# Human Facial Emotion Recognition from Digital Images Using Deep Learning

Rivenski Alexander

A thesis submitted to the Auckland University of Technology
in partial fulfillment of the requirements for the degree of
Master of Computer and Information Sciences (MCIS)

2022

School of Engineering, Computer & Mathematical Sciences

# Abstract

In this thesis, our aim is to investigate the best performing methods and algorithms of facial emotion recognition (FER) based on the seven classes of human facial expressions of emotion: Neutral, scared, angry, disgusted, sad, happy, and surprised. We classify human facial emotions from digital images. The existing methodology of FER has various limitations: Low training and testing result, and difficulty in classifying certain emotions. Convolutional Neural Network (CNN), Xception, Visual Transformers (ViT), Simple Deep Neural Network (SDNN), and Graph Convolution Neural Network (GCN) have been proposed to the FER with two types of methods: non-facial landmarking and facial landmarking. The first method that we proposed is the modified CNN with Haar Cascade algorithm for frontal face detection as an initial solution. While running our experiments in real time, our accuracy for FER is up to 90.0%. Our first effort was with the same methodology and parameters as our initial method Mini Xception. With the Mini Xception model, we achieved stable result with the highest accuracy of 99%. In addition, with the popularity of deep learning algorithm Transformers, we implemented Visual Transformer with the non-landmarking method. With the training and testing work conducted based on the proposed model, we achieved the highest training accuracy in fewer number of training epoch compared to the previous two models. Due to misclassification occurred in the previous methods, we developed a new method of facial landmarking. The models we proposed are Simple Deep Neural Network (SDNN) and Graph Convolutional Network (GCN). Our SDNN model was employed as a baseline to test the proposed method. With this model, we achieved 96.0% accuracy in our real-time testing experiments.

In future, we plan to train and test the GCN model with facial landmarking method to determine the best performing model. In addition, we will implement the ViT model in real time using the non-landmarking method.

**Keywords**: Facial expressions of emotion, Facial emotion recognition, Simple deep neural network (SDNN), Xception

# Table of Contents

# List of Figures

# List of Tables

# Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgments), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Signature: _____ Date: 05 July 2022

# Acknowledgment

First, I would like to thank my parents for providing financial support for my study. With their support, I have the opportunity to complete my master thesis work with the Auckland University of Technology (AUT), New Zealand.

I would also like to express my deepest gratitude to my supervisor Wei Qi Yan. In this study, he not only provided me with professional knowledge support and careful guidance, but also helped me enrich my learning experience. I believe I could not complete my study without Dr Yan's supervision and instructions.

Rivenski Alexander

Auckland, New Zealand

05 July 2022

# Chapter 1
## Introduction

*This chapter is composed of five parts. In the first part, we introduce the background and motivations, the second part includes the research question, followed by the contributions, objectives, and structure of this report.*

## 1.1 Background and Motivation

Facial emotion recognition (FER) is one of the aspects of Artificial Intelligence (AI) or machine intelligence. Human moods can be felt by looking at facial expressions. With the employment of FER in robotics, it allows a communication between robots and human, the reason is that human expressions of emotion imply on human behaviors. This refers to non-verbal communications.

Human moods are reflected in facial expressions of emotion. Facial emotion recognition (FER) allows to determinate the emotions of an individual. A slew of algorithms has been developed for FER by combining CNN with other algorithms (Mellouk, et al., 2020), such as Graph Convolutional Network (GCN) (Papadopoulus, 2021) and Graph Neural Network (GNN) (Ngoc, et al., 2020). Other models have also been implemented that provide better performances. The models include Exception, Simple Deep Neural Network (SDNN), and Visual Transformers (ViT). With the proposed algorithms and methodologies, various limitations such as low accuracy in training and testing with different dataset, and misclassifications of similar emotions still occurs. The focus of this thesis is on how digital images from a database can be employed to determine what emotions are shown and presented with the prediction accuracy. A wealth of databases has been collected for model training, testing, and validating. One of the exemplar databases is the JAFFE database, containing 213 grayscale images labelled with seven human expressions of emotion (Zadeh, et al., 2019) (Jain, et al., 2019).

In this thesis, we present our methods for FER. We implemented two types of methods to investigate the best method for FER. The first method is non-facial landmarking with the employment of three algorithms: CNN, Mini Xception, and Visual Transformers. The second method is facial landmarking with the employment of Single Deep Neural Network (SDNN) and Graph Convolutional Network (GCN). All these methods are with the use of the programming language Python. The output of all our models is a training accuracy in real time. We present our training result with the use of graph representation and time series analysis, while our real-time experiment is presented

with the metric of accuracy. The trend in this thesis is an increase in the number of epochs, which leads to an increase in accuracy.

Our initial FER algorithm, CNN achieved the accuracy 56.0% in the model training. However, with the real-time tests, our model achieved an accuracy rate up to 90.0% with one of the emotions, misclassification of emotions still occurs often. In order to improve the CNN methodology, we employed Mini Xception algorithm for training and achieved accuracy 68.0% and live accuracy 99%. With an increasing popularity of Transformer models, we implemented Visual Transformer model in this thesis project, from the training of our model, we achieved accuracy of 69% for our training dataset and 70% for our testing set. With further needs for improvement of the accuracy of our experiments, we explored the methods of facial landmarks and the simple deep neural Network (SDNN) as a test model. With model training having 300 epochs, we are able to achieve accuracy of 69% and up to 96% in our live test.

To further improve our accuracy, we trained the models with increasing number of epochs 400, 500 and 600. While increasing the number of epochs to 400, we are able to achieve accuracy 69.0% with the highest accuracy 77% as well as 600 epochs for our SDNN model. With the CNN model, we are able to achieve accuracy up to 76% while increasing the number of epochs to 600. With the current project, which is able to only train ViT model, we believe that this algorithm is the best method for FER task. However, based on the result we gathered using real-time test, we believe that the Mini Xception and the SDNN model are suitable for FER. In the near future, we plan to implement the training and real-time test of the GCN model with the method of facial landmarking.

## 1.2 Research Questions

In this thesis, our aim is to implement multiple types of deep learning algorithms to determine the better performing method and algorithm for human facial emotion recognition. This thesis is conducted by training and testing our models in real time. Therefore, the research questions of this thesis are,

(1) What method is better to be implemented for accurate human facial emotion recognition based on deep learning?

(2) What algorithm is better to be implemented with the method for accurate human facial emotion recognition?

The main idea of this thesis project is to determine which method and algorithm can provide the better performance and more accurate result for human facial emotion recognition. The methods that we implemented in our experiments are non-facial landmarking and facial landmarking. In this thesis, we train and test our algorithms and methods, with the addition of real-time experiments using conventional webcam. The result of our findings will be presented in the result chapter of this thesis.

## 1.3 Contributions

Due to existing algorithms and methodologies having various limitations of low accuracy in training and testing with different dataset, and misclassifications of similar emotions, our research focus is on determining the better performing deep learning algorithm that is able to perform human facial emotion recognition with accuracy above 90%. Based on our models that have been trained and tested, we achieved the classification of facial emotion recognition with 90% accuracy. By the end of this thesis project, we are able to:

- Pre-process dataset to a format that is suitable for the model
- Define and create our own deep learning model
- Train and test each model with the pre-processed dataset
- Apply the model to a real-time experiment
- Present the findings in accuracy percentage with emotion predictions

Our project is unique as we provide two types of methods and various algorithms or models to predict the emotion and its probability or accuracy. We implemented non-facial landmarking and facial landmarking method, with the following algorithms:

- Convolutional Neural Network (CNN)
- Mini Xception

- Visual Transformers (ViT)
- Simple Deep Neural Network (SDNN)
- Graph Convolutional Network (GCN)

## 1.4 Objectives of This Thesis

The objective of this thesis is to determine the better performing methodology and algorithm that can be employed for facial emotion recognition. To investigate our objective, Research of most popular and accurate algorithm, experimented on our modified algorithms and analyze our experiment result by comparing them to existing FER algorithms.

For our research process, we reviewed different models or algorithms that achieved accuracy over 90%. With the existing accurate algorithms, we then modified our algorithm and method similarly done by existing algorithm. We introduced non-facial landmarks and facial landmark methods with the algorithms detailed previously. We take use of each model for training and testing, and real-time experiments to determine which of the methods and models performs better. While analyzing the result, we compared our models to the existing deep learning methods.

