edge computing," *IEEE Access*, vol. 7, pp. 72622–72633, 2019. doi: 10.1109/ACCESS.2019.2918275

[23] E. Kodhai, A. Pooveswari, P. Sharmila, and N. Ramiya, "Literature review on emotion recognition system," in *2020 International Conference on System, Computation, Automation and Networking (ICSCAN)*, pp. 1–4, IEEE, 2020. doi: 10.1109/ICSCAN49426.2020.9262389

[24] K. E. Jones, T. Fér, R. E. Schmickl, R. B. Dikow, V. A. Funk, S. Herrando-Moraira, P. R. Johnston, N. Kilian, C. M. Siniscalchi, A. Susanna, *et al.*, "An empirical assessment of a single family-wide hybrid capture locus set at multiple evolutionary timescales in asteraceae," *Applications in Plant Sciences*, vol. 7, no. 10, p. e11295, 2019. doi: 10.1002/aps3.11295

[25] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding facial expressions with gabor wavelets," in *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 200–205, IEEE, 1998. doi: 10.1109/AFGR.1998.670949

[26] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, *et al.*, "Challenges in representation learning: A report on three machine learning contests," in *International Conference on Neural Information Processing*, pp. 117–124, Springer, 2013. doi: 10.1007/978-3-642-42051-1_16

[27] P. Ekman and W. V. Friesen, "Constants across cultures in the face and emotion.," *Journal of Personality and Social Psychology*, vol. 17, no. 2, p. 124, 1971. doi: 10.1037/h0030377

[28] P. Ekman, "Strong evidence for universals in facial expressions: a reply to russell's mistaken critique.," *Psychological Bulletin*, 1994. doi: 10.1037/0033-2909.115.2.268

[29] H. Gunes and B. Schuller, "Categorical and dimensional affect analysis in continuous input: Current trends and future directions," *Image and Vision Computing*, vol. 31, no. 2, pp. 120–136, 2013. doi: 10.1016/j.imavis.2012.06.016

[30] C. Shan, S. Gong, and P. W. McOwan, "Facial expression recognition based on local binary patterns: A comprehensive study," *Image and Vision Computing*, vol. 27, no. 6, pp. 803–816, 2009. doi: 10.1016/j.imavis.2008.08.005

[31] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 915–928, 2007. doi: 10.1109/TPAMI.2007.1110

[32] R. Zhi, M. Flierl, Q. Ruan, and W. B. Kleijn, "Graph-preserving sparse nonnegative matrix factorization with application to facial expression recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 1, pp. 38–52, 2010. doi: 10.1109/TSMCB.2010.2044788

[33] L. Zhong, Q. Liu, P. Yang, J. Huang, and D. N. Metaxas, "Learning multiscale active facial patches for expression analysis," *IEEE Transactions on Cy-*

bernetics, vol. 45, no. 8, pp. 1499–1510, 2014. doi: 10.1109/TCYB.2014.2354351

[34] A. Dhall, R. Goecke, S. Ghosh, J. Joshi, J. Hoey, and T. Gedeon, "From individual to group-level emotion recognition: Emotiw 5.0," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pp. 524–528, 2017. doi: 10.1145/3136755.3143004

[35] Y. Guo, D. Tao, J. Yu, H. Xiong, Y. Li, and D. Tao, "Deep neural networks with relativity learning for facial expression recognition," in *2016 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pp. 1–6, IEEE, 2016. doi: 10.1109/ICMEW.2016.7574736

[36] Y. Huang, F. Chen, S. Lv, and X. Wang, "Facial expression recognition: A survey," *Symmetry*, vol. 11, no. 10, p. 1189, 2019. doi: 10.3390/sym11101189

[37] N. Samadiani, G. Huang, B. Cai, W. Luo, C.-H. Chi, Y. Xiang, and J. He, "A review on automatic facial expression recognition systems assisted by multimodal sensor data," *Sensors*, vol. 19, no. 8, p. 1863, 2019. doi: 10.3390/s19081863

[38] M.-I. Georgescu, R. T. Ionescu, and M. Popescu, "Local learning with deep and handcrafted features for facial expression recognition," *IEEE Access*, vol. 7, pp. 64827–64836, 2019. doi: 10.1109/ACCESS.2019.2917266

[39] S. Rajan, P. Chenniappan, S. Devaraj, and N. Madian, "Facial expression recognition techniques: a comprehensive survey," *IET Image Processing*, vol. 13, no. 7, pp. 1031–1040, 2019. doi: 10.1049/iet-ipr.2018.6647

