Faculty of Informatics and Computer Science

Artificial Intelligence

Automatic Human Emotion Recognition System using Facial Expressions with Deep Learning Approaches

By: Mohamed Adel Ismail

ID:197908

Supervised By: Professor Amr S. Ghoneim

November 2022

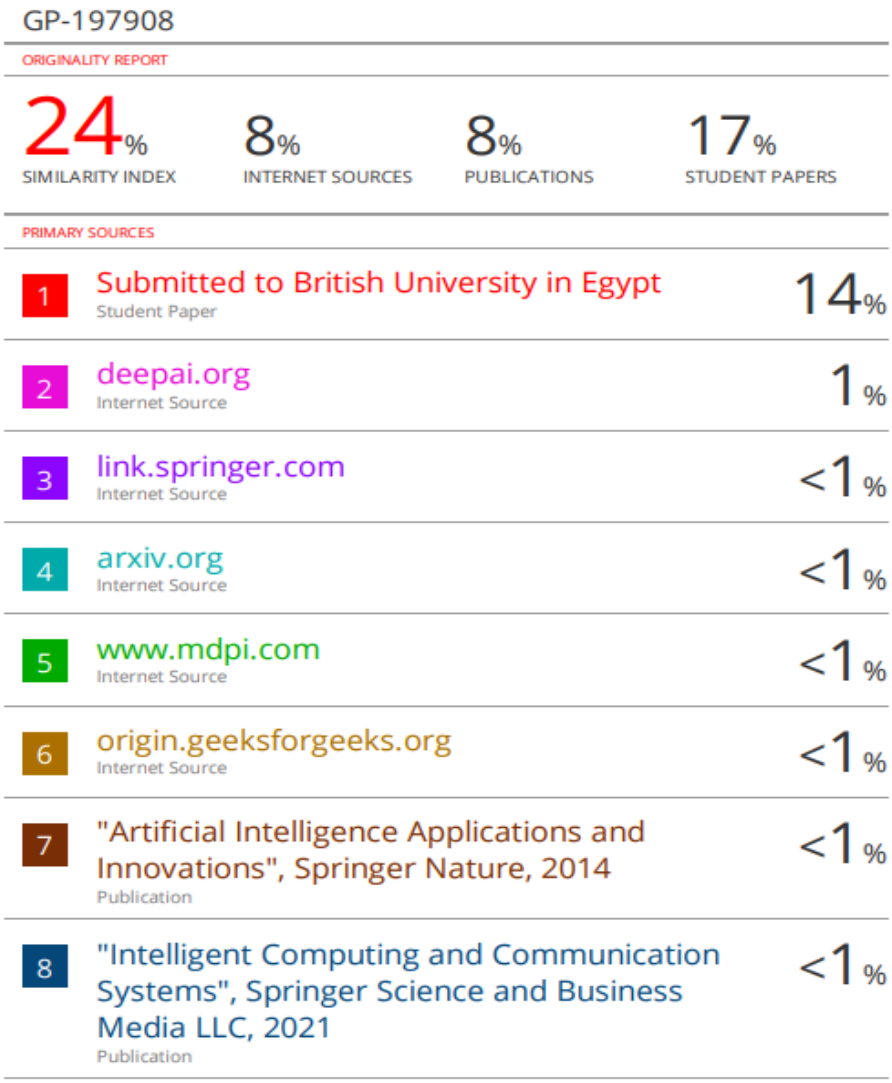
Attestation &Turnitin Report

I understand the nature of plagiarism and am aware of the university's policy on this.

I certify that this report reports original work by me during my university project, except for the following:

- Figure 1: from the University of Missouri.
- Figure 4: referenced from a website link.
- Figure 7: referenced from Prof. Pester's Lectures

Signature: Mohamed Adel Ismail Date 25/11/2022



turnitin

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	mohamed 197908
Assignment title:	Graduation project report-AI Part 1 (Moodle TT)
Submission title:	GP-197908
File name:	297_mohamed_197908_GP-197908_161976_1154393584.docx
File size:	1.59M
Page count:	34
Word count:	12,024
Character count:	66,447
Submission date:	12-Jun-2023 04:24AM (UTC+0200)
Submission ID:	2114049355

The BRITISH UNIVERSITY IN EGYPT

Faculty of Informatics and Computer Science

Artificial Intelligence

Automatic Human Emotion Recognition System using Facial Expressions with Deep Learning Approaches

By: Mohamed Adel Ismail

ID:197908

Supervised By: Professor Amr S. Ghoneim

November 2022

Acknowledgments

I want to acknowledge that Dr. Amr S. Ghoneim is one of the most professional doctors I have encountered at the University. This project was created with his complete guidance and support. I wish to influence others the same way he affects me to push myself and exceed my limits.

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Background & Overview

Humans communicate with each other using different forms. Over the centuries, distinct cultures have created languages to speak and express their feelings. However, humans can express their feelings in unusual ways, and spoken words are one way that is easily interpreted as sound. Another form used as a way of expressing feelings is facial emotions. Regardless of the language a person is speaking, face emotions are shared between people of every kind and are considered as the uniform non-verbal language of humans. Human emotions have been researched over the years and exists different emotional models [2]-[7]. These models suggest various categories of emotions and then classify every emotion based on the model's characteristics. Emotions play a significant role in everyday conversations [4]. Unlike verbal words, non-verbal reactions *(facial emotions)* happen cognitively and naturally. Understanding human emotional behavior is considered a competitive research area in several fields of study, like psychology, criminal assessment, gaming development, and customer satisfaction. These fields collect and interpret facial expressions to understand human emotions and behaviors. In today's world, everything is analyzed for better understanding. Whether the analysis happens on structured or unstructured data, the advances in hardware allow people to explore and train machines using a dataset and predict outputs upon the trained model. This work examines conducted studies on the FER-2013 to create a benchmark result from all previous work employing a single-network model architecture.

This study aims to create the best-fitting model for real-time emotion detection and recognition for the F.E.R. dataset.

1.2 Problems & Challenges

Facial Emotion Recognition *(F*.E.R.) was challenged by the ICML in 2013. The challenge was designed for competitors to develop the best system for recognizing facial emotion from a picture [1]. In this contest, the FER-2013 dataset was collected by Pierre Luc Carrier and Aaron Courville for this specified task [1]. This dataset was collected using Google Image search API for images that included faces that matched a set of 184 emotion-related keywords. Goodfellow reported that the developed final dataset has a human accuracy of 65±5% [1]. Because of the intra-class variations and the imbalanced nature of the emotions, the FER-2013 dataset is more challenging than other facial expression datasets [1]. Therefore, proving that the FER-2013 is a complex dataset which cannot be easily learned.

However, although humans can easily detect emotions naturally with little to no effort, this simple task remains challenging for computers [2]. Humans can interpret these characteristics easily because of our mind's ability to perceive and understand complex data. When a computer tries to do the same, it faces different difficulties related to the image's pixel resolution. Other lighting conditions also affect the photos. These problems are tackled by image enhancement techniques that result in a normalized dataset for easier feature extraction. However, images included in the dataset are pre-processed and normalized into the same size. The need for collecting these emotions enables a company to connect with its customers' honest opinions deeply. Leading to the analysis of the overall

customer experience to increase the overall performance of a company and reveal business insights. For example, a real-time facial emotion recognition security application would not allow customers to withdraw money from an A.T.M. when the model predicts that this customer is threatened or in fear [5]. Another use case would be for the gaming industry, in which emotion detection software can adapt to each level's difficulty based on the gamer's perceived emotions [5]. Moreover, An example of a surveillance application is monitoring patients who have trouble speaking based on their emotional state [2].

1.3 Aims and Objective

This study will act as a benchmark analysis for different Artificial Neural Network models from past research papers. These papers will be heavily examined and analyzed from various points that include pre/post-processing techniques, chosen model architecture, and chosen hyperparameters to understand the features collected by the network entirely. However, all these papers share the same dataset and do not add auxiliary data for training the model, thus, creating a challenge. Regarding the approach of this work, images contain many features; hence, regular machine learning algorithms will not be efficient if used as classifiers. Therefore, we will be using Deep learning approaches in this study. Therefore, experimented models may include probabilistic Deep Belief Networks *(i.e., Restricted Boltzmann machines),* regular Convolutional Neural Networks, Recurrent Neural Networks (*i.e., LSTM*), and Hybrid model networks *(i.e., CNN-RNN).* In this research, we aim to improve the classification accuracy of 7 different emotions stated by Ekman emotion model using a newly proposed single-network model that achieves higher accuracy than the state-of-the-art models.

In this research, our objectives are:

- Create benchmark analysis for related work focusing on what researchers missed.
- Identify the best pre-processing technique for the dataset using E.D.A.
- Find a best-fitting single network model that achieves higher accuracy than state-of-the-art models.
- Tuning the final model hyper-parameters for achieving higher results.
- Create a robust face emotion recognition model that classifies emotions into seven categories.

1.1 Report Organization

This study is structured as follows: Chapter One introduces different background topics, an overview of the study, and research objectives. The Background Topics part of chapter two introduces the emotion Recognition Model. Three deep learning models are discussed in chapter three with details for brief knowledge. Chapter four of the Literature Review provides an overview of related research focusing on studies that use a single network architecture. The Implementation Details & Computational Setup section describes the environment and tools used for this study. The Dataset Exploration section describes the dataset utilized in the research, while the section on Exploratory Data Analysis displays hidden findings within the F.E.R. dataset. The Experiments section contains the study's findings and separate experiments performed using the CNN-LSTM model and the Vision Transformer - ViT model, which contains experiments with pre-processing techniques.

1.2 Work Methodology

In this project, the work methodology used will be the waterfall methodology. This methodology follows a sequential development process, meaning a phase should is completed before moving to the next phase. A waterfall methodology consists of five separate stages, as shown in Figure 4.

1. Gathering requirements for further reading and understanding of the topic approached.
2. Designing the project workflow is subdivided into a logical and physical design phase.
3. Implementation of the proposed algorithm.
4. Testing the proposed algorithm.
5. Maintaining the entire system and updating when necessary.