## 1.5 Structure of This Thesis

The structure of this thesis is described as follows:

- In Chapter 2, we discuss the methods that have been implemented by related work in literature for FER. The algorithms such as Convolutional Neural Network (CNN), Mini Xception, Deep Neural Network (DNN), Visual Transformer (ViT), and Graph Convolutional Network (GCN) will be detailed. In addition, the results of the algorithms will be demonstrated.
- In Chapter 3, we will introduce our method of FER. It includes the database we employed, the pre-processing of the datasets, the algorithms that was implemented, our time series analysis method, and our real-time experiments.

- In Chapter 4, we will depict our findings and results. In addition, we will compare the results of our methods to the related work.
- In Chapter 5. we will draw our conclusion and our future work.

# Chapter 2
# Literature Review

*The focus of this thesis is on identifying the best method and algorithm for human facial emotion recognition from digital images based on deep learning. In this chapter, we will introduce a plenty of methods and the new knowledge in deep learning.*

## 2.1 Introduction

A many of algorithms has been proposed by various studies for human facial emotion recognition. CNN has been employed as one of the most popular algorithms for FER. A two-level CNN network has been proposed for FER (Mehendale, 2020). The CNN was employed for feature extracting from the input images. Conventional CNNs were applied to extract the primary expressional vector (EV). The network consists of a 3×3 kernel matrix and the minimum error coding for optimization of filters. With the added datasets, the accuracy rate is up to 96.0%. There are a group of other algorithms for FER that have been compared to perform a better FER task. The algorithms include Xception, Visual Transformers (ViT), Simple Deep Neural Network (SDNN), and Graph Convolutional Network (GCN). These algorithms will be further investigated in this project.

## 2.2 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) is a model in deep learning that is employed to eliminate visual feature extraction. It can be employed in computer vision, where the classification of human faces is one of the usages of the applications. CNN is built up of three kinds of neural network layers: Convolutions, pooling, fully connected layers. The main purpose of the layers is for feature map extraction where the network is trained based on training dataset which has the labels of each class.

In terms of classifications, CNN has two layers known as fully connected layer and softmax layer. The purpose of the fully connected layer is associated with the output vector that predicts probabilities of each class of input images, while the softmax layer is to provide the output of the classification, given the input data. CNN has been employed as one of the most popular algorithms for FER. A two-level CNN network has been proposed for FER (Mehendale, 2020). The CNN was employed for feature extracting from the input images. Conventional CNNs were applied to extract the primary expressional vector (EV). The network consists of a 3×3 kernel matrix and the minimum error coding for optimization of filters. With the assistance of the proposed datasets, the

accuracy is up to 96.0%.

A method (Duncan et al., 2016) was proposed based on real-time facial emotion recognition. Three databases: CK+, JAFEE, and home-brewed database are employed to determine the seven classes of primary emotions. The initial image classification method was implemented with VGG network. Due to the low accuracy of 24% based on the CK+ dataset, 14% on the JAFFE, and the misclassification, corresponding improvements were needed. It was conducted by using transfer learning with the addition of randomized jitter for the dataset, by randomly changing the cropping 10% and the brightness by 20%. With the three databases, a total of 2,118 labeled images are obtained.

The method was proposed as follows, firstly the use of Haar Cascade to detect the human faces from the input images. The face detection is then put through the CNN network. The network structure is designed as: Five convolutional layers, three max pooling layers, one softmax classifier, and three fully connected layers.

The convolutional layers start as 7×7, then decrease as it goes through the first max pooling layer of 3×3 to 5×5, then finally to 3×3. The second max pooling layer was resized to 2×2 and followed by 3×3 after the last convolutional layer. Due to time limitations, only the last three fully connected layers are trained. With the improvements, the accuracy is increased to 90.7%. It is a significant increase in comparison to the previous accuracy.

Another method was proposed by using CNN and Gabor filters for FER (Zadeh, et al., 2019). The image pre-processing involves resizing the input image to128x128 which is then applied to two Gabor filter layers to output a filtered image. The filtered image is then harnessed for the CNN network. The CNN network consists of a convolutional layer of 6×6 kernel that filters the image, a max pooling layer reduces dimension to 128×128×6, a convolutional layer is applied to 16×16 filter size, the max pooling layer reduces the size to 64×64×16, the convolutional layer is applied to the data of 120×120 filter size, flatten layer or function converts a vector size 432,000 to a vector with a length of 84, then seven. The seven vectors are the representations of the seven classes of facial emotions. The JAFFE database, which contains 213 Japanese female models, has seven classes of facial expressions of emotion. The accuracy that the proposed model achieved was 91% compared to 82% without the Gabor filter.

A video-based human emotion recognition with hybrid network of RNN (i.e., Recurrent Neural Network) and C3D (3D Convolutional Network) has been proposed (Fan et al., 2016). A specific kind of RNNs, namely, LSTM (Long Short-Term Memory) network was employed to solve traditional RNN problems in learning long-term dependencies. LSTM was combined with a deep neural network for the training model and video footages. The model LSTM, also known as CNN-RNN, is from the FC layer of the VGG-16 model. It then got fine-tuning with the FER2013 face emotion database.

The other network is C3D with the structure of eight convolutional layers, five max pooling layers, two fully connected layers, and a softmax output layer. This model was trained based on AFEW 6.0 database, which contains 1,750 video clips. The videos were split into 774 clips for training, 383 clips for validation, and 593 clips for testing. The output accuracy was 59.02% in FER.

## 2.3 Deep Neural Network (DNN)

A proposed method was use of a single DNN (Deep Neural Network) for human emotion recognition (Jain et al., 2019). The datasets which were considered are CK+ (Cohn-Kanade). The dataset contains 8,363 images, 8,200 images for training and the remainder as testing and validation. Before applying the DNN algorithm, data pre-processing was conducted by using Gaussian normalization and standard deviation. The pre-processing of the images allows the removal of background and unnecessary contents such as human hairs. It was accomplished by cropping vertically and horizontally.

The cropping was completed by using a ratio 4.7 while the horizontal is only by 2.5. Another image pre-processing was employed with contrastive equalization. This was a two-step method. The first is to subtract the local contrast of images, then local contrast normalization was applied to the images. The value of the pixels was generated by contrastive equalization which is subtracted from the Gaussian-weighted average of its neighbors and then divided by the standard deviation of neighbor pixels.

The process of FER is to run the algorithm and get the pre-processed images. The network of DNN consists of six convolutional layers, two blocks of deep residual

operations, a max pooling layer for each convolutional layer, and two fully connected layers. The fully connected layers contain ReLU activation function and dropout for training.

The deep residual blocks are harnessed after the 2nd and fourth convolution layer. The structure of the residual block contains four convolutional layers. The first convolution is structured with the size of 1×1×64, the second convolutional layer with 3×3×64, the third one with 3×3×128 and the last layer with the size 1×1×256. The single DNN model was employed to train the model based on the given dataset and it could achieve the accuracy up to 95%.

The novel Deep Neural Network (DNN) was proposed for multi-view FER with facial landmarking (Zhang, 2016). The six classes of emotions are angry, disgust, fear, happy, sad, and surprise. The model DNN was inspired by artificial neural networks based on facial expressions of emotion. The facial landmarking and scale-invariant feature transform (SIFT) are associated with the DNN model. The proposed methodology was divided into three steps.

The first step is the deployment of 2D SIFT feature matrix that contains low-level extracted facial landmarks. Then the projection layer is used for the discriminative of facial features across all the facial landmarks. Lastly, they employed 1D convolutional layer to extract high-level features. SIFT is employed as a feature extraction method. The feature extraction method consists of four steps: (1) Scale-space extrema detection, (2) key point localization, (3) orientation assignment, and (4) key point description.

For locating key points, the method is accomplished by annotating a fixed number of key points on facial images. The key points are placed on visual features such as nose, mouth, and eyes. For the step of key point description, normalization is applied to ensure that there is no change of illumination. The next step is that DNN will work with the SIFT feature extraction to get facial feature vectors. The architecture of the DNN network consists of six layers. The six layers are two projection layers, one 1D convolution layer with a max-pooling layer, two fully connected layer, and one softmax layer. The network will require low-level features and high-level features. The low-level features are extracted with the 2D SIFT descriptor while the high-level features will be used by 1D

Convolutional operation. The model takes feature matrix as an input to be trained.