[40] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, *et al.*, "Challenges in representation learning: A report on three machine learning contests," in *International Conference on Neural Information Processing*, pp. 117–124, Springer, 2013. doi: 10.1007/978-3-642-42051-1_16

[41] E. Barsoum, C. Zhang, C. C. Ferrer, and Z. Zhang, "Training deep networks for facial expression recognition with crowd-sourced label distribution," in *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pp. 279–283, 2016. doi: 10.1145/2993148.2993165

[42] A. Mollahosseini, B. Hasani, and M. H. Mahoor, "Affectnet: A database for facial expression, valence, and arousal computing in the wild," *IEEE Trans-*

actions on Affective Computing, vol. 10, no. 1, pp. 18–31, 2017. doi: 10.1109/TAFFC.2017.2740923

[43] H. Zhang, A. Jolfaei, and M. Alazab, "A face emotion recognition method using convolutional neural network and image edge computing," *IEEE Access*, vol. 7, pp. 159081–159089, 2019. doi: 10.1109/ACCESS.2019.2949741

[44] S. Minaee, M. Minaei, and A. Abdolrashidi, "Deep-emotion: Facial expression recognition using attentional convolutional network," *Sensors*, vol. 21, no. 9, p. 3046, 2021. doi: 10.3390/s21093046

[45] Y. Khaireddin and Z. Chen, "Facial emotion recognition: State of the art performance on fer2013," *ArXiv Preprint ArXiv:2105.03588*, 2021. doi: 10.48550/arXiv.2105.03588

[46] M. V. Valueva, N. Nagornov, P. A. Lyakhov, G. V. Valuev, and N. I. Chervyakov, "Application of the residue number system to reduce hardware costs of the convolutional neural network implementation," *Mathematics and Computers in Simulation*, vol. 177, pp. 232–243, 2020. doi: 10.1016/j.matcom.2020.04.031

[47] W. Zhang, K. Itoh, J. Tanida, and Y. Ichioka, "Parallel distributed processing model with local space-invariant interconnections and its optical architecture," *Applied Optics*, vol. 29, no. 32, pp. 4790–4797, 1990. doi: 10.1364/AO.29.004790

[48] C. Mouton, J. C. Myburgh, and M. H. Davel, "Stride and translation invariance in cnns," in *Southern African Conference for Artificial Intelligence Research*, pp. 267–281, Springer, 2021. doi: 10.1007/978-3-030-66151-9_17

[49] A. Van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," *Advances in Neural Information Processing Systems*. https://proceedings.neurips.cc/paper/2013/file/b3ba8f1bee1238a2f37603d90b58898d-Paper.pdf [accessed: Jan. 10, 2023].

[50] "Convolutional neural network." https://en.wikipedia.org/wiki/Convolutional_neural_network [accessed: Jan. 10, 2023].

[51] D. C. Ciresan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *Twenty-second international joint conference on artificial in-*

telligence, 2011. https://people.idsia.ch/~juergen/ijcai2011.pdf [accessed: Jan. 10, 2023].

[52] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3642–3649, 2012. doi: 10.1109/CVPR.2012.6248110

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

[54] J. He, L. Li, J. Xu, and C. Zheng, "Relu deep neural networks and linear finite elements," *ArXiv Preprint RrXiv:1807.03973*, 2018. doi: 10.48550/arXiv.1807.03973

[55] S. Raschka, "Why is the relu function not differentiable at x=0?," Jan 2021. https://sebastianraschka.com/faq/docs/relu-derivative.html [accessed: Jan. 10, 2023].

[56] M. Wang, S. Lu, D. Zhu, J. Lin, and Z. Wang, "A high-speed and low-complexity architecture for softmax function in deep learning," in *2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, pp. 223–226, IEEE, 2018. doi: 10.1109/APCCAS.2018.8605654

[57] D. Refaeli, "Sigmoid, softmax and their derivatives." https://themaverickmeerkat.com/2019-10-23-Softmax/ [accessed: Jan. 10, 2023].

[58] C. F. G. D. Santos and J. P. Papa, "Avoiding overfitting: A survey on regularization methods for convolutional neural networks," *ACM Computing Surveys (CSUR)*, vol. 54, no. 10s, pp. 1–25, 2022. doi: 10.1145/3510413

[59] P. Mahajan, "Max pooling," Jul 2020. https://poojamahajan5131.medium.com/max-pooling-210fc94c4f11 [accessed: Jan. 10, 2023].