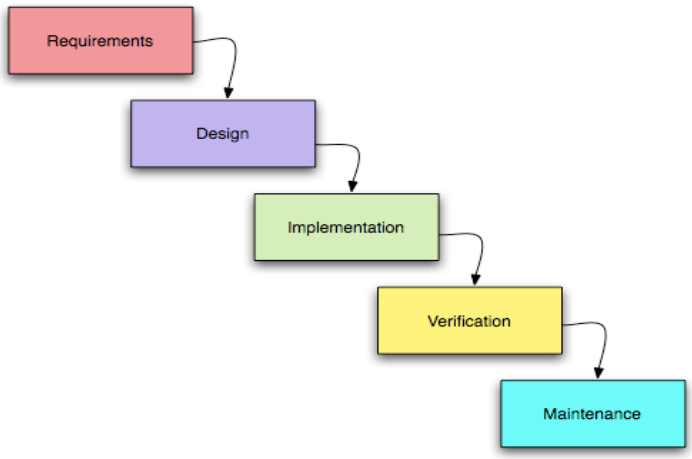


Figure 1: Work methodology

1.3 Work Plan (Gant Chart)

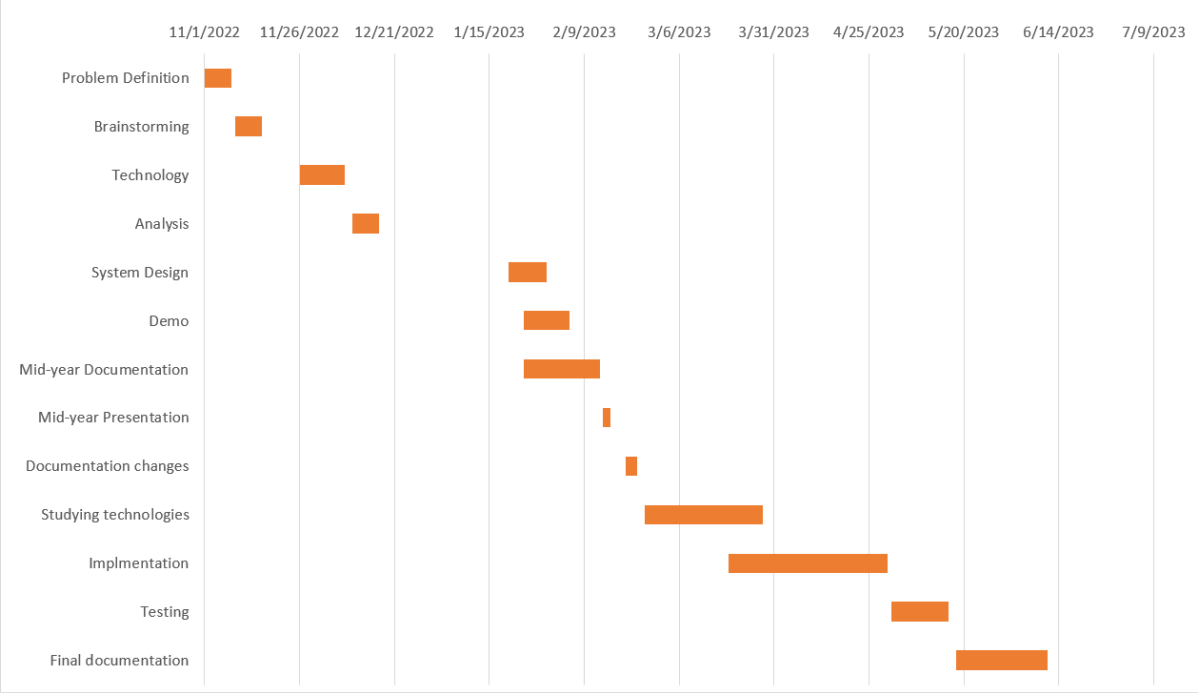


Figure 2: Work Plan

2. Background Topics

In this section, we focus on introducing background topics needed to understand the report further. These background topics explain different emotion models used in their related studies. Moreover, another essential background topic is the different deep learning models, which ensures that a reader has a basic understanding of how these deep learning models work.

2.1 Emotion Recognition Model

When trying to recognize the facial emotion of a person, humans look at the other person's face. They predict one's emotional state based on what they perceive from face muscle contractions. This simple task for the human brain can be done in seconds. However, it could get complicated for a computer to extract notable features representing a person's mood. Because a computer understands the image as a collection of pixels, it is hard to tell a computer what a tree looks like. Computer vision is the science in which computers learn to derive key features or objects from data *(i.e., images, videos)*. When it comes to face recognition, there are two techniques by which a machine can extract features of the face. The first method is geometric-based feature extraction. In this type of feature extractor, the computer is concerned with modeling the face *(e.g., mouth, eye, nose)* with respect to their location from each other. However, another common type of feature extractor is an appearance-based extractor which understands images or faces as objects made of pixels. After receiving the image as a matrix of pixels, the computer iterates over the image using a matrix filter. When both matrices are multiplied, they result in a new matrix with an image's key features. Research has shown that in human communication, 55% of all information

perceived from others is perceived from facial expressions [9]. This proves the importance of facial emotion recognition. In [7], Ekman produced an emotional model in which this model suggested that all humans share six basic emotions: anger, surprise, happiness, fear, sadness, and disgust. Regarding our application in business analytics, this emotion model will be chosen for the classification of faces into one of the six classes. Moreover, another neutral class will be added to the list of emotions, making them 7 emotional categories for classification.

3. Deep Learning Models

3.1 Deep Belief Network

Unlike single layered perceptron that focuses on solving linear problems and consists of an input layer, a single hidden layer, and an output layer. Deep Belief Nets (DBN) were created for more complex neural network structures. The main improvement of the DBN is that it focuses on extracting more features. When the neural net was first invented, it was created to act like the neurons of the human brain. The human brain consists of billions of attached neurons that happen to exchange information using axons which are represented as the weights in neural nets. Therefore, the idea of DBN came to life using the same concept of connected neurons. A deep belief network is composed of a single-layer neural network, with more hidden layers that focus on collecting information and passing this information to the second layer of hidden units. Figure 2 visually illustrates the structure of DBN. The training of this type of network happens within two phases which are called: the forward phase and the backward phase. In the forward phase, random weights are given to the network, and the activation function is chosen. Then, the network produces an output that is compared to the actual results before the start of the second phase. Finally, the backward phase is the place where the weights of the system are updated by calculating the rate of actual–predicted outputs.

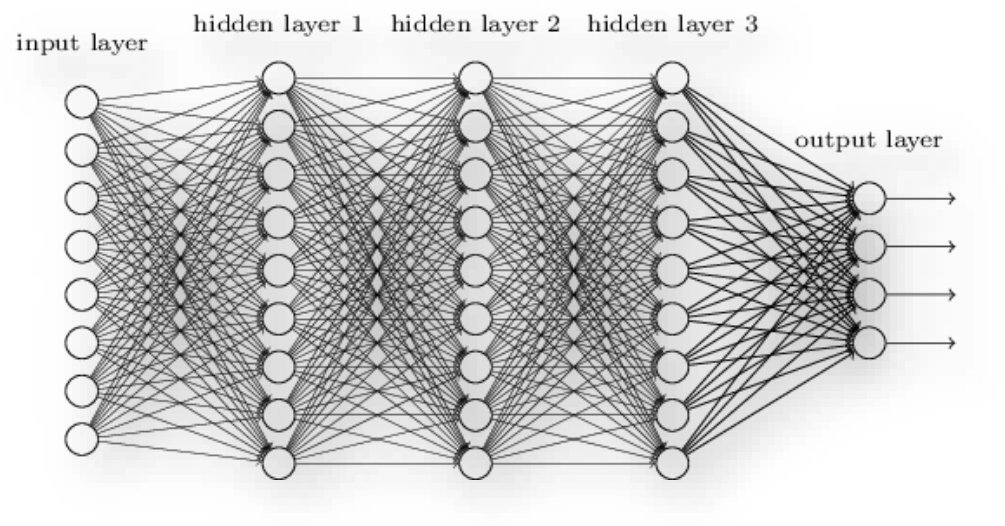


Figure 3: DBN structure. Visible, hidden, and output layers.

3.2 Convolutional Neural Network

A convolutional neural network is considered one of the most exploding algorithms in the field of computer vision [5]. This type of neural net is concerned with feature extraction and image classification [1]. The main difference between a regular ANN (artificial neural network) compared to a CNN (Convolutional Neural Network) is that it can understand a relation between nearby pixels. Therefore it achieves higher accuracy when working with unlabelled data. The CNN architecture is created in a way that makes it easier for the model to interpret the key features of an image. CNN is composed of a convolution layer, a max-pooling layer, and fully connected layers. The feature extraction process takes place in the convolution layer, where a matrix that works as a feature detector (Kernel) iterates over the input image and results in a feature map. The second layer is the pooling layer, in which the image dimensionality is reduced to focus on more detailed features [3]. When working with CNN, there are two types of images that we can work with, a 1-dimensional black and white image and a 3-dimensional RGB image. Because an RGB has three dimensions that will be processed by the conv layer, an RGB image requires more computational resources than a B.W. image. Moreover, when a computer receives an image, it is interpreted as a matrix of intensities where every intensity is in the range from 0 to 255. However, whether the image is RGB or B.W., CNN works the same way. First, a model receives the input image as a matrix. Second, a convolution layer (also referred to as the ReLU layer) is applied to the image to extract key features. After that, a max or average pooling layer is applied to reduce the dimensionality. In the fully connected layer, the final Feature map is flattened and considered as the input of the ANN. Figure 3 shows an illustration of CNN architecture. In this type of neural network, the model is trained using backpropagation by calculating the gradient descent with respect to the given weights.

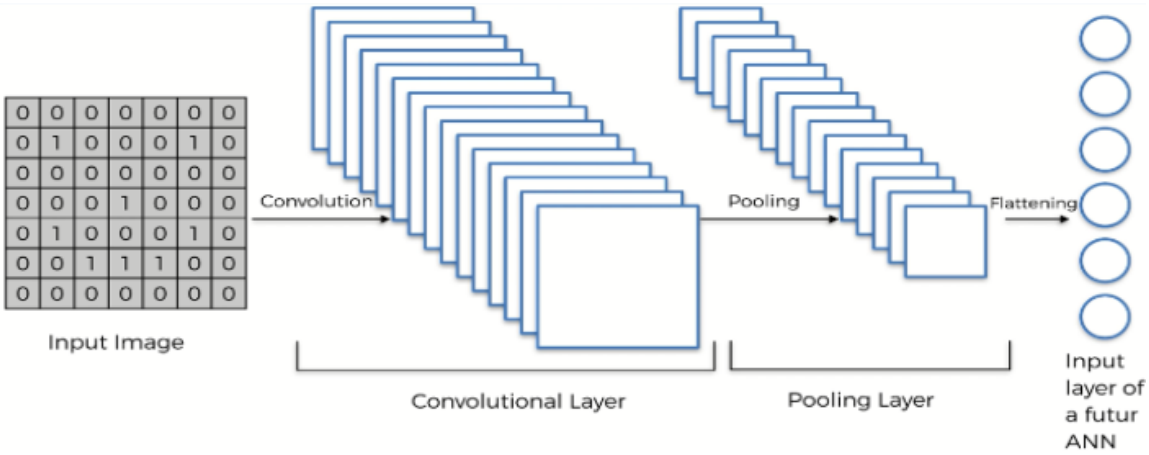


Figure 4: Convolution Neural Network Structure.

3.3 Recurrent Neural Network

A recurrent neural network (R.N.N.) is well-known in sequential data analysis and natural language processing. This family of neural networks excels at tasks like language modeling and temporal prediction because they are primarily concerned with capturing dependencies in sequential data. The ability of an R.N.N. to keep a hidden state, which enables it to recall past inputs and account for the order of data points in a sequence, is the essential difference between an R.N.N. and a standard convolutional neural network. Using specialized layers called

recurrent layers, the R.N.N. architecture is designed to make it easier to handle sequential input. These layers include recurrent units that may store information over numerous time steps, such as Long Short-Term Memory (LSTM) cells or Gated Recurrent Units (G.R.U.s). The hidden state of an R.N.N. is updated at each time step based on the current input and the previous hidden state as input data is processed sequentially. The output produced at each time step depends on the hidden state present at that moment. Therefore, an R.N.N. is a great model, especially in cases where the data include dependencies of each other, like time series problems.

4. Literature Review

This chapter reviews the relevant literature on the report's topic. It discusses the findings of previous studies and identifies the gaps in knowledge. In this chapter, we are concerned only with papers on the FER-2013 dataset that did not include additional data. Hence, it is more challenging to create the best-fitting model without the use of auxiliary data. The focus of the literature review is to shed light on single-network models to classify emotions. By applying these rules, the dataset challenge becomes more challenging.

Saravanan et al. conducted a survey that focused on building a best-fitting neural network from scratch to classify facial emotions [5]. The author was concerned with classifying facial emotions into Ekman's seven emotional states [7]. In his survey, the author built three neural network models that included a feed-forward neural network and two CNN networks composed of different architectures. The final model architecture included a total of 6 conv layers. The networks start with two convolutional layers, each with a 64-filter size, followed by a max pooling layer and a dropout of 0.25 to reduce overfitting. The second convolutional block consists of two conv layers with 128 filter sizes each. The final conv block included two convolution layers using a filter size of 256 followed by max-pooling, 0.25 dropout, and a fully connected layer that uses the SoftMax activation function for classification. The dataset did not include any pre-processing, and it was split into 80%-20% for training and testing, respectively. The author proposed model achieved a 55% accuracy before tuning. After tuning the hyperparameters using Adam as an optimizer and batch size 128, the model showed improved performance reaching an accuracy of 60.5%.