The feature matrix is applied to the landmark points and the SIFT feature vector. The feature matrix is **M**×**N**, where **M** is the vector consisting of landmark points and **N** is the SIFT feature vector. The **M** vectors are also utilized with the projection layer to produce discriminative facial features in association with human expressions of emotion. In the second projection layer, the **N** vectors are used to extract high-level discriminative features.

The next step is the 1D convolutional layer with its max pooling layer. The convolutional layer acts a filter that processes the small local parts. It is a 1D sequence that is able to extract high-level features by using the ensembled features from the projection layer. The result of filtering was conducted by using the convolutional layer which are then passed to the max pooling layer that outputs new feature maps. The max pooling layer is an extra filtration that results in the robustness of the network.

Lastly, the fully connected layer and softmax layer have the same function as CNN network, which was employed for the classification of facial expressions of emotions. When it comes to the training of the proposed model, the training dataset are divided into subsets. The metrics for training is loss function while the experiment is evaluated in terms of accuracy across the datasets.

In this study, two databases are taken into consideration: Multi-PIE and BU-3DFE. The Multi-PIE dataset contains six facial expressions of emotion: Disgust, neutral, scream, smile, squint, and surprise. It was collected with 337 people having 15 viewpoints, which was from 337 people, 235 males and 102 females who are from multiple countries.

The BU-3DFE dataset also contains six emotions: Anger, disgust, fear, happiness, sadness, and surprise. It was collected from 100 people, 56 females, and 44 males which results in 2,400 facial expressions of emotions. The models have 3D geometrical shapes and colors with 83 feature points (FPs). The training metric is measured by loss function which is employed to evaluate the differences in predicted results and annotated labels. The parameters of the network layers are updated accordingly to the loss function.

It is achieved with the employment of backpropagation algorithm. In the first iteration, the mean negative value of the predicted probability of training samples was

calculated, while the second and third iteration ensure a sparse structure of the matrices within the projection layers. The sparse matrices in the projection layer purpose are employed to measure the discriminative features through weights on the landmark points. Lastly, the steps of the algorithm are based on the proposed model for classifying the facial expressions of emotions with the deployment of the two datasets.

Multi-PIE dataset is experimented based on six facial expressions of emotions having the seven views: 0°, 15°, 30°, 45°, 60°, 75°, and 90° of one illumination condition. There are various parameters for the model based on the Multi-PIE dataset. The first of the parameter is the feature matrices of the input layer. The matrix size is 68×128 which was associated with the SIFT descriptor. The next step was to construct the matrix related to the projection layer; for the left projection layer, the matrix size is 5×1×30×68, which is translated to five one-channel matrices, each matrix of the one channel contains 30 rows and 68 columns. The following parameter is the filter size of convolutional layers with the size 5×5×1×3, the number of channels is as same as the left projection layer. For the right projection layer, the sizes are 5×5×63×30.

The next is fully connected layer, the layer has a transformation matrix with the size of 4500×400. In other words, it transforms the dimension of visual feature 4,500 pixels to 400. For the second fully connected layer, the size of the transformation matrix is 400×6, since it is the last layer for classification, the number six corresponds to the number of classes of facial expressions of emotion that the dataset has. The experiment was conducted with cross-validation. The dataset was split into 80% for training and 20% for testing, which results in 3,360 images for training and 840 images for testing. Due to the low number of training images, image augmentation was adapted.

The methods for image augmentation are mirror transformation and image rotation. The mirror transformation was conducted based on the result of extracted images from SIFT descriptors of the original image. For the image rotation, the original images are rotated by 10 degrees in both clockwise and counterclockwise. The result of the image augmentation with an increase in the number of training dataset to 13,440 images. With all the training and testing, the model based on Multi-PIE dataset has produced an accuracy 82% for the dataset with 3,360 images which was an average accuracy across

the seven different views.

While looking at the confusion matrix result, by classifying scream and surprised, it achieved accuracy over 90%, followed by over 80% for the emotions like disgust, neutral and smile, and 72% for squint. The next experiment is on the BU-3DFE dataset. The process is similar to the Multi-PIE dataset. The differences are the parameters. For the input layer, the size is now 83×128 instead of 68×128, it means that there are 83 rows and 128 columns. The number 83 is the representations of the key points from the annotated images within the dataset. The left projection layer is also with the size 5×1×30×83 instead of 5×1×30×68. The convolutional layer consists of 1×3 filters with five channels. The right projection layer and the fully connected layer have the same metrics.

Table 2.1: Result comparison of existing popular methods and algorithms

| **Algorithms** | **Accuracy Rates** |
|---|---|
| CNN-based dynamic facial emotion recognition (Duncan et al., 2016) | Training - 90.7%<br>Test - 57.1% |
| FERC (Mehendale, 2020) | Training - 96% |
| CNN-RNN and C3D (Fan et al., 2016) | Training - 59.02% |
| Convolutional neural networks and Gabor filters (Zadeh, et al., 2019) | Training - 91% |
| Single DNN (Jain et al., 2019) | Training - 95.23% |
| DNN (with landmark) (Zhang, 2016). | Multi-PIE Training- 85.2%<br>BU-3DFE Training – 80.1% |
| ViT (Dosovitskiy, et al., 2021). | Training - 99.68% |

Due to the datasets, the number of training and testing samples will be different. BU-3DFE dataset contains 12,000 facial images from 100 people with five angles of view: 0°, 30°, 45°, 60°, and 90° with 83 facial marking points. Similarly with the previous experiments, cross-validation is employed with 80% for training and 20% for testing.

Along with the splitting ratio, the training set has 9,600 images and the test set is with 2,400 images. In this dataset, image augmentation was provided, as the dataset contains adequate number of training images. The training data with five angles of view has produced an accuracy of 80.1% which is slightly lower than other datasets. While looking at the confusion matrix, the highest accuracy for the emotion surprise was 91%, followed by happy and sad at 80% and 66% for the emotion fear.

From CNN and Single DNN that have been proposed, we see the highest accuracy is illustrated in Table.2.1. With this result, we plan to employ conventional CNN and Simple DNN as part of our methodology. The model will be modified, trained, and tested.

## 2.4 Facial Landmark

Facial landmark detection is one of the methods that was employed as an improvement of the previous methods. With the goal of tracking key features of human faces such as eyes, eyebrows, nose, mouth, and jawline, it allows better to detect human facial expressions of emotions. A group of algorithms have been developed for facial landmark detection. The algorithms such as LBF, Dlib, MTCNN, and MediaPipe have been employed.

Dlib is a popular open-source library that was employed for face detection and landmark detection. It is a pre-trained detector that estimates the location of landmarks with the use of coordinates (Lini, 2021). The algorithm produces 68 landmarking points that are identified on the face as shown in Fig. 2.1, the best machine learning algorithm is determined for landmark detections based on Cohn-Kanade dataset (Alvarez, et al., 2018). The method for face detection is with the applications of Histogram of Oriented Gradients (HOG) and a landmark algorithm based on Dlib model. The result was from multilayer perceptron that provided the best performance. Although the project was based on machine learning algorithm, Dlib model is still employed for landmark detection.

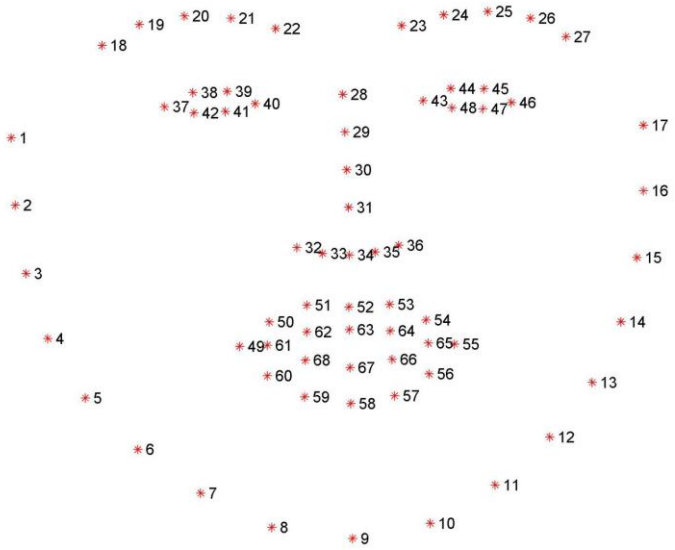



Fig. 2.1: Visualization of 68 facial landmarks

Facial landmark detection has been utilized with various datasets, algorithms, and pre-processing method. A method of FER was proposed to prevent road accidents, which were implemented by detecting facial expressions of emotions (Poulose, et al., 2021).