[60] H. Lee and J. Song, "Introduction to convolutional neural network using keras; an understanding from a statistician," *Communications for Statistical Applications and Methods*, vol. 26, no. 6, pp. 591–610, 2019. doi: 10.29220/CSAM.2019.26.6.591

[61] V. Passricha and R. K. Aggarwal, "End-to-end acoustic modeling using convolutional neural networks," in *Intelligent Speech Signal Processing*, pp. 5–37, Elsevier, 2019. doi: 10.1016/B978-0-12-818130-0.00002-7

[62] G. Singhal, "Gaurav singhal," May 2020. https://www.pluralsight.com/guides/introduction-to-resnet [accessed: Jan. 10, 2023].

[63] Y. Zheng, C. Yang, and A. Merkulov, "Breast cancer screening using convolutional neural network and follow-up digital mammography," in *Computational Imaging III*, vol. 10669, p. 1066905, SPIE, 2018. doi: 10.1117/12.2304564

[64] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*, pp. 94–101, IEEE, 2010. doi: 10.1109/CVPRW.2010.5543262

[65] M. Valstar, M. Pantic, *et al.*, "Induced disgust, happiness and surprise: an addition to the mmi facial expression database," in *Proc. 3rd Intern. Workshop on Emotion (Satellite of LREC): Corpora for Research on Emotion and Affect*, p. 65, Paris, France., 2010. http://lrec.elra.info/proceedings/lrec2010/workshops/W24.pdf#page=73 [accessed: Jan. 10, 2023].

[66] G. Zhao, X. Huang, M. Taini, S. Z. Li, and M. PietikäInen, "Facial expression recognition from near-infrared videos," *Image and Vision Computing*, vol. 29, no. 9, pp. 607–619, 2011. doi: 10.1016/j.imavis.2011.07.002

[67] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding facial expressions with gabor wavelets," in *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 200–205, IEEE, 1998. doi: 10.1109/AFGR.1998.670949

[68] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, *et al.*, "Challenges in representation learning: A report on three machine learning contests," in *International Conference on Neural Information Processing*, pp. 117–124, Springer, 2013. doi: 10.1007/978-3-642-42051-1_16

[69] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-pie," *Image and Vision Computing*, vol. 28, no. 5, pp. 807–813, 2010. doi: 10.1016/j.imavis.2009.08.002

[70] C. Fabian Benitez-Quiroz, R. Srinivasan, and A. M. Martinez, "Emotionet: An accurate, real-time algorithm for the automatic annotation of a million facial expressions in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5562–5570, 2016. doi: 10.1109/CVPR.2016.600

[71] S. Li, W. Deng, and J. Du, "Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2852–2861, 2017. doi: 10.1109/TIP.2018.2868382

[72] A. Mollahosseini, B. Hasani, and M. H. Mahoor, "Affectnet: A database for facial expression, valence, and arousal computing in the wild," *IEEE Transactions on Affective Computing*, vol. 10, no. 1, pp. 18–31, 2017. doi: 10.1109/TAFFC.2017.2740923

[73] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "From facial expression recognition to interpersonal relation prediction," *International Journal of Computer Vision*, vol. 126, no. 5, pp. 550–569, 2018. doi: 10.1007/s11263-017-1055-1

[74] S. Li and W. Deng, "Deep facial expression recognition: A survey," *IEEE transactions on affective computing*, 2020. doi: 10.1109/TAFFC.2020.2981446

[75] G. Verma and H. Verma, "Hybrid-deep learning model for emotion recognition using facial expressions," *The Review of Socionetwork Strategies*, vol. 14, no. 2, pp. 171–180, 2020. doi: 10.1007/s12626-020-00061-6

[76] L. Wang, S. Xu, X. Wang, and Q. Zhu, "Eavesdrop the composition proportion of training labels in federated learning," *ArXiv Preprint ArXiv:1910.06044*, 2019. doi: 10.48550/arXiv.1910.06044

[77] "Data processing." https://en.wikipedia.org/wiki/Data_processing [accessed: Jan. 10, 2023].

[78] "Welcome to pytorch tutorials." https://pytorch.org/tutorials/ [accessed: Jan. 10, 2023].

[79] "Pytorch." https://en.wikipedia.org/wiki/PyTorch [accessed: Jan. 10, 2023].

[80] M. Sambare, "Fer-2013," Jul 2020. https://www.kaggle.com/datasets/msambare/fer2013 [accessed: Jan. 10, 2023].

[81] G. Singhal, "Gaurav singhal," May 2020. https://www.pluralsight.com/guides/introduction-to-resnet [accessed: Jan. 10, 2023].

[82] A. R. Khan, "Facial emotion recognition using conventional machine learning and deep learning methods: Current achievements, analysis and remaining challenges," *Information*, vol. 13, no. 6, p. 268, 2022. doi: 10.3390/info13060268