In [4], the author proposed a deep learning approach that considers the use of an attentional mechanism. Using a saliency map, the author can focus on and capture important parts of the face. The saliency map is generated by determining the importance of each pixel when multiplied by zero. Therefore, when zeroing an important pixel, the accuracy of the input image is changed. As mentioned, the authors proposed model consists of two main parts: a feature extraction and a localization network, i.e., the attentional mechanism that focuses on important face regions when identifying certain emotions. This model was tested on different facial emotion datasets, including JAFFE, CK+, FERG, and FER-2013. However, this research scope is only concerned with experiments done on the FER-2013 dataset. The proposed architecture is composed of four convolutional layers where each two is followed by a max pooling layer and has a ReLU activation function. Then, they are followed by a dropout layer and two fully connected layers with a SoftMax activation for predicting the output. The second part of the network is the attentional mechanism is composed of two convolutional layers, each followed by a max-pooling and ReLU activation function. Then, the network applies affine transformation to wrap the localization layer input to the output and finalize the network with a fully connected layer. The author visualizes notable features captured by the saliency map, revealing the essential parts of each emotion to understand the network better. Finally, the

author's final model was trained, validated, and tested on a dataset split of 60%, 20%, and 20%, respectively achieving an absolute accuracy of 70.02%.

In [17], the author Khaireddin et. al. has demonstrated a model that uses V.G.G. network and tunes for the purpose of classifying emotions. The model created achieved an accuracy of 73.2% on the testing set recording the highest single-network accuracy on the F.E.R. dataset. In his paper, the author mentions several papers that included auxiliary data but refuses to do so when training his model, focusing on creating a model with no additional data. Furthermore, for visualization, the author created a saliency map to further understand what the network sees as important parts of an image. However, in the pre-processing, the author uses data augmentation techniques, including rotating, flipping, and rescaling the dataset images. This will increase the training data of the model without creating bias. For the model training, the model epochs were set to 300, and categorical cross-entropy as the loss function. For the optimizers, the author has chosen different optimizers and then compared them all. These optimizers included S.G.D., S.G.D. with Nesterov momentum, Average S.G.D., Adam with A.M.S. Grad, Adadelta, and Adagrad. For simplicity, the highest performance within these optimizers was acquired by the S.G.D. with Nesterov momentum along with the Reduce Learning Rate on Plateau callback. The model's final accuracy achieved is 73.28% for the validation set.

In the research referenced [2], the author is concerned with using predefined architectures aLexNet and GoogLeNet, and comparing them against state-of-the-art architectures. The author mentions that he will be using the F.E.R. dataset hence it is a widely used benchmark for evaluating experiments of facial emotion recognition. First, the author explains GoogLeNet's architecture which consists of 22 deep-layer networks. This model architecture was first introduced in the original GoogLeNet in 2014 [18]. The total 22 is after counting layers that have trainable weights and biases (i.e., Fully connected layer, pooling layer). The network has nine layers or modules that are called the "Inception layer" executed in parallel [2]. The inception layer applies a 1x1 convolution first to reduce the number of input channels, followed by more extensive convolutions (3x3 and 5x5) applied in parallel. This saves on computational costs by the use of 1x1 convolutions to reduce dimensionality before applying more expensive operations. Therefore, GoogLeNet architecture uses 12 times fewer parameters than AlexNet. The difference in parameter count is found because inception modules do not use fully connected layers. Instead, they use an average pooling layer that transforms the final layer into the output layer without the need for F.C.L. Figure 7 shows the GoogLeNet architecture. On the other hand, AlexNet architecture was created for use in a competition in 2012. The network submitted at the ILSVRC contest consisted of 8 fully connected layers. The first layer of AlexNet is a convolutional layer. These layers learn local features from the input images and introduce non-linearity using the ReLU activation function. The max pooling layers reduce the size of the feature maps, which helps to speed up the training process and prevent the network from overfitting. The fully connected layer classifies the input images into one of 1000 classes. The author managed to train the two models with the CPU; he also mentions a lot of hyper-parameters for each of the architectures.

To conclude his findings, we will mention the final accuracy of the Ekman emotional model, which classifies emotions into seven categories, matching our problem, after training both models for more than 4000 epochs.

AlexNet achieved an absolute accuracy of 65%, while GoogLeNet achieved a slightly better performance of 65.5%.

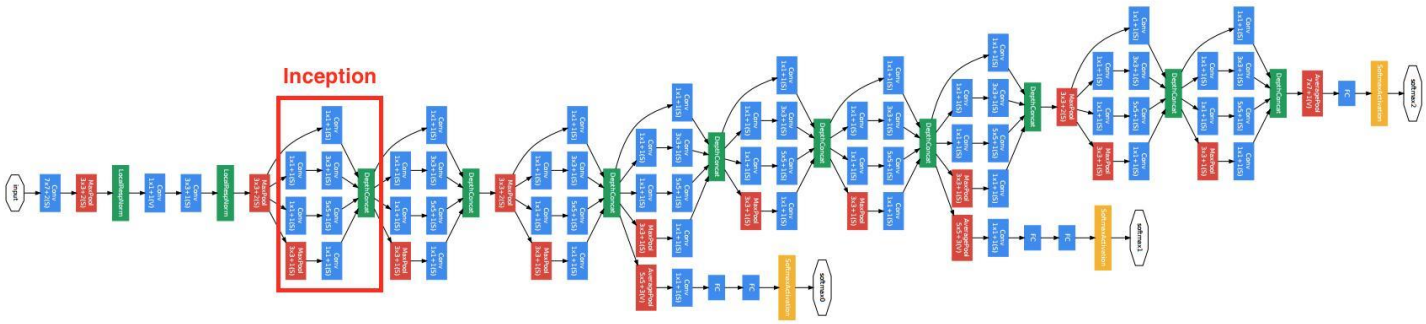


Figure 5: GoogLeNet architecture. Inception module.

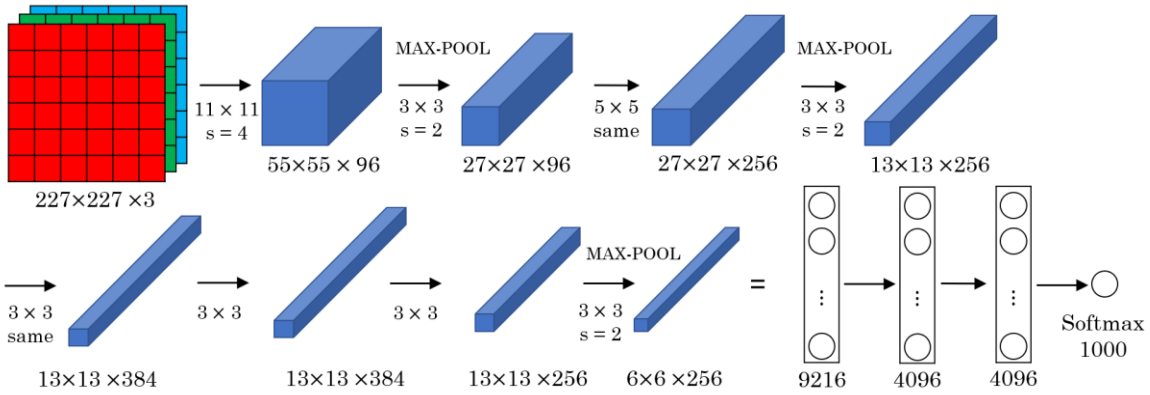


Figure 6: AlexNet architecture.

The research titled "Real-time facial emotion recognition using Deep Learning" focuses on analyzing face emotions into seven categories mentioned in Ekman's six emotional states along with a neutral state [3]-[7]. In this research, the author uses the FER-2013 dataset to build a recognition model without any pre-processing of the dataset. However, the author is concerned with creating a real-time face detector. This implementation requires face detection from an input frame using CV2 library. The captured feed is then fed into a face extraction network that uses the Harr Cascade algorithm to detect a face from an input image. Then the extracted image is resized to match the trained model's target size and fed into the model for classification. This research uses the predefined model Xcpetion by Google [6]. The xception model is an improved version of the Inception-V3 model. The model comprises four separable residual depth-wise convolutions, followed by a batch normalization and a ReLu activation function after each convolutional layer. The last layer has an average pooling layer instead of a fully connected layer to help reduce dimensionality with a SoftMax activation to classify each input to one of seven emotions. Figure 6 shows the Xception model architecture. The author also mentions using stochastic gradient

descent as the optimizer, categorical cross-entropy as a loss function, and a learning rate 0.0001. The training happened over 100 epochs with a batch size of 64, achieving a final validation accuracy of 68.57% [3].

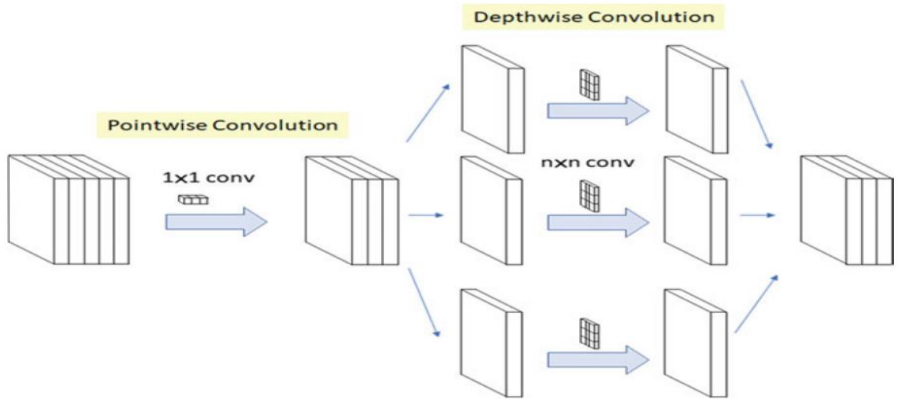


Figure 7: Xception model applying pointwise then a depthwise convolution for feature extraction [3].

The authors in a paper titled "Facial Expression Recognition Using Convolutional Neural Networks: State of the Art" study using convolutional neural networks (CNNs) for facial expression recognition [19]. The authors compare the performance of various CNN designs on the FER2013 dataset, which serves as a standard for facial expression recognition. They discover that a deep CNN architecture with residual connections performs best, with a test accuracy of 72.7%. In addition, the authors chose the best-performing models from all available benchmarks. V.G.G., Inception, and ResNet are the three CNN architectures compared by the authors. They describe each network's design, including the number of layers, parameters, and accuracy on the FER2013 dataset. The V.G.G. architecture is similar to VGG-B but with significant tweaks, the Inception architecture is based on GoogLeNet with consistent feature map sizes, and the ResNet architecture is identical to a related work of 34-layer ResNet but with fewer parameters and dropout. The authors emphasize that despite being deeper, V.G.G. and Inception have fewer parameters than architectures utilized in related studies. The V.G.G. architecture outperformed prior CNN architectures in test accuracy, with a score of 72.7%. The authors also discovered that employing a pre-trained model and augmenting the data can increase the performance of CNNs for facial emotion identification. The process of increasing the size of the training dataset using synthetic techniques to create new images from existing images is known as data augmentation. This can help to improve CNN performance by making them more resistant to changes in position and illumination. A pre-trained model can increase CNN performance by giving a starting point for Learning.