A feature vector extraction method was proposed with the use of facial landmarks. The feature vector extraction takes use of the combination of pixels and the landmarks for model training. For training, a dataset was created with nine people that includes 9,095 images and emotions such as happy, sad, angry, surprise, disgust, fear, and neutral which were captured by using a digital camera. The neural network model ResNet was trained based on the given dataset.

The proposed method consists of three steps: Pre-processing dataset, feature vector extraction, and ResNet training. The pre-processing method converts color images into grayscale ones, removes the background noises from the facial images using image thresholding. The purpose of grayscale thresholding is to reduce the complexity during the training process of the model.

In the next step, the facial images are segmented from the background with the deployment of foreground extraction, which gives a better representation of facial expressions of emotion. The foreground image thresholding (FIT) is applied. It is to crop and resize the facial images. The output of the image pre-processing is pixels which will

be employed as an input for the feature vector extraction. With the implementation of facial landmark detector, it was able to detect 68 landmarks from the facial images that has been pre-processed.

The next method is to combine the pixels with the pre-processing and the landmarks. The combination of these two kinds of visual components is applied for training the ResNet model. The model consists of the various parameters, input layer, Conv2d, batch normalization, activation, max pooling, average pooling, flatten, and dense. The structure of the ResNet model consists of nine blocks.

The structure of the first block consists of nine hidden layers: Input layer, Convo2D layer, batch normalization layer, activation layer, max pooling layer, another Convo2D layer, another batch normalization layer, another activation layer and lastly the average pooling layer. The structure of the second block is similar to the first block, however, it does not contain the input layer, as it is an addition to the first block. It contains a block of filtering layer which consists of batch normalization and activation layer followed by a Convo2D layer, a block of filtering layers, and average pooling layer.

The structure of the second block is as same as the even numbers of blocks, Block 4, Block 6, and Block 8. The odd number of blocks, Block 3, Block 5, and Block 7 contain similar structure to the even number blocks, however additional hidden layers are added. For example, in the third block, the structure consists of a block of filtering layer, a Convo2D layer, another block of filtering layer, an extra Convo2D layer and lastly an AveragePooling2D layer. Lastly, the last block or the ninth block, it only contains 3 hidden layers. The first layer is a block of filtering layer, followed by an average pooling layer.

The ResNet model takes use of the optimizers of Adam and single channel. The model is also trained with a number of epochs with output metrics of accuracy and loss with the utilization of early stopping. The experiments were able to achieve 100% for classification of happy emotion, 99% for disgust, sad, surprised, and neutral, and 86% for angry.

Dlib was employed for landmark extraction and OpenCV for the real-time test (Gupta, 2018). The work investigated real-time and static image for FER. The dataset was Cohn-

Kanade Database (CK) and the Extended Cohn-Kanade Database (CK+). For static images, the methodology consists of pre-processing the dataset, face detection, model training, and result analysis. The pre-processing of the dataset is split into three steps, the first is face detection, then feature extraction, and lastly facial emotion classification.

The research work focuses on eight emotions: Happy, sad, fear, anger, surprise, disgust, contempt, and neutral. The first step of the pre-processing is to separate the dataset into two folders. One folder is for text files and the other is for images. In addition, within the images folder, there are folders for each of the emotions that contains the specific images for the categories.

In the next step, after the pre-processing, regarding face detection, all the images have been cropped and stored into a folder with grayscale images. The method was applied to face detection by using the Haar filter from OpenCV library. The output of these images is then placed or stored into another folder. The folders represent and contain the eight classes of emotions which have been cropped and converted from color images to grayscale images.

With the pre-processed dataset, the proposed model has been trained. The dataset is split into 80% for training set and 20% for test set. The training set is for the training of the model to classify the emotions while the test set is employed to evaluate the performance of the model. Support Vector Machine (SVM) is trained through 20 iterations. The method with the static images has achieved an accuracy of 93% in the classification.

The next part of the research work is model testing in real time with the facial landmarks and Dlib model. The pre-processed dataset in this method consists of a webcam video that detects human faces based on each frame by using OpenCV library. The face images are converted into grayscale images and optimized with adaptive histogram equalization.

The next is feature extraction, with the face images after the object detection, it then will be pre-processed to extract feature maps for classifying facial expressions of emotion. The feature maps include the following visual objects: Eyes, eyebrows, noses, corners of face and mouth. It outputs landmarking positions on the detected face region when

different emotions are turned up. The position will be reflected on coordinates of the landmarking points. The means of both *x* and *y* coordinates are calculated. The output of the calculation is one center point based on all landmarks.

SVM model training is fulfilled in two steps. Firstly, the overall accuracy was achieved from ten different methods: Data segmentation, model training and result prediction. The second part is to evaluate the prediction probability. The result of the real-time test is that with linear SVM, it is able to achieve the highest accuracy 94.1% compared to multiple classifiers.

A more recent result is generated based on Dlib toolkit and SVM (Chouhayebi, et al., 2021). An approach of FER was proposed with geometric feature-based approach. Firstly, Dlib is employed for frontal face and facial landmark, the geometric features (GF) are extracted. SVM model is employed for the classification of facial expressions of emotions.

Dlib face detector was successfully operated by combining with SVM classifier and Histogram of Oriented Gradients (HOG). The second method is the facial landmark detector. Regarding facial landmark detection, Dlib toolkit was employed, specifically with the Dlib landmark detector. The landmark detector has the ability to estimate up to 68 positions. These are represented in *x* and *y* coordinates of facial parts such as eyes, mouth, chin, eyebrows, and lips.

The next step is the GF extraction, which takes use of three features: Angle feature, distance feature, and triangle feature. The first of the features is the distance feature. In this feature, Euclidean distance between the 68 landmarks is taken into account. In addition, the corners of human mouth, the range of eyes, and the range of mouth are recorded as facial features. Therefore, the feature vectors are composed of the distance between upper and lower lip, the distance in both eyes, the angle of the corners of the mouth, center of gravity (COG) of the face coordinates, COG of the left and right eyes, COG of the mouth.

For the triangle features, four triangles are employed to represent the features of eyes, eyebrows, and lips. These features from the distance, angle and triangle features are fed into the SVM classifier which outputs the probability of emotion classification. The experiment was conducted by using two databases: Personal database and BUHMAP-DB

(i.e., the Bogazici University Head Motion Analysis Project Database).

The BUHMAP-DB database was from eight people including three males, which contains four classes of emotions: Happy, sad, neutral, and surprised. The BUHMAP-DB contains 440 videos with various emotions, however, in this dataset, only the four classes of emotions were selected. The results are calculated and measured by accuracy and F1 score. SVM classifier with geometric features is able to achieve the accuracy 92.91% while the BUHMAP-DB dataset was able to produce 88.50%. In addition, with 10% test set, it is able to achieve accuracy 92.41%. The model was able to output accuracy 80% for all classes of emotions.

With facial landmarking (Tautkute, et al., 2018) and Deep Alignment Network (DAN), Emotional DAN model was proffered. The associated datasets used in the study are CK+ and ISED. Compared the methods of facial landmarking (Tautkute, et al., 2018), adjusting of facial landmarks and handling of an entire face image rather than patches are introduced. The training and testing of the DAN model that was implemented in Python are with the assistance of TensorFlow and Keras library. The results are measured in accuracy based on various benchmark datasets. For the training, the model was trained with AffectNet dataset. It is one of the largest datasets that contains over 1,000,000 face images and its emotions.

In model testing, the benchmark datasets include CK+, JAFFE, and ISED. These datasets contain over 180 individuals with genders and ethnicity. In addition, two methods are employed, one with seven classes of emotions: Happy, angry, sad, surprised, disgust, fear and neutral. The other is with three classes of emotions: Positive, negative, and neutral. With seven classes of emotions, the output accuracy is 73.6% based on CK+, 50.2% on JAFFE, and 62% on ISED. With only three classes of emotions, the output accuracy is 92.1% based on CK+, 76.5% on JAFFE, and 89.6% on ISED, respectively.

Facial micro-expression (FME) feature was proposed (Choi, et al., 2018) with the use of 2D landmarks (LMF) as well as CNN and Long-Term Memory (LTSM). The methodology is grouped into two steps. The first is LMF synthesis and the second is the classification. Essentially, the detection method contains landmarks which are fed into frame-based LMF, CNN-LSTM-based classifier, and lastly fed into a softmax layer for

the classification and probability prediction.

The frame-based landmark is working with the feature synthesis which outputs landmark feature maps. The next is the CNN-LSM-based classifier which consists of VGG-16 network and generates the output of frame-based LMF. It is then fed into stacked LSTM which consists of numerous LSTM blocks. The first step is to pre-process the frames to feed into the LMF synthesis. It takes use of the point-wise distance of two adjacent frames from the video input. The distance is then converted into a 2D LMF image which was gathered from each frame.

The landmarks were obtained with the active appearance model (AAM). This resulted in 68 facial landmarks. An approach is implemented that converts landmarks into LMF feature map. It was conducted with CNN-based network. In order to provide a robust LMF, normalization is applied to the landmarks of human faces. The result of the pre-processing is a unique image accordingly to the changes of direction or position of the facial landmarks. In other words, it is a landmark feature map that contains patterns of facial landmarks.

CNN-LSTM-based classifier is use of two networks, namely, the CNN and stacked-LSTM. The CNN network takes use of a VGG-16 network to classify landmarks. In addition, it is use of 1D feature factor with the size of 1.024 which was generated by using VGG-16.

The stacked LSTM was employed for the classification of the temporal characteristics. The network has intermediate feature generation process, it means that it can increase the capacity of the network. Both the CNN and stacked-LSTM correlate with each other to become the main framework of the CNN-LSTM-based classifier, however each of them takes use of different inputs. The CNN network takes an input of landmark feature maps while the stacked-LSTM takes an input as the output of the VGG-16 model that has been fed with a 1D feature vector.

The experiment was conducted with four datasets. The first dataset is the CK+ dataset, the second is facial micro-expression dataset (FME), the third is the general facial expression dataset (GFE), and last one is the mixed dataset. The CK+ dataset was employed for validation; a baseline dataset is applied for the creation of FME and GFE

dataset. GFE dataset is one part of the CK+ dataset, which has seven basic emotions that are shown in the video sequences.

In order to create the FME dataset, GFE dataset was taken into consideration. It was created from the frames of facial expressions. The next three frames are also taken into account that produces a sequence with nine frames. The last of the dataset is called mixed dataset. It is selected randomly with a ratio of 50:50 from the GFE and the FME datasets. The training was conducted with 10-fold validation, where nine subsets are employed for training and the remainder for validation. The result of the experiment is evaluated in accuracy percentage. With the general facial expression dataset, it was able to achieve accuracy up to 92.66% while with the facial micro-expression, it was able to achieve 77.98%. Lastly, with the mixed dataset, it was able to attain 87.46% accuracy.

Similar work for face landmarking was experimented with the use of CNN net (He, et al., 2017). The work is based on Robust FEC-CNN (RFC). FEC- CNN refers to fully end-to-end cascaded CNN. In the work, the FEC-CNN was taken into account as a basic method. The method consists of three parts: FEC-CNN for facial landmark detection, bounding box invariant, and model ensemble.

Firstly, the FEC-CNN is applied to a nonlinear mapping for the detection of facial landmarks. It is conducted by employing cascading sub-CNN networks. The work took use of the FEC-CNN as the basic. The difference is the bounding box invariant method and the model ensemble. The bounding box invariant technique is employed for improving prediction and the model training quality. In the work, face detection results are obtained with different face samples and detectors, the result of the cropped faces is fed into the FEC-GCN model to achieve the cropped images. The result of the FEC-GCN model has produced a lower variance which allows the minimum enclosing ROI or bounding box.

With the minimum bounding box and the cropped landmarks, FEC-GCN models were trained. The models were trained with various augmented data, various network structures, and various cropped face images. The last process of the Model Ensemble is to take the average mean of the facial landmarks created by each trained model. The experiment was conducted with the use of three datasets: 300W, 300W Competitions, and

Menpo. The 300W dataset contains numerous datasets, LFPW, HELEN, AFW and IBUG. To detect semi-frontal landmark, the datasets are divided into two parts. 300W Competition and Menpo are all used for the training. For the testing set, IBUG was selected.

The pre-processing methods are related to data augmentation. The methods include random rotating, translating, horizontal flipping and resizing. The data augmentation was conducted based on the training set.

AFLW is a large dataset that contains profile faces with 19 labels. In addition, the FEC-CNN is fine-tuned with the Menpo training set. With the RFC model, it outperformed to the previous FER-CNN models while looking at the mean error value.

Another method of facial landmarking is the use of Facial Action Unit (AU). The first step is to investigate micro-expressions based on graph representation learning done (Lei, et al., 2021). The network structure of proposed methodology is divided into three. The first channel shows facial graph representation, another channel reflects the AU matrix and the last one combines the outputs of the two channels to classify micro-expressions.

The basic process of the proposed methodology starts with the input. The input of the proposed model is one onset frame and one apex frame. The two inputs are then fed into MagNet to find magnified shape features. Then 30 patches of the eyebrows and mouth landmarks are extracted. The landmark of the two features was achieved with the employment of Dlib, which outputs the 68 landmarks that are separated into the two features. The result are the 30 nodes which are the graph representation of the facial landmark features. The facial graph is then fed into the node learning and edge learning model, this classifies as our Graph Learning model. Our AU model belongs to the same two features of the face, eyebrow, and mouth. The AUs are then embedded and fed into the GCN model. The result of this is AU feature matrix.

Lastly, the fusion strategy which combines AU with the learnt facial graph representation. Graph are represented as $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ where $\mathbf{V}$ is the node vector and $\mathbf{E}$ is the edge vector. While checking these nodes, a model called Depthwise Convolution (DConv) is implemented. The DConv starts with the input of the 30 nodes of facial graph. The patches of the landmarks then are extracted to create multidimensional matrix. To

convert the matrix, a straightforward method was employed to compress the 2D patches into 1D vector. With the conversion, it loses the vertical information between the pixels in the patches. At this point, the graph structure consists of node patches with patches which in this study refers to as channels. The DConv are then applied to the features from each channel to keep the spatial information of each node patch.

The proposed model is called Encoder Transformer (ETran). The structure of the encoder is six layers of a multi-head-self-attention and a fully connected feedforward network. ETran is employed to extract features as it allows computation of the relationships of the components. The node patches are fed into ETran to learn the relationships between edges and nodes. The result of ETran is a new matrix that will be used for the fusion.

The next method is AUFusion. By focusing on two features, eyebrows, and mouth, it outputted nine AUs that was selected. The relations of between the nine AUs can be presented by using an adjacency matrix while the node matrix was obtained by using word embedding. In other word, AUFusion is the combination of the nine AUs vector that produces a representation in adjacency matrix.

The last method is the GCN for obtaining the node matrix which is called AUGCN. A node matrix was generated with word embedding. The nine AUs are represented in a number format from 0 to 8. These numbers are stored in an input vector. The input vector is than mapped by using the embedding class in Pytorch which results in the node matrix. The adjacency matrix is computed by using conditional probability. Both the node matrix and adjacency matrix are then fed into the GCN for feature learning. The GCN consists of two layers. The output from the GCN is then split into two parts: One is the AU of the eyebrows (AoE), the other is AU of the mouth (AoM). With this method, the facial features can be flexibly separated, where the eyebrows facial features are fused with eyebrows AU features and same goes for the mouth. The AoE and AoM take the dot product from the corresponding vector from ETran. It then cascaded for classification.

The experiment was conducted by using two datasets: One is from the Chinese Academy of Sciences, Micro-expression II (CASMEII), Spontaneous Activity, Micro-Movements (SAMM), and Spontaneous micro-expression corpus (SMIC). The CASMEII

contains 255 samples with emotion labels, apex frame labels, and AU labels.

The action unit features were the eyebrows and mouth. Before running the experiment, pre-processing method was applied. The methods include alignment, cropping, gray processed and augmented. The result of the experiment was conducted through ablative analysis. It looks at all the models that has been proposed, DConv, ETran, AUGCN and AUFusion. The ablative analysis was conducted on CASMEII with 4 classes.