Title	Year	Objective of the study	Dataset type	Split, Pre-processing	Classifier Details	Accuracy
Facial Emotion recognition using Convolutional Neural Networks [5]	2019	Classify human faces into 7 face emotions	CSV	80-20 split	10-layers CNN	60.5%
Deep learning approaches for facial emotion recognition: A case study on FER-2013 [2]	2018	Examining performance of Deep learning approaches of facial expression recognition	JPG	75-25 split	AlexNet architecture	65%
Deep learning approaches for facial emotion recognition: A case study on FER-2013 [2]	2018	Examining performance of Deep learning approaches of facial expression recognition	JPG	Not mentioned	GoogLeNet architecture.	65.2%
Going Deeper in Facial Expression Recognition using Deep Neural Networks [16]	2015	Proposes a deep neural network architecture to address the FER-2013 dataset problem trying to achieve stat-of-the-art performance.	JPG	Not mentioned	Two convolutions each followed by max pooling, then four inception layers.	66.4%
Real-Time facial emotion recognition using deep learning [3]	2021	Analyse different emotions represented by the human face in real time.	CSV	Not mentioned	Xception model, implement pointwise conv followed by depthwise conv.	68.57%
Deep-Emotion: Facial Expression Recognition using attentional convolutional network [4]	2019	Propose a deep learning approach based on attentional CNN	JPG	Data Augmentation	CNN + attentional mechanism	70.02%
Facial expression recognition using convolutional neural networks: state of the art performance on FER2013 [19]	2016	Experiments aim to overcome the limitations of shallow and basic CNN architectures commonly used in FER.	JPG	Data Augmentation	Inception architecture	71.60%
Deep Learning using Linear Support vector Machines [20]	2015	Examines the effects of using SVM as an activation function for the last layer.	JPG	No	CNN using SVM as activation function.	71.2%
Facial Expression Recognition Using Convolutional Neural Networks: State of the Art [19]	2016	Experiments aim to overcome the limitations of shallow and basic CNN architectures commonly used in FER.	JPG	Data Augmentation	ResNet Architecture	72.4%
Facial Emotion Recognition: State of the Art Performance on FER2013 [17] [19] VGG – 71.7%	2021	Examining the performance of single-network research done on the FER-2013, comparing models created from scratch with predefined models.	JPG	Yes	VGG architecture	73.2%

Table 1: A summary of the related work Employing Single network architectures.

5. Implementation Details & Computational Setup

We implemented various machine learning models in the experiments using Python, a versatile and widely used programming language in data science and artificial intelligence. We leveraged multiple popular libraries, such as TensorFlow and PyTorch, for designing, training, and evaluating our models. These libraries offer a comprehensive ecosystem for deep Learning, providing extensive functionality and optimization for efficient model training and deployment. For pre-processing and data manipulation, we used libraries like Pandas and NumPy, which offer powerful capabilities for handling large datasets and performing complex numerical operations. To visualize our results, we employed Matplotlib and Seaborn, enabling the creation of informative plots and graphs that aid in interpreting our findings. Our computational setup consisted of high-performance hardware, including modern GPUs with extensive parallel processing capabilities, which significantly accelerated the training and evaluation of our deep learning models. We ran our experiments on a Collaboratory equipped with GPUs, taking advantage of the GPU acceleration features provided by TensorFlow and PyTorch. The operating system used for the experiments was Windows distribution, which offered an Intel(R) Core(T.M.) i7-8750H CPU @ 2.20GHz 2.21 GHz. This device provided performance and compatibility with the required tools and libraries. All relevant configuration details, such as hyperparameters, optimizer settings, Tensor Board evaluation, and network architectures, were carefully documented to ensure the reproducibility and reliability of our experiments.

6. Dataset Exploration

6.1 Dataset

The FER-2013 dataset is a facial expression recognition dataset that was released in 2013 as a challenge by Google. The dataset consists of 30,000 grayscale images of faces, each with a corresponding label indicating the emotion being expressed. The feelings that are represented in the dataset are Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral. This dataset was created using Google API's engine collecting images that correlate to emotion-like states using 182 keywords [1]. This dataset contains 35887 images of different aged people, including children and adults. Therefore, the dataset has no specific pose or a generic technique for the pictures collected. The dataset was published at the ICML 2013 challenge [2]. Where the images were pre-processed to ensure that they were of high quality and that they were all the same size, but no other characteristics were unified. The FER-2013 dataset is a popular benchmark dataset for facial expression recognition. It has been used to evaluate a variety of different facial expression recognition methods. Images in the dataset are all pre-processed in the same way. The dataset is available as images and as a . CSV file. According to the pre-processing steps chosen, the appropriate dataset will be used. However, the CSV file is better used for E.D.A. hence it only needs a little pre-processing like the image's dataset.

- The images in the dataset are 48x48 pixels in size.
- The images are grayscale.
- The dataset is split into a training set of 28,709 images and a test set of 3,589 images.
- The images within the dataset include different poses and different occlusions *(i.e., hand, glasses, mic)*.
- The images within the dataset appear to have a tremendous intra-class similarity.
- The dataset is available in .csv and .jpg format.

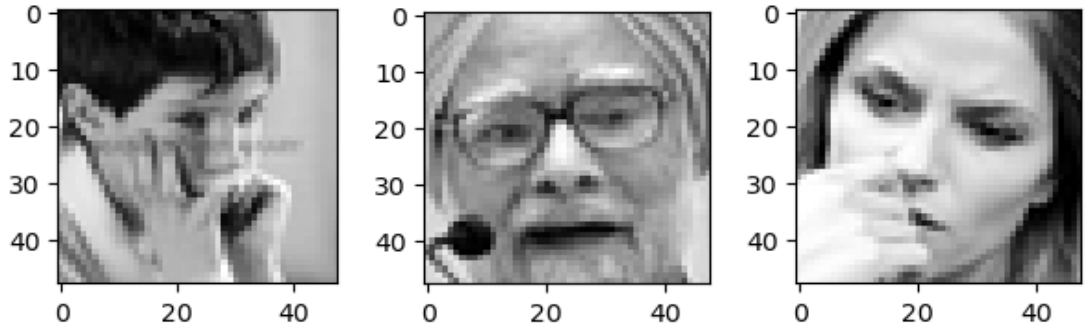


Figure 8: samples of different occluded images within the F.E.R. dataset.

7. Exploratory Data Analysis

The pre-processing of the F.E.R. dataset is one of the crucial steps in creating a best-fitting model, particularly for this challenge. Moreover, all past researchers appear to give this challenge a code-driven approach, meaning that they do not focus on the dataset as much as they focus on building a complex model for the dataset. A model's training data is the knowledge of a model's network. Good training data eventually leads to a model with high generalization and low bias; these characteristics indicate a future of a model that can classify correctly. Therefore, in this section, we will investigate problems within the F.E.R. dataset, then figure out ways to solve the issues found. Our goal is to explore the dataset using E.D.A. techniques, then comes the pre-processing, where the goal is to overcome challenges encountered in the dataset using data science and machine learning techniques.

7.1 Exploring for outliers

A simple function for identifying the outliers creates a box-and-whisker plot using the average pixel values for the whole pixels' column. This box plot helps show the distribution of the pixel values across the dataset, showing the five-number summary of data, including min, max, lower quartile, and upper quartile, as shown in Figure 9. These numbers help decide whether the average pixel value is considered an outlier. However, for this to happen, a simple function was created to compare the average pixel value of each image to a threshold value and then decide if the image is considered an outlier. The outcome of the function returned samples is shown in Figure 10.

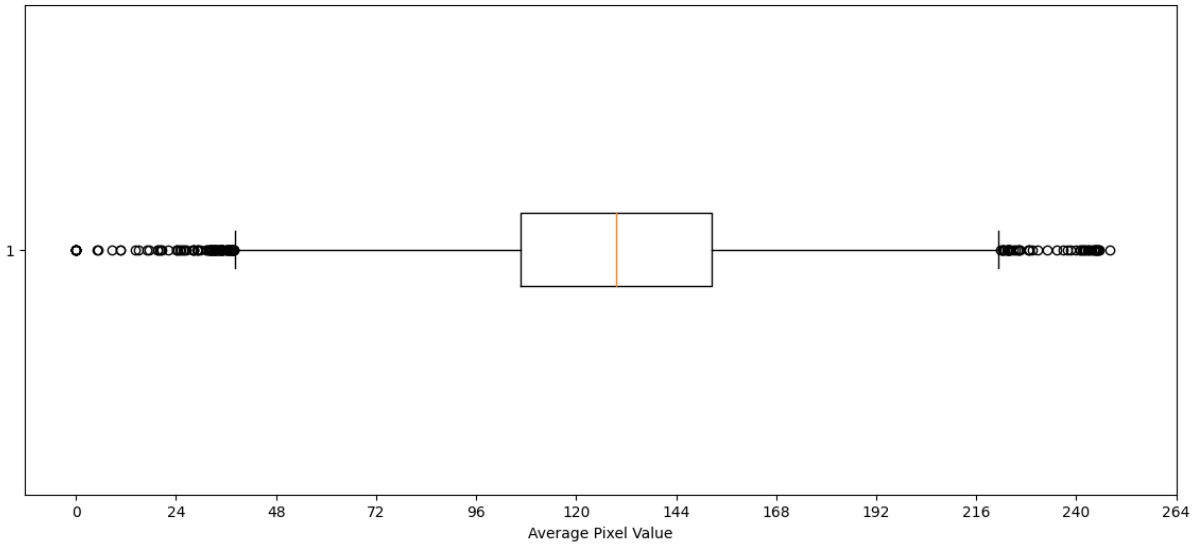


Figure 9: Box-and-whisker plot showing the average distribution of pixel values.

The samples we found were images that belong to cropped images of text or figures. These images severely affect model training because training a model with values extracted from these images will only mislead the model when classifying to one of the emotions. The outliers are collected by a simple function that compares the average pixel count of images to a certain threshold, and if the value is greater than the threshold value, the pixel values are appended into a list to be dropped from the original dataset.

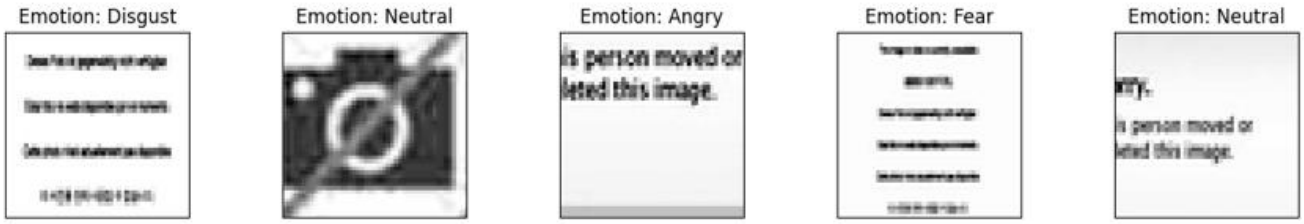


Figure 10: Instances of misleading Outliers within the dataset.

7.2 Duplicates with different emotions

Another major problem found in the dataset that was discovered in the E.D.A. phase is that there are duplicates within the dataset. But duplicate effects on a model can vary in some cases; it could benefit a model's training in others, it does not. Keras pre-processing library includes Image data generator, which applies techniques close to duplicating. It duplicates an image by rotating or flipping the image therefore resulting in an image that is repeated in the dataset with a factor of change. Figure 12 shows the number of duplicates in each class, along with a total number of instances. All instances of duplicates were not dropped from the dataset, instead a simple function was created to iterate over all duplicates and return copies with non-matching emotions. We are trying to find if there exist duplicates of an image with different assigned emotions. To investigate if these non-matching duplicates increase the bias of a model. Surprisingly, 282 images were found that were duplicates with non-matching emotions. Hence, these duplicates are investigated in depth, finding exactly 160 duplicated images that vary in the number of duplications. For instance, a single image was found repeated four times in the dataset, mentioned as happy once and surprise three times in the dataset. However, these duplicates will affect the model's training negatively. Therefore all duplicates with non-matching emotions were dropped.

Figure 11: instances of images that have duplicates with non-matching emotions.

7.3 Imbalanced Classes

One of the problems within the F.E.R. dataset is the imbalanced classes. This imbalance in the samples of the classes leads to a model which classifies the majority class correctly but fails to classify minority classes due to a lack of samples. To overcome this, several machine learning techniques are used—for example, Random sampling and SMOTE from imblearn. Different methods, like Keras's Image Data Generator library, are used to process images. Figure 8 shows the number of instances within each class. The visualization shows that happy emotion is the majority class, and disgust is the minority class. Therefore, using this dataset without considering the class imbalances will negatively affect the model's training. For example, the weights of a model will adjust more to the instances of happy images than disgust simply because it has seen more cases of happiness than disgust.