While looking at all the proposed models, the combination of all four models has produced the best result of accuracy of 80.080%. If the model is employed on other dataset such as SAM with four classes, it was able to produce accuracy of 82.39%. The model has also been experimented on the combination of the two datasets with four classes and produced an accuracy result of 79.95%.

Furthermore, the experiment was done on the same two datasets with five classes, the result of CASMEII with five classes is 74.27% while SAMM with 5 classes has produced 74.26%. As can be seen the model has provided a consistent accuracy result ranging from lowest of 74.26% to highest 82.39%.

Facial AU (Benitez-Quiroz, et al., 2016) was proposed for an algorithm that uses a large database for facial emotion recognition. In the proposed method, high accuracies are achieved based on various databases in real time. The facial expressions of emotion were grouped into 23 basic classes. It is implemented by using the detection of the activation pattern of the AU. In addition, if the images do not contain AU active, it will be classified as neutral emotion.

The first part of the method is defined as the feature space. It is employed to represent the AUs in the images. Configural features are defined by the statistics of the facial landmarks which includes distances and viewing angles between the points. It refines the configuration of the face. Facial landmarking vectors are 2D image coordinates. The landmarking points was employed by using two methods. The two methods include Supervised Descent Method (SDM) and Regression Tree which results in 66 landmarks.

Normalization was applied to the training images with the same inter-eye distance. When looking at the center of the right and left eyes, the location is computed as the geometric mid-point between landmarks. This then defines the two corners of the eye.

The shape feature vector was defined by using Euclidean distance between the normalized landmarks.

In addition, the angles of the normalized landmarks are defined by using Delaunay triangles. The next step is to use Gabor Filters. It is applied at the center of each normalized landmark coordinates or point with the purpose of modeling the changes in the shade which is caused by the deformation of the skin. Deformation of the skin refers to the changes in the skin when specific emotions are express, for example the wrinkle of the face. This results in the change in reflectance properties of the skin and the change in light source on the surface of the skin.

In other words, Gabor filter is employed to identify the changes in the shade if emotions are shown. The result of the Gabor filter is the feature vectors that defines the shape and the shade changes of the AU. The next process of the methodology is image annotation. In this step, the three datasets contain annotated AUs and AU intensity for the training of the classifiers. The dataset includes Denver Intensity of Spontaneous Facial Action (DISFA), shoulder pain database was fixed with Compound Facial Expressions of Emotions (CFEE). These datasets provide a large number of samples that was accurately annotated with AUs and AU intensities. In addition, it contains samples from both genders and different ethnicity subjects.

The algorithms are trained with these datasets to learn various AUs and AU intensity which allows them to be used for automatic annotation for the one million images in the wild. Wild refers to the images which were collected from the internet. Annotating images is conducted by checking if the activation pattern of the AU is listed in each row.

The experimental result is produced in three-fold. The first is the within-databases classification. The second result is the across dataset classification, and lastly the classification of the proposed algorithm is with the wild dataset. Within dataset classification, the datasets are CK+, DISFA and shoulder pain dataset by using 5-fold-cross validation. The metrics to evaluate the performance were F1 Score.

The F1 across the CK+ dataset ranges from approximately 0.8 to as high as approximately 1.0. This is similar to the result of the DISFA dataset, however the lowest of the score around 0.7. Lastly, for the shoulder dataset, the proposed algorithm score

stays in the range of above 0.95 consistently.

By comparing the result for CK+ set, the AUs classification scores are better than the other method, however, when it comes to the DIFSA dataset, the F1 score are all higher than the other studies.

The next is the across-dataset classification. The three datasets are CK+, DISFA and CFEE. In these sets, the leave-one-database is applied, two of the datasets are employed for training and one for testing. The F1 score for CFEE dataset ranges from approximately 0.5 to approximately 0.95, while the DIFSA ranges from approximately 0.5 to approximately 1.0. Lastly, CK+ has produced the lowest out of the three, ranges from approximately 0.4 to 0.9.

The next result set is from the wild dataset by using the proposed algorithm. In this set, the metric is the accuracy of automatic annotations. The experiment takes use of sorted dataset that was randomly selected. The random selection is 3,000 images from the top 1/3, 3,000 from the middle 1/3, and 3,000 from the bottom 1/3. The result of the top 1/3 was claimed to output 80.9% while the middle has produced 74.9%, and the bottom 1/3 has produced the lowest accuracy of 67.2%.

From the different studies of landmarking technique, we see that Dlib is the most popular of the method with most of the results. Instead of using the Dlib as an individual, it is used as a utility method that can be combined with other methods. One of the interesting methods that we found was with the use of Facial Action Unit (AU). It utilizes the Dlib for retrieval of the landmark and implemented them with other techniques.

## 2.5 Xception and Time Series

An approach has been proposed with Haar cascade and Xception for the utilization of time series analysis (Behera, et al., 2021). Haar Cascade was employed for face detection in video frames, when the Xception was employed for human emotion recognition. The data was collected over a period of time and saved into a CSV file. The CSV file contains facial readings that were collected around four times daily lasting for three days. It was saved into two columns – date-time stamp and recorded emotions, which contains 15,000

data points. The file is used as input for the FB prophet model. FB Prophet model is a forecasting tool that Facebook provides for the prediction of future emotions. The result of the model is the prediction of accuracy percentage for each emotion.

Time series analysis is a method that shows the data trend for time intervals according to Prabhakaran (2019). FER was employed to predict human emotions over a period of time and forecast human behavior. With time series analysis, the model predicts two emotions: Sad and angry. It was employed to show the trend over a period of time and a prediction of changes over a day. With sad and angry emotions along with the trend of a period, it shows an increasing possibility over time. Regarding the prediction, time series was applied to predict values for the next one hour of the input data. It takes all the inputs from the hourly data frames within that day of recorded data to predict the future classes of emotion.

A similar study to FER, with the use of environmental data to classify expressions of emotion was proposed (Maeda, et al., 2021). The development of a system that measures both environmental and emotional data simultaneously was proposed. The data was measured in time series. It takes use of deep learning model to predict future emotions in a few ten seconds. The work was experimented in two rooms at which the 14 subjects are ID with numbers which contains personal sensors. Furthermore, the rooms are installed with indoor sensors.

The data collection was accomplished within three months. Essentially, the experiment is fulfilled in three parts: (1) The use of time series model, (2) the environmental data, and (3) the emotional data. For the time series model, due to the collected data being multimodal data, to ensure the relevance of the features, in the work, a CNN-LTSM model is created. The CNN layer is applied to extract the features of the environmental data while the LTSM will get the time series data of the environmental data and the emotional data. The structure of the model consists of five CNN layers and one LTSM layer. The training process is accomplished based on 500 epochs with mean squared error (MSE) for its loss function.

The environmental data were collected from two sources. The first one is the indoor sensor, the second is the personal sensor that is attached to the individual’s desk. The

emotional data was collected with the employment of a wearable device with the use of NEC emotion analysis solution. NEC emotion analysis is based on biometric information from the wearable device. It measures the arousal level and emotional valence. The format of the collected data is a continuous stream that is calculated every five seconds. The results are presented in the method of time series analysis, the autocorrelation function (ACF). ACF determines the correlation of two data, in this study, time and emotional data.

## 2.6 Vision Transformers (ViT)

Due to the popularity of Vision Transformers in deep learning, we employed it to our method and determine whether the model provides a better accuracy to our experiment. A study on human facial expressions of emotions using ViT was conducted (Dosovitskiy, et al., 2021). In the work, with a small to mid-sized dataset, it was able to attain better accuracy compared to the traditional convolutional network. It takes advantage of the original Transformer model. In the model, the pre-trained prediction head is removed, a zero-initialized feedforward layer was attached. The patch size of the model stays the same.

In addition, 2D interpolation of the pre-trained position embedding was employed. These are referred to as resolution adjustment and patch extractions which are then employed onto the ViT model. The relevant experiment was conducted by using three algorithms: ResNet, ViT, and Hybrid. The models were pre-trained based on various sizes dataset. The metrics used for the experimental result is few-shot or fine-tuning accuracy.

The dataset was ILSVRC-2012 ImageNet Dataset. It contains 1,000 classes and 1.3 million images. In addition, the dataset is then combined with ImageNet-21k that includes 21,000 classes and 14 million Images, JFT with 18,000 classes and 303 million images.

The three models were then transferred to other datasets to perform benchmark tasks. The datasets include ImageNet with original validation labels, CIFAR-10/100, Oxford-IIIT Pets, and Oxford Flowers-102. The model has three variations: ViT-Base, ViT-Large, and ViT-Huge. Each of them provides different input of parameters. ViT-Base provides 12 layers with 12 heads and 86 million parameters.

ViT-Large is also known as ViT-L/16 is with the input batch size of 16 ×16 and provides 24 layers with 16 heads and 307 million parameters. ViT-Huge provides 32 layers with 16 heads and 632 million parameters. The models are trained with used of ADAM with a linear rate warm up and decay. For the model fine-tuning, SGD was employed with batch size 512. The result of the experiment was the comparison of the largest model on the different datasets. The models were employed for comparison with ViT-H/14 and ViT-L/16. The model ViT-L/16 was pre-trained based on JFT which shows a better performance than previous study by using Transfer Learning with ResNet.

Furthermore, ViT-H/14 outperforms all the other model based on all the datasets with highest performance of 99.68% on Oxford Flowers-102 dataset. Overall, the result shows that ViT outperformed conventional CNN models with less computational costs.

With the use of multiple Transformers model (Xue, et al., 2021) including Multi-Attention Dropping (MAD), ViT-FER and Multi-head Self-Attention Dropping (MSAD) have been proposed. The method of using the Transformers refers to TransFER at relation-aware facial parts. It is to distinguish the difference in local patches in different type of emotions. With the use of ViT-FER to better understand Transformer model, relation-aware local patches are from human faces for the FER task.

MSAD was employed to remove or drop the self-attention module. It is to further explore rich relations on different local patches. Then, MAD were applied to randomly drop an attention map. In addition, it is used to further extract the comprehensive local patches focusing on the diverse local patches instead of the discriminative patches. Lastly, experiment was conducted with the use of three datasets, RAF-DB, FERPlus and AffecNet.

The architecture of the TransFER consists of three algorithms. It is composed of the stem of CNN, Local CNN and MSAD. The stem of CNN was IR-50 with the purpose of feature map extraction. The local CNN is employed to extract the diversity of the local features with the utilization or guidance from the MAD model.

The local CNN architecture is divided into three steps. The first step is to input the multiple feature maps and feed it into LANet model. LANet model was employed in multiple local branches to automatically locate the important parts of the face. The layer

consists of two 1×1 convolution layer followed by a ReLU layer. On the second layer, the number of feature map is decreased to one and the generation of the attention map is with employment of the Sigmoid function.

The second step of the architecture is the use of MAD which acts like a Dropout layer in CNN, instead it takes use of feature maps and treats the feature maps as a whole. It ensures that the multiple local branches to learn the diverse and useful parts of the facial features. It harnesses the branches as the input and randomly drops a branch by setting the value to zero. This result in a creation of attention maps.

With the attention maps that was generated by the second step, we move on to the third step where the attention maps are combined into a singular attention map with the use of element-wise operation. The process results in all the unimportant areas of the feature map.

In summary, the local CNN locates the diversity in the local patches with the utilization of LANet. LANet also locates various discriminative areas and combined them with the use of maximum operation and element-wise multiplication. As TransFER is a combination of three models, the last model is MSAD. It is employed to learn the rich relationships between different local features that has been outputted by the local CNN. The structure of the MSAD consists of a Transformer encoder that contains MAD behind every Multi-head Self Attention module and MLP classification head. The relationship between the local patches is explored with the deployment of Transformers.

In order to transform the Transformer into 2D sequence, a projection module is introduced. Before feeding them into the encoder, the feature maps were split along the channel dimension and align them as a sequence vector, another word is to take the map features and reshaping them into a format that the Encoder can use.

The transformer encoder consists of multiple blocks. Each block is composed of multiple layers of MSA and Multi-Layer Perceptron (MLP). At the end of the encoder, a MLP head block acts a classification block that results in an output of emotions classification. Essentially the structure starts with MSA block, then MAD block then MLP block, lastly, at the end of the entire block of MSAD, the structure is the MLP head that classify emotions.

The process of the encoding is conducted in three steps. Firstly, the input of feature maps that has been reshaped is linearly transformed into queries, then the attention weight is calculate and lastly the weighted sum of all the values are calculated. The MSA blocks run its self-attention operations in parallel and linearly embeds the combined outputs which results in the final output for the MAD block. The MLP blocks consist of two fully connected layers and Gaussian Error Linear Unit (GELU). The fully connected layer is for the feature classification or projection while the GELU is for the non-linearity. The dropout and the MSA blocks are only applied during the training of the TransFER model.

Lastly, the experiment is accomplished with three datasets: RAF-DB, FERPlus and AffectNet. RAF-DB is a dataset that contains 29,672 labeled facial images which was collected by Flickr image search API. The 15,339 images from the 29,672 images contain six classes of facial expressions of emotion: Happiness, surprise, sadness, anger, disgust, and fear. The remainders are for neutral emotion.

For the training of the model, 12,271 images were employed while the remaining is used for testing. FERPlus is an extended dataset from the FER2013 dataset. It contains 28,709 training images, 3,589 validation images and 3,589 for test images. It contains eight classes of expressions of emotion, six of them are the basic expressions and the remainder are Neutral and contempt.

AffectNet is one of the largest FER datasets, which contains over 1 million facial images. In this experiment, 280,000 images were used for training and 3,500 were for validation. The experiment result is outputted in metric of accuracy. The model TransFER is trained with the employment of Stochastic Gradient Decent (SGD) optimizers and data augmentation. The augmentation methods consist of randomly rotating, and cropping, random flipping, and random erasing.

With regard to the RAF-BD and FERPlus, the model was trained on 40 epochs while the AffectNet was trained with 20,000 iterations. With the training, they were able to achieve accuracy of 90.91% for the RAF-DB, 66.23% for AffectNet and 90.83% for FERPlus. The method outperformed the state-of-the-art ones for all three datasets by a range of 1% to 2.9% accuracy.

A similar algorithm of ViT but with modification and addition called Visual

Transformers with Feature Fusion (VTFF) was proposed (Ma, et al., 2021). The VTFF model is pre-trained with two pre-trained ResNet-18 that contains two main components.

The first is the Attentional Selective Fusion and the second is the multi-layer Transformer Encoder. The feature extraction method consists of two ResNet-18 model. ResNet-18 is for the RGB image, and the other is for the Local Binary Pattern (LBP) feature image.

The next is the Attentional Selective Fusion (ASF), which consists of two classes of attentions: Global attention and local attention. The global attention model structure consists of one average pooling layer, followed by convolution layer, ReLU layer, batch normalization layer, an additional convolution layer, and batch normalization layer.

On the other hand, the local attention model consists of convolution layer, ReLU layer, batch normalization layer, another convolution layer and batch normalization layer. The input of the attention model is RGB image and the LBP images. Each of the input is fed into a convolutional layer with an element-wise addition operation. Both the global and local attention models are then supplied for a sigmoid function as the activation function.

The global and local selective fusion are conducted with the deployment of global average pooling and pixel-wise convolution. These are treated as global context for the global selective fusion and local context for the local selective fusion. The global context gradually condenses the feature map size into a scalar. In addition, the global context utilizes the inter-channel relationship of the features. Combining the global and local context will make the recognition of different features and uncertain emotions more accurate.

The next is multi-layer Transformer, the input of the encoder is the 2D feature maps that were outputted by using the ASF. The feature maps are then linearly flattened to create 1D position embedding which allows it to be fed into the encoder. The structure of the encoder is similarly done. Before the Multi-head self-attention (MHSA) and the MLP, instead of MLP head, one fully connected and a softmax layer are included to classify the emotions and its probability. Therefore, the structure of the encoder includes one normalization layer, followed by MHSA layer, another norm layer and lastly the MLP layer. The MLP layer are made out two position-wise feed-forward layer and Gaussian

Error Linear Unit (GELU), based on non-linear activation function.

The experiment was conducted with four datasets. The three FER datasets are RAF-DB, FERPlus, and AffectNet. The last of the dataset is the cross-dataset of the CK+. The RAF-DB and FERPlus are the same, however, with the AffectNet, 287,652 images were used as the training dataset and 4,000 images as test dataset.