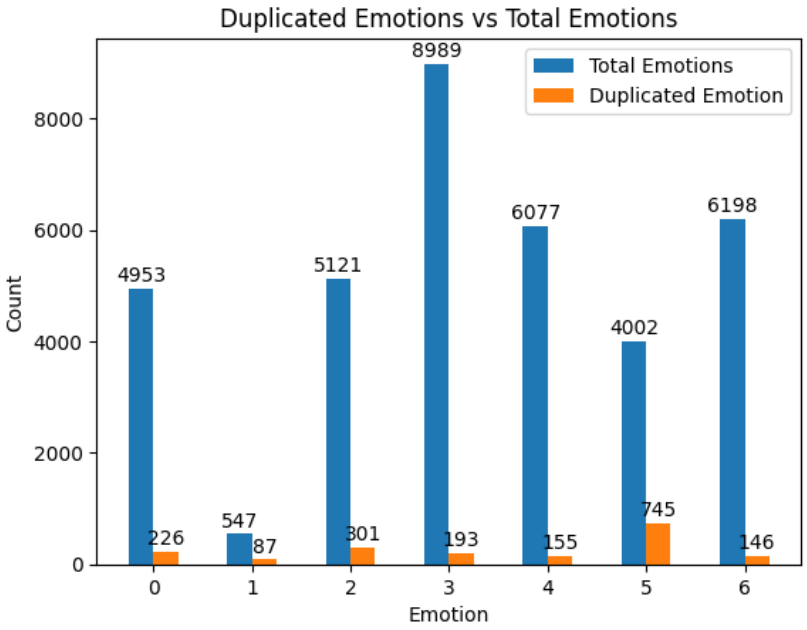


Figure 12: Total number of instances and the total number of duplicates for each of the emotions.

For pre-processing the imbalanced classes in the Fer-2013 datasets, several methods can be applied:

- The first method used to prepare the training and test images is ImageDataGenerator or Augmentation techniques. To improve the data given for a model, techniques such as random rotation, width and height changes, rescaling, and horizontal and vertical flipping are included in this. The rescale option normalizes the pixel intensity by scaling all pixel values by 1/255. Grayscale versions of the photographs are also created. As a result, this method is used to handle the dataset's images. It seeks to expand the amount and diversity of the training data through various augmentations, as well as normalizing the input data to enhance the neural network's overall performance.
- RandomOverSampler is the second technique used to balance the classes by oversampling minority classes to address the class imbalance issue. The pixel values are normalized by dividing all values by 255 to scale them between 0 and 1. The pixel data is then reshaped into a 4D array suitable for the CNN model. The emotion labels are converted to a 2D array and then one-hot encoded. Finally, the data is split into train and test sets, and the one-hot encoded labels are returned along with the pre-processed pixel data arrays. This pre-

processing technique focuses on balancing the data, normalizing the features, and transforming the data into the appropriate format for model training.

- SMOTE (Synthetic Minority Over-sampling Technique) is a sampling technique used to address class imbalance in a dataset. The SMOTE technique generates synthetic samples by interpolating between existing minority class samples. Specifically, for each minority class sample, SMOTE selects k-nearest minority class neighbors and creates synthetic samples by randomly sampling points along the line segments connecting the minority class sample to its k nearest minority class neighbors. This results in a larger and more diverse minority class, which can improve the performance of machine learning models.

During our experiments, all methods are going to be used and evaluated. Therefore, a brief comparison of these techniques is introduced. However, imblearn sampling techniques are preferred over data augmentation because of the variability and stability of their use. Random Oversampling and SMOTE preserve the original distribution of the samples and do not introduce any artificial variations. Data augmentation might generate fewer representative samples due to the applied transformations. Oversampling provides a consistent training set throughout the training.

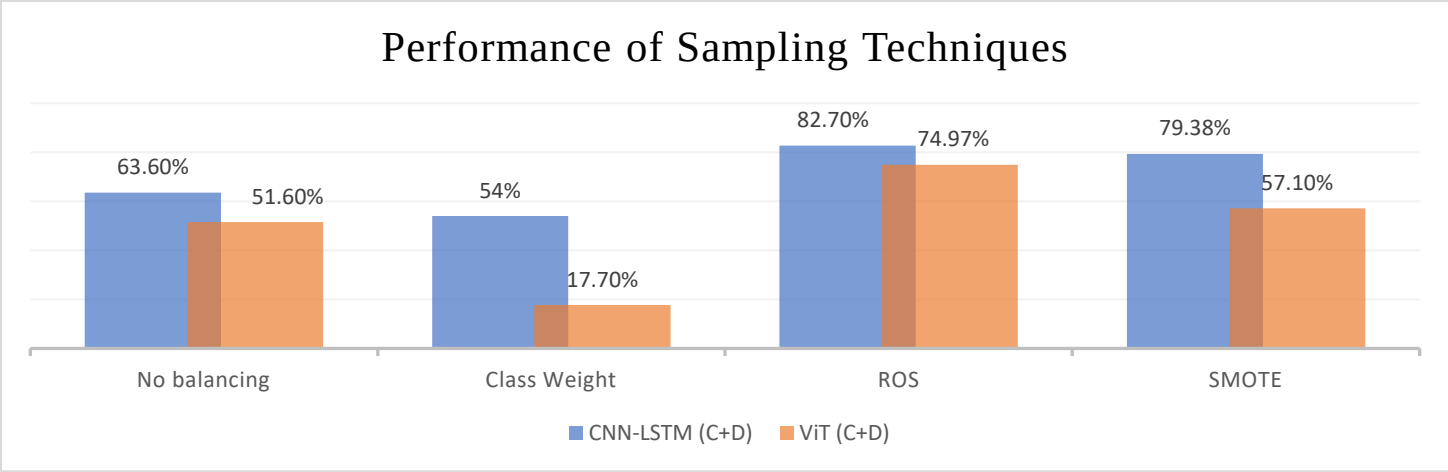


Figure 13: Our proposed model's performance using different sampling techniques used in related work.

8. Experiments

8.1 CNN-LSTM

A. Model intuition

The proposed model architecture employs a hybrid CNN-LSTM scheme for facial expression recognition. First, two convolutional blocks are used for spatial feature extraction. The first block contains two convolutional layers with 32 and 64 filters, respectively, followed by batch normalization and rectified linear unit (ReLU) activation. The second block has two convolutional layers with 64 and 128 filters, respectively, followed by batch normalization and ReLU activation. Max pooling layers are utilized after each convolutional block to down-sample the feature maps. These convolutional blocks somehow act like the attentional mechanism found in reference [4]. Next, the extracted high-level spatial features from the CNNs (Convolutional Neural Networks) are flattened and fed into an LSTM layer with 128 units to capture temporal dependencies. An additional LSTM layer with 64 units further models the temporal information. Two fully connected layers with 200 and 7 units, respectively, act as a classifier to assign expression probabilities to the input sequence. The first fully connected layer employs ReLU

activation, while the final layer utilizes SoftMax activation for multi-class classification into seven expression categories. In addition, a dropout layer is inserted between the two fully connected layers to reduce overfitting. This architecture was chosen because it leverages convolutional layers to extract spatial features from input images while utilizing LSTM layers to model the temporal information within the spatial features. The model's total trainable parameter is 457,031, which is relatively small compared to other research models. This model is appropriate because its hierarchical convolutional blocks and the increasing number of filters enable it to learn representations at multiple scales, improving expression discrimination. The hybrid CNN-LSTM architecture combines the strengths of both models for facial expression sequence recognition. The hybrid model, in our case is helpful since the Fer-2013 dataset contains high rates of intra-class similarity. Unlike machine learning, deep learning models are prone to overfitting, which is a big problem in the DL field. Using this CNN-LSTM hybrid model can have significant drawbacks if the model cannot generalize well to the data. Instead, the model starts to memorize the weights instead of creating a good hypothesis to classify correctly. To overcome this problem, different techniques like callbacks and dropouts are used.

The CNN-LSTM design is divided into five variations. Each variation is created to show the improvements in E.D.A. and data pre-processing which all researchers did not acknowledge. Variation 1 is a regular model that does not include any of the pre-processing mentioned in Chapter 7 E.D.A. Secondly, variation 2 is created to show the impact of the duplicated image on the model performance. Thirdly, variation 3 shows what data-driven questions lead to. Variation 4 shows the effects of outliers on the model training performance. Finally, a final variation five is created to deliver everything together, offering a final model with the best generalization to the cleaned dataset.

For the first time, this model architecture includes a hybrid approach of CNN and LSTM. The goal was to test the capability of the hybrid model to see how well the model can differentiate between the seven emotions. For this experiment, no pre-processing was done. This way, we know how the model is performing against state-of-the-art models. As we mentioned in dataset section 3.1, there are two types of datasets CSV and images folder. For this experiment, we chose to work with the CSV file for easier interpretation of the nulls, duplicates and to be able to see the five-number summary and many more calculations. Hence, the CSV file was more appropriate. At first, when looking at duplicates in the dataset, we found a total of 1853 instances that are duplicated. For starters, these duplicates were thought to be of the same emotion. However, duplicates can improve the dataset by adding training samples to the dataset, which happens either way by ImageDatagenerator or by Oversampling techniques. However, in this variation of the CNN & LSTM model, duplicates were not dropped for further investigation on how they affect the model. Figure 14 shows duplicates of the same emotion (Angry).

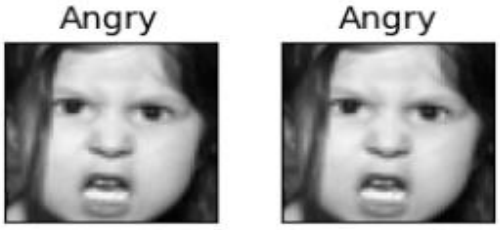


Figure 14: instances of duplicates found in the dataset.

B. Pre-processing & model callbacks

Pre-processing of this model was done by function ***pre_proc_data***, which receives an input of the data frame that includes target column data["emotion"], and feature columns data["pixels"]. The function first extracts pixel data and emotion labels from the input data frame. Then the function applies RandomOverSampling techniques by using the appropriate conversions for the pixel column to correctly interpret the pixel values, which are represented as strings and then converted into float datatype. Next, the pixel values are normalized by dividing all instances by 255. Finally, the x_data and y_data variables that hold values of target and features are then reshaped to 4D and 2D arrays, respectively. The function returns four variables, x_train, y_train, x_test, and y_test, with the test split being 0.1 for testing. For the model training, callbacks were defined to avoid overfitting. The callbacks included ReduceLROnPlateau, which helps with reducing the learning rate by monitoring val_loss, setting factor to reduce the learning rate by 0.1, setting min_delta to 0.0001, which refers to the smallest value for the learning rate, and we have set the patience to 1, which specifies the number of epochs to wait before triggering the learning rate reduction. EarlyStopping was also introduced to the model with patience 8 for this model, while the number was decreased for other models. Moreover, tensor board was also introduced to show the accurate plots of the training/validation accuracy and loss plots. Finally, a checkpoint is added to the callback to save the best checkpoint as a .h file, which can be loaded separately to test the model's accuracy.

C. Model Evaluation

For all models, the evaluation is the same in which model accuracy and loss plots are saved in the corresponding file of the model. In addition, a confusion matrix and classification report are created to show the different classification errors that the model fell for. Lastly, a saliency map highlights the important part of an image that the model is gaining information from. The experiment was done using the Keras library. The model training epoch was set to 100, but the early stopping was there to stop the model from overfitting. This exact model was trained for a total of 16 epochs. They reached a final validation accuracy of 78.3 % and 0.691 loss. As shown in Figure 15, evaluation metrics for the model include confusion matrix and accuracy.

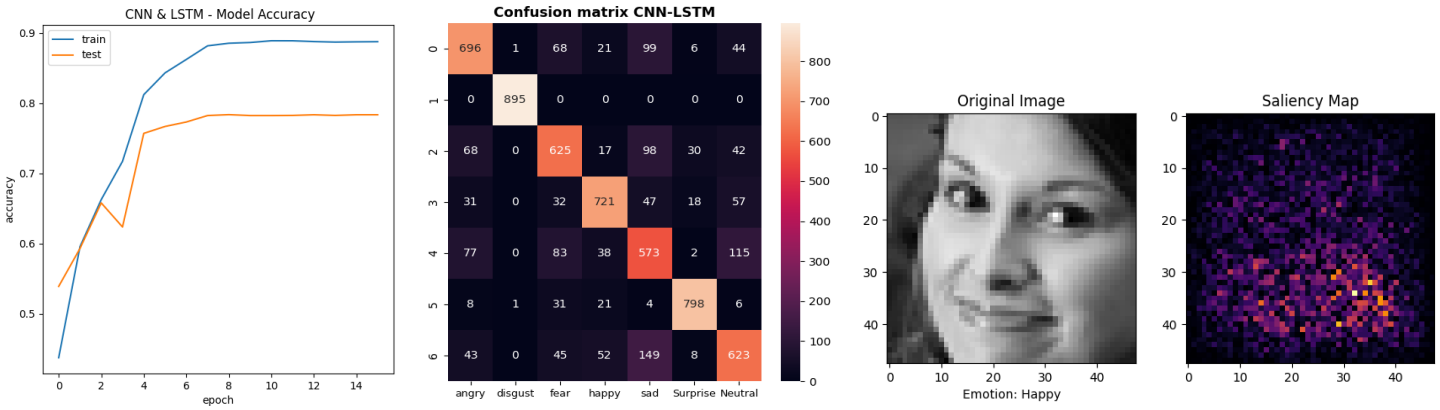


Figure 15:Evaluation metrics for CNN-LSTM Variation 1 model.

8.2 Duplicates Removed (CNN-LSTM)

A. Pre-processing and Model Evaluation

For this variation, the same model was created, but the pre-processing included another step: removing all duplicates found in the dataset. A total of 1853 duplicates were removed from the original dataset, which contained

35887 instances for all training and testing. The resulting dataset had a total of 34034 cases after dropping duplicates. The variation duplicates removed were created to show the difference between the regular model generalization capabilities and this model. The results were astonishing after the model had trained for the same 16 epochs and stopped by an early stopping callback to avoid overfitting. The overall training of the model showed much smoother curves for the accuracy and loss plots. The final accuracy achieved by this model increased by 2.3% while the loss decreased by 0.06, reaching an absolute accuracy and loss of 80.68%, 0.697. The model's generalization performance was better than the previous variation in classifying fear, sadness, and surprise emotions. The duplicate instances in each emotion are 301, 155, and 745, respectively. On the other hand, angry, happy, and neutral showed a loss in correctly classified instances compared to the previous model variation.

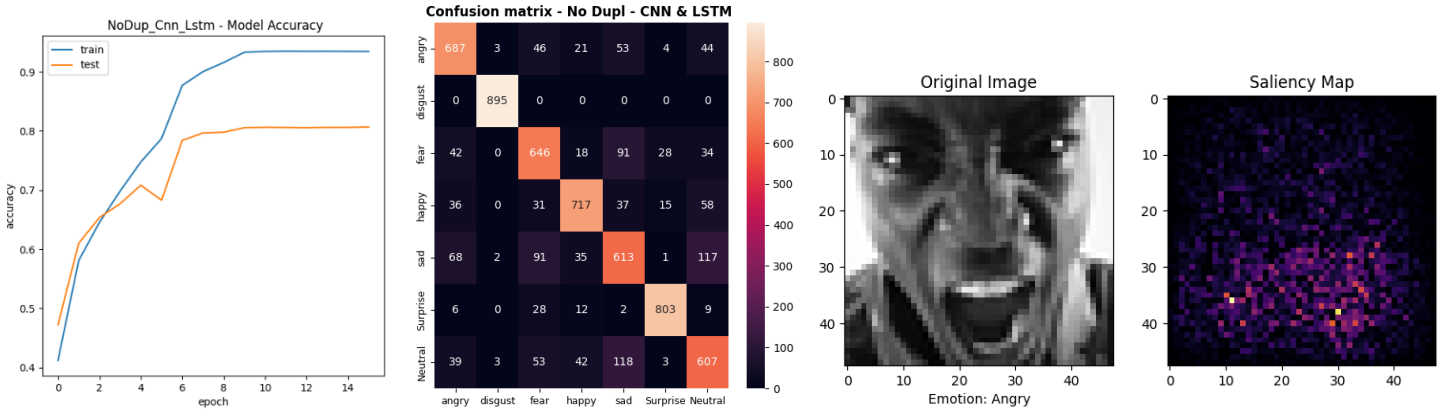


Figure 16:Evaluation metrics for CNN-LSTM Variation 2 model.

8.3 CNN-LSTM (Non-matching Dup Removed)

A. Data-driven insights

After removing duplicates, the overall performance of the model suggests that duplicates did negatively affect the training performance, showing that they were holding the model from a better generalization rate. But the reason behind the effect of these duplicates needed to be clarified. Therefore, more analysis on the duplicate data frame was done, focusing on showing the bad side of these duplicates. Data is the most important factor in machine learning and deep Learning, good training data results in a model with a high ability to generalize to unseen data. Therefore, asking the right questions will lead to more data-driven insights. Hence, one example of these duplicates negatively affecting the model training is finding duplicate images with non-matching emotion labels. This irregularity could lead to a model with high bias because it has seen the same image pixel values once as fear and once as sad, as Figure 12 suggests. Therefore, this led us to the non-matching duplicate variation of the same CNN-LSTM.

B. Pre-processing (Non-matching Dup Removed)

For the pre-processing of this model variation, the only additional step was to remove non-matching duplicates from the dataset. This was done by the function ***duplicates_non_match_emotion,*** which iterates over the original dataset returning the duplicates with non-matching emotions from the total 1853 duplicates. This function compares the emotion label for each duplicate image appending the duplicates with non-match emotions. As mentioned earlier, the scope of this research was to find the best-fitting single model for the F.E.R. dataset, but until now, the objective has changed into a data-centric view. Showing the different anomalies in the dataset and

cleaning it for better performance and generalization using the same hybrid model consisting of CNN-LSTM. The total number of non-matching duplicates is 282 images, where each image can be found twice or more. The total number of instances dropped as non-matching emotion was 160. Figure 12 shows examples of these emotions with non-matching emotions.

C. Model Evaluation

Variation 3 of the hybrid model showed increased accuracy and correctly classified emotions. Compared to the two previously created models, this variation showed an increase in correctly classified instances for angry, disgust, happy, surprise, and neutral emotions. While on the other hand, the model had failed to generalize emotions of classes fear, and sad. This way shows an improvement in the overall accuracy and loss. The model achieved an accuracy of 81.56% higher than the variation 2 model by a 0.88 % increase in accuracy. Moreover, the model loss decreased from 0.697 to 0.641 with a total 0.056 loss improvement. However, as we can see from the confusion matrix, the model is biased between fear and sadness because these emotions show significant intra-class similarity.

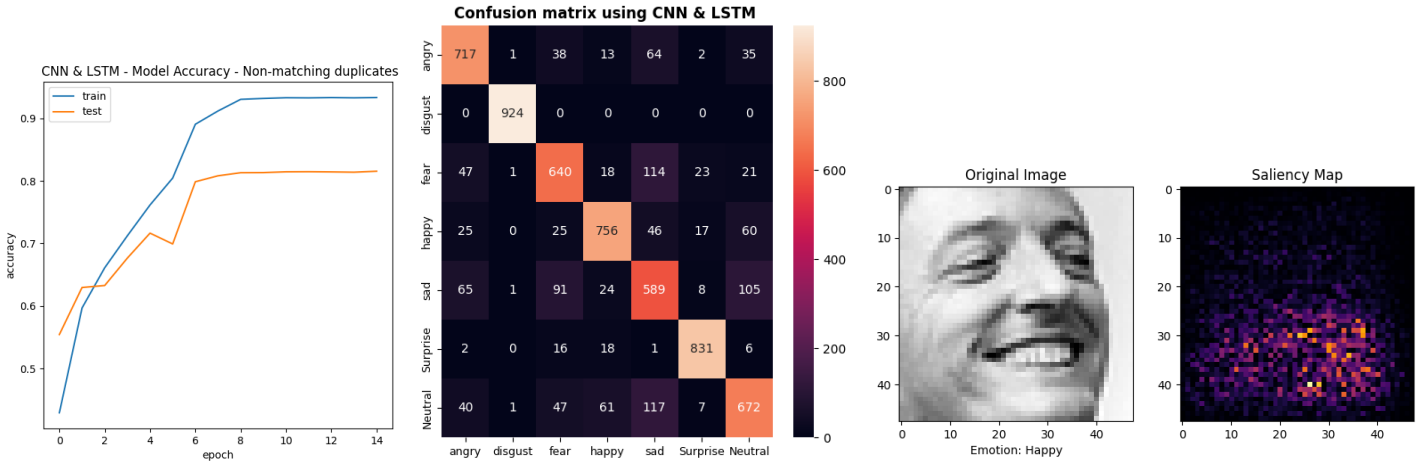


Figure 17: Evaluation metrics for CNN-LSTM Variation 3 model.

8.4 CNN-LSTM (Outliers Removed)

A. Pre-processing (Outliers)

Another variation of the model was created, but this time to check for outliers and check the outlier's effects on the model's ability to learn from the F.E.R. dataset. As mentioned before, the F.E.R. dataset was collected using Google's API searching for faces that match 184 emotion-related keywords [1]. *(i.e., these keywords contained words like "Blissful," "Enraged," etc.)* Therefore, the dataset can have specific images that do not belong to any of the emotions. Therefore, our approach to this challenge remains a data-centric approach. We should focus more on the data, not only the model architecture. Hence, to find outliers within the FER-2013 dataset, we created a function to show a box-and-whisker plot for the average pixel values of each image. This box plot shows us the distribution of the pixels of all images, as Figure 11 shows. Creating this simple function helped us reveal the five-number summary for the whole dataset. Hence, we could recognize an appropriate threshold for the outlying images. These outliers were then collected by another function that returns samples of outliers, as shown in Figure 10. The resulting outliers were images that were part of a figure or text that did not correspond to the emotions.

B. Model Evaluation

Removing the outliers from the dataset should be beneficial to the model training. The model accuracy should be higher than the last variation of the model. While the model created without duplicates of non-matching emotion reached an accuracy of 81.5%, generalizing well in the new data without misleading images. This model achieves higher results, reaching an accuracy of 82.3%. This shows that these images negatively affected the model and needed removal.

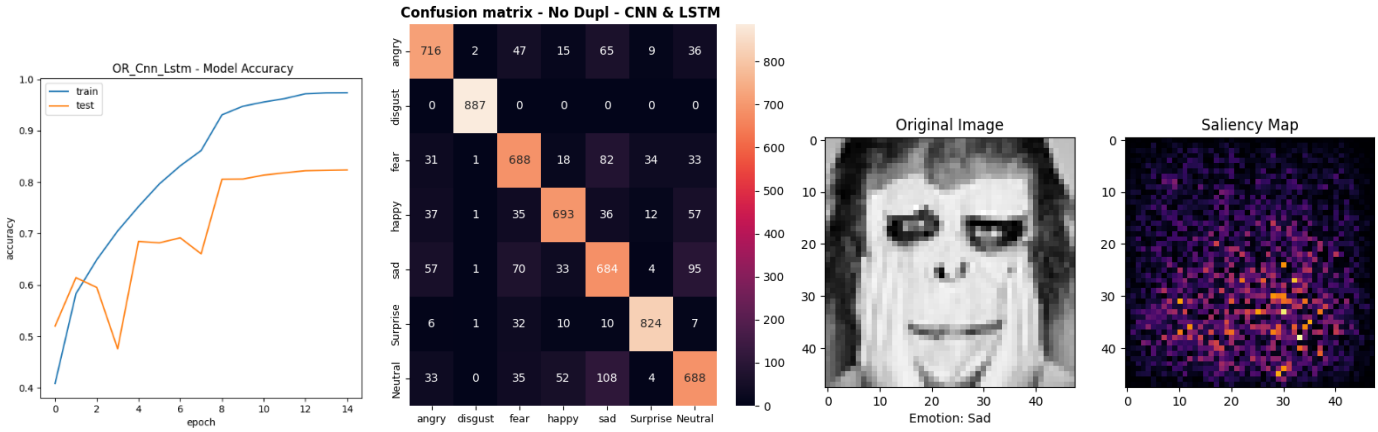


Figure 18: Evaluation metrics for CNN-LSTM Variation 4 model.

8.5 CNN-LSTM Final

A. Data Cleaning *(putting it all together)*

For this model, all of the above pre-processing techniques were done before the model training. First, exploratory data analysis was done again to ensure that everything was present. A total of 35887 images is found in the original dataset. These images are divided into three usage cases. Training which includes 28709 images, private test which includes 3589 images, and public test which also includes 3589. When exploring the dataset for duplicates, there were found 1853 duplicated images distributed across the 7 emotions. Among these duplicates were 160 duplicates with non-matching emotions that were dropped using the function remove_duplicates_with_diff_emotion(). After that, using the box-and-whisker plot, we found 28 misleading outliers that represent figures or values that do not correspond to human emotion. These outlier instances were dropped using the function drop_outlier_images(). Leaving us with a total of 35699 images within our dataset.

B. Data pre-processing

To prepare our data for the model, we first applied oversampling to the data and converted it to the correct format. Then we normalized the data by dividing pixel values by 255. The features data were reshaped to meet up with the height and width of the original image of size (48*48). Moreover, the target variable holds values from 0 to 6 for emotions "Angry," "Disgust," "Fear," "Happy," "Sad," "Surprise," and "Neutral," respectively. These values were converted into one-hot encoded vectors to meet up with the model requirements. For this specific model, the dataset was split using the train_test_split function from keras with test_size set to 0.2.

C. Model architecture

For the final model architecture, the model is the same as before but with some minor changes in the model's hyper-parameters. First, our model accepted input of 48*48*1 according to our image size and its grayscale dimensions. Second, a 32-filters Conv2d layer is applied to the data with stride (1,1) and valid padding, meaning that no padding is added. All Conv2d blocks are followed by a batch normalization layer to improve convergence

during training and a Relu activation function to add non-linearity. The second and third blocks in the architecture both have a Conv2d layer with 64 filters, stride (1,1), with valid padding for block two and the same padding for block three. However, to reduce overfitting, the first 64 filters block contains a Maxpooling layer to minimize dimensions and aggregate features and followed by a dropout rate of 0.1. After these two 64 blocks are two 128 filters Conv2d layer blocks, with the first one having both max pooling and a dropout rate of 0.1, and the second one only having a max pooling layer to downscale the feature maps by taking the maximum value in a 2x2 window. Therefore, taking the most prominent feature in each region of the (2,2) window.

The LSTM layers are defined after the fifth Conv2d layer. The first Reshape layer reshapes the input to (-1, 128) to match the LSTM (128) layer, followed by a dropout of 0.3 to avoid overfitting from the LSTM temporal sequences. After that is another LSTM layer that includes 64 neurons, to finalize the model architecture, Two fully connected layers with 200 and 7 units, respectively, act as a classifier to assign expression probabilities to the input sequence using a softmax activation function. However, the first dense layer has a dropout of 0.3 to avoid LSTM's overfitting the data. The total number of trainable parameters is 457,031. This number of parameters is way less than other researchers using pre-trained models and complex model architectures. This model proves that a complex model cannot consistently achieve higher results than a simple one. Instead, building a deep learning model is a craft with many variables to consider.

D. Hyper-parameters and Fine tuning *(avoiding overfitting)*

The final hybrid model includes all pre-processing of the other variations, including removing the outliers, removing duplicates with non-match emotion, and applying the appropriate data pre-processing to match the data with the model's input layer for training. For this specific model, the callback reduced learning rate was set to monitor the val_loss with a factor of 0.1, and the patience was set to 2 before lowering the learning rate. The early stopping patience parameter was reduced from 8 to 5. Hence the model should wait for 5 epochs before stopping the model's training. The model optimizer used is Adam, and the learning rate value was set to 0.002 instead of 0.0002. Lastly, the batch_size parameter was also selected to 64 to reduce the batches in each training epoch to 685. The result of all the pre-processing steps and the fine-tuning of the model led to the highest performance of a single network model that is able to generalize well to the unseen data and classify correctly. Figure 18 shows the evaluation metrics for the final hybrid model.

E. Evaluation and analysis of past work

Variation 5 of the hybrid model achieves better performance than state-of-the-art techniques. Past research included different models and methods; we can see from the literature review that this challenge was approached in the same way by all researchers in [2]-[3]-[4]-[5]-[16]-[17]. All researchers were focused on creating a model that is complex enough to generalize well on the dataset FER-2013. Researchers in [5]-[16] were able to create a convolutional neural network model from scratch. The author in [4] added an attentional mechanism focusing on different parts of the images reaching the highest performance accuracy of 70%. Other researchers in [2]-[3]-[16] were focused on using pre-trained models like Xception, AlexNet, and V.G.G. model architecture, with the highest performance acquired by the V.G.G. network reaching 73.2% accuracy.

On the other hand, our proposed model surpassed all state-of-the-art models achieving 82.7% and 0.66 accuracies and loss, respectively, on the validation set. Our model design includes a simple architecture of the hybrid model CNN-LSTM with less than ¼ of other researchers' model parameters. Our approach differs from past research because of our data-centric approach, focusing more on the data instead of the model. However, the model architecture was not ignored at all. Instead, appropriate callbacks and hyper-parameters were set to the model to avoid overfitting and optimize the model with the best parameters for a high generalization rate.

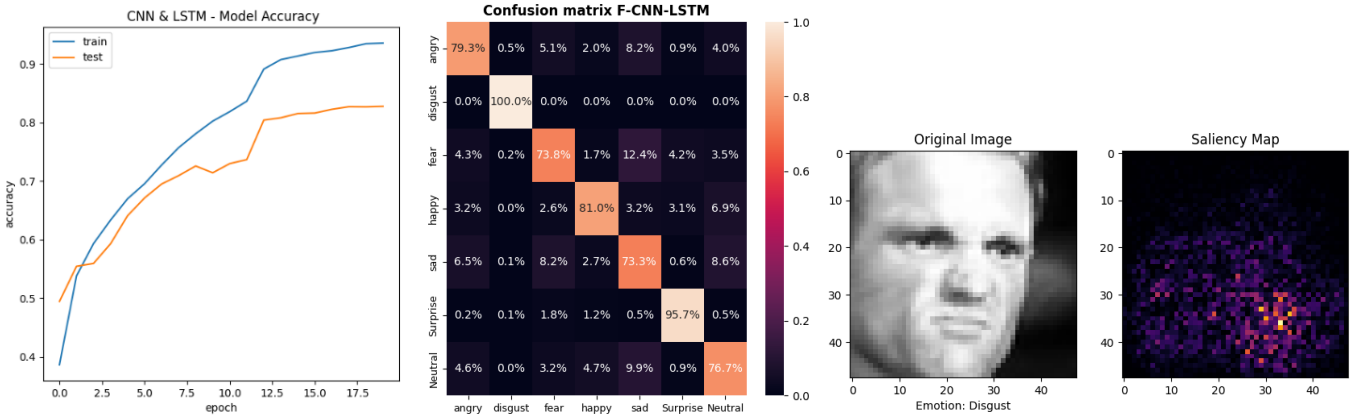


Figure 19: Evaluation metrics for CNN-LSTM Variation 5 model.

8.6 Vision Transformer - ViT

A. Intuition (Attention is all you need)

A recurrent neural network has a short reference window, so R.N.N.s cannot refer to past data sequences when a relation within data gets longer. However, Long-Short-Term-Memory, LSTM has a longer reference window than R.N.N. The attention mechanism has an infinite range of reference windows. Therefore, it is used for tasks like text generation. An attention mechanism can reference sequences of data from the entire context when generating a story. In a paper titled "Attention is all you need," the authors introduced a new neural network called transformers, which is a self-attention mechanism based on an encoder and a decoder architecture. The encoder aims to map an input sequence into a continuous representation that holds learned information about that input sequence. The decoder's goal is to take the continuous representation of the input and generate a single step-by-step output while feeding the decoder the previous outputs. The encoder module consists of two sub-modules: MultiHeadAttention, a fully connected network, and a layer normalization. This multi-head attention mechanism is a self-attention mechanism. Self-attention allows the module to associate each image in the input with other images. In this experiment, we try implementing the mutliHeadAttention mechanism with a CNN for image classification problems.

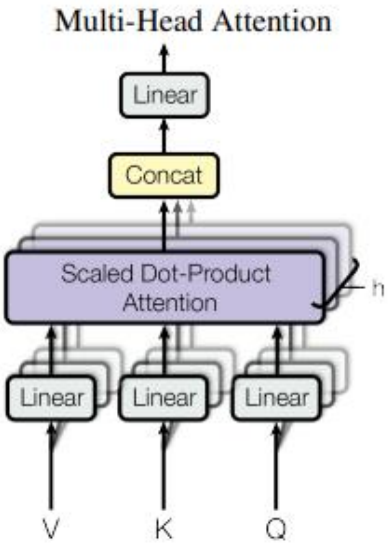


Figure 20: Transformer model with the encoder and decoder modules.

B. Model Architecture

The model architecture combines CNNs and Transformer (via multi-head self-attention) for image classification. The CNNs extract visual features, and the Transformer layers learn contextual relationships between those features to aid in variety. This type of model that combines CNNs and Transformers is known as a Vision Transformer or ViT. First, the model receives an input (48, 48, 1), indicating our grayscale images of size 48*48 pixels. The first convolutional block applies a Conv2D layer with 32 filters, a kernel size 2*2, ReLU

activation, and the same padding. Followed by the first block is a layer normalization and max pooling with a pool size of 2*2. The second convolutional block applies a Conv2D layer with 64 filters, a kernel size of 3*3, ReLU activation, and the same padding followed by the same normalization layer and 2*2 max pooling. After the second convolutional block, a multi-head attention layer with eight heads and a key dimension of 8 is applied. The attention mechanism is performed on the spatial dimensions of the feature maps. A dense layer with 200 units and layer normalization follows the multi-head attention layer. Then, two more convolutional blocks are applied. The first one has a Conv2D layer with 64 filters, a kernel size 3x3, ReLU activation, and the same padding, followed by layer normalization. The second one has a Conv2D layer with 128 filters, a kernel size 3x3, ReLU activation, and the same padding, followed by layer normalization. After the second convolutional block, the feature maps are reshaped to have a 2D shape (-1, 128), where -1 indicates an inferred size based on the other dimensions. Another multi-head attention layer is used with 8 heads and a key dimension of 8, but now the attention is performed along the temporal dimension (after reshaping). A dense layer with 200 units and layer normalization follows the second multi-head attention layer. A dropout layer with a rate of 0.2 is applied. A dense layer with 128 units and ReLU activation is added, followed by layer normalization. Global average pooling is performed along the temporal dimension, resulting in a 1D representation. The output layer consists of a dense layer with seven units and softmax activation to classify the input to one of the seven categories of our multi-class classification problem. The final model consists of 197,799 trainable parameters. However, this model is not considered to be a complex model compared to other model architectures mentioned in literature review.

C. Pre-processing

For this experiment, we have a variation of a model with all pre-processing mentioned in chapter E.D.A. of the dataset. The pre-processing included data pre-processing for converting the features and target columns appropriately to train the model. The data pre-processing had random oversampling to solve the imbalanced class problem within the dataset, converting pixel features into a normalized vector and the target feature into one-hot encoded vectors. The pre-processing was done by the function pre_proc_data().

D. Hyper-parameters

To create a multi-headed attention model, we must understand the attention mechanism that takes a query, key, and value as input and a computer-weighted sum of the values. Where weights are computed using the dot product of the query * key. The parameter num_heads in the function MultiHeadAttention() provided by Keras determines the number of parallel attention heads in the multi-head attention layer. Each head computes an independent attention mechanism, and the results are concatenated and linearly combined at the end. This allows the model to focus on different aspects of the input simultaneously, which can improve the quality of the linearly combined results. The parameter key_dim indicates the dimensionality of the keys; it determines the size of the dot product space, which impacts the expressiveness of the attention mechanism. A higher key dimension allows the model to learn more complex relationships between the queries and keys. In practice, these hyperparameters are often tuned through experimentation to find the best combination that maximizes performance on a validation set. Hence, choosing the hyper-parameters for the first model can be random. The initialized parameters for the num_head and key_dim was set to 8 for both parameters to test the initial performance of the model.

Moreover, to avoid overfitting and increase generalization within our model's training, we introduced callbacks to the model. Three main callbacks were implemented in the code, which is ReduceLROnPlateau monitoring val_loss with the patience of 2 and min_delta = 0.0001, EarlyStopping was set with the patience of 4 while monitoring val_loss, ModelCheckpoint is used to save the best checkpoint in training. The optimizer used to train this model

was Adam optimizer, with an initial learning rate of 0.0002. The model had been trained for 25 epochs with consistent improvement in the validation accuracy but not in the validation loss.

E. Model Evaluation

Using the hyper-parameters mentioned above, where the num_heads=8 and key_dim=8 lead to an overfitting within the final five epochs of training. The model's performance is consistent until epoch 20. However, the confusion matrix suggests that the model cannot generalize to unseen data. Disgust was the only class classified correctly by the model showing promising results. However, the saliency map shows that the model's self-attention head cannot focus on important pixels that refer to emotions. The map suggests that the number of MultiHeadAttention modules in the model architecture is complex and needs to be more capable of understanding where to direct its attention mechanism. However, because this is the first model using this architecture, the model can be fine-tuned to show promising results. This can be established by focusing the Multi-Headed attention mechanism on essential parts of the face. This model's final achieved accuracy and loss are 74.97%, and 0.850, respectively. The increased value of loss compared to accuracy suggests that the model is still able to correctly classify some of the validation data despite not predicting the correct class probabilities. This can happen if the model makes confident predictions, but those predictions could be more accurate. To conclude, this model shows promising results from the first training. However, it needs to be fine-tuned and regularized to achieve lower loss values and good generalization on training data.

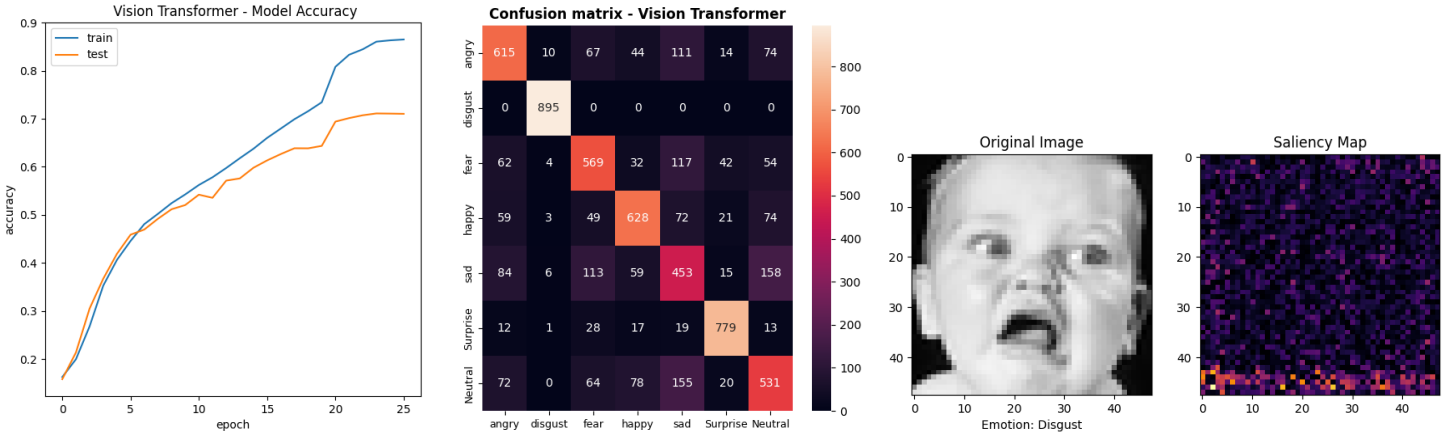


Figure 21: Evaluation metrics for the Vision Transformer model.

Proposed Models						
Model architecture	**Variation**	**Test split**	**Epochs**	**Sampling**	**Acc**	**Loss**
CNN & LSTM	**A.** Regular	**10%**	**14**	**ROS**	**78.36%**	**0.69**
	B. Dup Removed	**10%**	**16**	**ROS**	**80.68 %**	**0.69**
	C. Non-matching Dup Removed	**10%**	**15**	**ROS**	**81.56 %**	**0.64**
	D. Outliers Removed	**10%**	**15**	**ROS**	**82.37 %**	**0.744**
	I. CNN-LSTM (A)	**20%**	**53**	**Class weight**	**54%**	**1.19**
	II. CNN-LSTM (C-D)	**20%**	**14**	**NO**	**63.6 %**	**1.12**
			14	**SMOTE**	**79.4 %**	**1.15**
			20	**ROS**	**82.7 %**	**0.66**
Variation Transformer	**III.** ViT (A)	**20%**	**6**	**Class weight**	**17.2%**	**1.9**
	IV. ViT (C-D)	**20%**	**22**	**NO**	**51.6 %**	**1.32**
		20%	**17**	**SMOTE**	**57.1 %**	**1.13**
		20%	**28**	**ROS**	**74.97 %**	**0.85**

Table 2: a benchmark for the proposed models with their performance on their different performance.

9. Conclusion

In this paper, our objective was to create a single best-fitting model for the FER-2013 dataset. Our interest in related work is research done on the same dataset using a single network architecture with no additional data with the FER-2013. This paper introduces different deep learning models, focusing on hybrid models, which include regular CNNs, R.N.N., and MultiHeadAttention modules. These models have achieved state-of-the-art accuracy compared to past work done by others mentioned in the literature review. The related work discussed above focused on creating the best single-model architecture suitable for the dataset. For example, researchers in [2][3][17] tuned predefined models to the F.E.R. dataset; in addition, they added some trainable layers at the end of the predefined models to classify images of the F.E.R. dataset into seven emotional categories. Other researchers, like the one mentioned in [5], created a model architecture composed of CNN, pooling, and dense layers. Only research paper [4] has introduced a hybrid approach between CNN and a Localization layer, in which the localization layer focuses on the essential parts of the face. However, the key to our success in reaching 82.7% accuracy is our Data-Centric Approach, which is a way of designing and developing a model that strongly emphasizes the data used instead of focusing on the model architecture and hyper-parameters. The main principle of a data-centric approach is to focus on understanding the data, identifying potential issues and biases, optimizing the data to improve the model's performance, and choosing the appropriate model for the clean data.

In this work, we have not only achieved the highest attained accuracy of 82.7 % using a single network with no additional data, but also, we have managed to introduce new issues that reside within the F.E.R. dataset that past authors still need to consider. Moreover, we have proved that data quality and pre-processing techniques can lead to higher performance without using complex models. Data is the most significant factor in creating robust deep-learning models. Increasing the model's ability to generalize is directly related to the quality of the data and the pre-processing done to the data. Future work may include incorporating the clean data with variations of the Vision Transformer model; hence, it has shown to the extent that it does have potential if fine-tuned.

10. Future Work

In future work, we expect that considering the problems found within the dataset should be considered by researchers. Our study shows that the improvement led by dropping misleading images and outliers is non-negotiable. For future experiments, we can improve and fine-tune the Vision Transformer model to be more suitable for the F.E.R. dataset. In addition, future work may include studying the possibility of creating a cleaned F.E.R. dataset that consists of merged clean data representing real world scenarios of facial emotion for public use. Moreover, this study was concerned with assembling the best single-network model. Future studies may include investigation of different ensemble models. These models can be chosen so that they oppose each other (i.e., models for different emotion classifications). Suggestion of ensemble models is creating a Binary classification Tree that classifies emotions using one Vs All methodology.

11. References

[1] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, Y. Zhou, C. Ramaiah, F. Feng, R. Li, X. Wang, D. Athanasakis, J. Shawe-Taylor, M. Milakov, J. Park, R. Ionescu, M. Popescu, C. Grozea, J. Bergstra, J. Xie, L. Romaszko, B. Xu, Z. Chuang, and Y. Bengio, "Challenges in representation learning: A report on three machine learning contests," *arXiv.org*, 01-Jul-2013. [Online]. Available: https://arxiv.org/abs/1307.0414. [Accessed: 04-Apr-2023].

[2] I. Hatzilygeroudis and V. Palade, "Deep Learning Approaches for Facial Emotion Recognition: A Case Study on FER-2013," in *Advances in hybridization of intelligent methods models, systems and applications*, Cham: Springer International Publishing, 2018, pp. 10–25.

[3] S. Chand, A. Singh, R. Bhatia, I. Kaur, and K. R. Seeja, "Real-time facial emotion recognition using Deep Learning," *Algorithms for Intelligent Systems*, pp. 219–226, 2021.

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

[5] A. Saravanan, G. Perichetla, and D. K. S. Gayathri, "Facial emotion recognition using convolutional neural networks," *arXiv.org*, 12-Oct-2019. [Online]. Available: https://arxiv.org/abs/1910.05602. [Accessed: 25-Nov-2022].

[6] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[7] P. Ekman, "Basic emotions," *Handbook of Cognition and Emotion*, pp. 45–60, 2005.

[8] https://www.kaggle.com/datasets/msambare/fer2013

[9] A.: Communication without words. Psychol. Today 2(4), 53–56 (1968)

[10] H. Larochelle, "Learning Algorithms for the Classification," *Journal of Machine Learning Research,* vol. J1K 2R1, no. 643-669, p. 27, 3/12/2012.

[11] C.-D. Caleanu, "Face expression recognition: A brief overview of the last decade," *2013 IEEE 8th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, 2013.

[12] E. Sariyanidi, H. Gunes, and A. Cavallaro, "Automatic analysis of facial affect: A survey of registration, representation, and recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 6, pp. 1113–1133, 2015.

[13] urthy, G.R.S., Jadon, R.S. Recognizing facial expressions using eigenspaces. In: IEEE International Conference on Computational Intelligence and Multimedia Applications. 3, pp. 201–207 (2007)

[14] I. Perikos, E. Ziakopoulos, and I. Hatzilygeroudis, “Recognizing emotions from facial expressions using neural network,” *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pp. 236–245, 2014.

[15] I. Perikos, E. Ziakopoulos, and I. Hatzilygeroudis, "Recognize emotions from facial expressions using an SVM and neural network schema," *Engineering Applications of Neural Networks*, pp. 265–274, 2015.

[16] A. Mollahosseini, D. Chan, and M. H. Mahoor, “Going deeper in facial expression recognition using deep neural networks,” *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016.

[17] Y. Khaireddin and Z. Chen, “Facial emotion recognition: State of the art performance on FER2013,” *arXiv.org*, 08-May-2021. [Online]. Available: https://arxiv.org/abs/2105.03588. [Accessed: 04-Apr-2023].

[18] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2014, September 17). *Going Deeper with Convolutions*. arXiv.org. https://arxiv.org/abs/1409.4842

[19] M. K. Pramerdorfer and Christopher, "Facial expression recognition using convolutional neural networks: state of the art," arXiv, vol. 1612.02903, 2016.

[20] Y. Tang, "Deep learning using linear support vector machines," arXiv, vol. 1306.0239, pp. 1-4, 2013.