With the addition of CK+ dataset, the dataset contains 593 video sequences at which 327 of them are annotated with seven basic emotions and contempt as an extra emotion. The dataset obtained 618 images with seven emotions and 654 images with eight emotions for testing. With the training of the model, the face images are detected with the employment of Multitask Cascaded Convolutional Network (MTCNN) algorithm. The face images are then resized to a size of $224 \times 224$.

The backbone algorithm or model is ResNet-18 which is pre-trained on MS-Celeb-1M dataset. The model is trained for 20,000 steps for RAF-DB and 40,000 for FERPlus. Other parameters such as optimizers, cross-entropy loss, batch size, and learning rate are applied.

For the optimizers, ADAM optimizers were supplied on the model, the batch size is set to 32, the learning rate is assigned to 0.005. The experiment was conducted in Python with the utilization of Pytorch toolbox. The result of the experiment is the accuracy from each of the datasets. For the RAF-DB dataset, the model has outputted an overall accuracy of 88.14% with the highest of 94.09% for the Happy emotion and the lowest of 64.86% for Fear. For the AffectNet dataset, the model was able to achieve overall accuracy of 61.85% with the highest accuracy of 88.40% for the Happy emotion and lowest of 53% for the Disgust emotion.

Lastly for the FER dataset, it was able to achieve accuracy 88.81% which was the highest one out of all the three datasets. While evaluating the cross-dataset on CK+, with the training on RAF dataset and testing on the CK+, the model was able to achieve accuracy of 81.88%. With the training process based on FERPlus and the same testing dataset, it was able to achieve 83.79% accuracy. With the highest accuracy 86.24%, the model was trained based on AffectNet dataset.

## 2.7 Graph Convolutional Network (GCN) and Graph Neural Network (GNN)

A graph is denoted as **G** = {**V**, **E**}, where **V** is the set of nodes and **E** is a set of edges. The size of graph is determined by the number of **V** where **E** describes the connections between nodes within the graph. A graph can be represented with the adjacency matrix which describes the connections between the nodes.

Graph can be utilized for machine learning. It is grouped into two categories in deep learning, node classification and graph classification (Ma & Tang, 2021). Node classification is a part of node-related tasks where the entity of the data is represented as one graph and the nodes are the data samples. On the other hand, graph classification is a part of graph focused on the data nodes consisting of multiple graphs and each data sample is an individual graph.

GNN refers to graph neural network that can be applied to graph-structured data. Pertaining to node-focused tasks, GNN aims at extracting features from each node so that node-focused tasks can be facilitated. Graph-focused tasks benefit from representative features of graphs. The graph features and graph structures are treated as an input, new node features are referred to graph filtering.

In order to create a GNN model, a message-passing operation was accomplished to modify the network of the proposed model. Pytorch was employed in Python as a tool for creating a GNN model. The landmarks of human face are associated with the Directed Graph Neural Network (DGNN) (Ngoc, et al., 2020). The DGNN consists of directed graph convolution blocks which accumulate information regarding vertices and edges. The blocks of DGNN are made up of temporal convolutional blocks that obtain temporal information from a video.

GLU was applied to the temporal convolutional blocks to prevent the vanishing gradient problem. The FER methodology consists of multiple steps. Firstly, facial landmarks are obtained from each frame by using a landmark extractor. Secondly, landmark locations are applied to reconstruct a graph structure. The landmarks that the

detectors have constructed are composed of 68 parts of human faces. These parts include nose, eyes, eyebrow, and mouth. Due to adverse effects on the FER performance, the features are deducted to only 51 landmarks. Owing to the nature of DGNN and the need for nodes, a master node has adopted the structure to capture information among distinctive nodes. Lastly, the DGNN algorithm was employed for graph structure analysis.

The experiment was conducted by using the CK+ dataset. The preprocessing methods include Gaussian noising which injected into the landmarking locations of each frame. With the data type of landmarks and DGNN, an accuracy is attained up to 96%. With the combined DGNN and C3D-GRU, the accuracy reaches 98.47%.

Graph Neural Network (Zhang, et al., 2021) was proposed based on spectral domain (FDGNN). The proposed method is divided into four steps. The first step is the pre-processing with data augmentation. The graph network was created with feature points and Euclidean distance, which is fed into the GNN for emotion recognition. The dataset consists of eight expressions of emotion: Four micro-expressions, and four macro-expressions. The micro-expression datasets consist of CASME II, SMIC, SAMM and SPOS. The macro-expression datasets contain CK+, JAFFE, TFEID and RAF.

All the datasets contain a diversity of samples which reflect various subjects. The micro-expression dataset CASMEII is composed of 26 subjects which includes 247 samples with six classes of emotions. SMIC consists of16 subjects which has 164 samples with 3 classes of emotions. The SAMM has 159 samples with seven classes of emotions, and the SPOS contains 231 samples with six classes of facial expressions of emotions.

The macro-expression, on the other hand, seems to have more samples. CK+ contains 593 samples with six classes of emotions, the JAFFE has 213 samples with six classes of emotions, the TFEID includes 7,200 samples with eight classes of emotions, the RAF consists of 29,672 samples with 6 classes of emotions. An additional dataset was created called FEC-13, which was collected from the internet and those existing datasets. The emotions are grouped into seven classes. The classes are the six basics emotions except neutral one with high intensity.

The basic methodology is divided into three steps. Firstly, the dataset was pre-processed, then the data was fed into spectral domain of graph neural network, lastly,

classification of emotions is conducted, its probability is calculated as the output. One of the pre-processing methods is data augmentation. The method of data augmentation was image rotating, image scaling, and Gaussian noising. The rotation was conducted in a range of -20° to 20°, the image scaling was scaled between 0.5 times and 1.5 time, the brightness adjusting, and Gaussian noising are with the mean value of zero.

After data augmentation, it is then made into an undirected graph that contains nodes and edges. The result of undirected graphs is input the GNN in spectral domain. Spectral Domain GNN architecture can be divided into four steps. The first step is to use the input from the data augmentation as an undirected graph. Since the graph is presented in terms of $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where **V** is the node and **E** is the edge, the undirected graphs contain nodes and edges. The node within the undirected graphs is connected to the nearest nodes, the distance was usually set to eight. The distance between nodes refers to Euclidean distance, which is used as an initial value of the adjacency matrix.

The third step is to calculate adjacency matrix. Because of undirected graph, the matrix would be considered as a symmetric matrix. The result is a matrix that can be used for the next step.

In the next step, the adjacency matrix is fed into three layers to produce an output: ReLU layer, fully connected layer, and the output layer, which is known as convolution operation. The adjacency matrix is imported into the ReLU activation layer, then fully connected layer, till the output layer with the use of softmax operation for the classification.

The model training and testing are conducted with the use of all the datasets. The training and testing datasets were split into ratio starting from 0.1 to 1.0. By comparing the results in accuracy as the metric, it has achieved accuracy of 100% with ratio 0.8 and 0.9; with three datasets: SMIC VIS, SMIC 2Class, and CK+, it also produced high accuracy 99%. With the training and testing datasets having various ratios, the model FDGNN is sensitive to feature vectors.

The next step is the result of facial expressions of emotion classification. With the datasets, the proposed method was able to achieve the highest accuracy 95.49% based on the CASMEII dataset, 82.73% on the SAMM set, 82.30% on the SMIC set, 79.11% on

the CK+ set, 78.68% on the TFEID set, 74.99% on FEC-13 set, 73.56% on the SPOS set, 53.61% on JAFFE set, and lastly 48.07% on the RAF set.

We see the results based on the micro-expression datasets tend to be better than the result of the macro expression. While checking the confusion matrix resulted from the experiments based on the FEC-13 dataset, we see that two classes of the expressions of emotion: Happiness and neutral have shown the highest recognition rate, however, the other classes of emotions are also misclassified with the emotions: Happiness, sand, and anger with the high intensity.

GCN is a kind of graph neural networks that include graphs for node classification. GCN is like the CNN. CNN makes use of neurons that contain weights, filters, or kernels to train its neighboring cells. It is akin to GCN, where the model also learns from the features by using its neighboring nodes. The difference between CNN and GCN is that CNN was operated on the structured data, whereas GCN takes effect on unordered nodes, the number of node connections varies. GCN is also grouped into two algorithms: Spatial Graph Convolutional and Spectral Graph Convolutional networks.

A similar approach has been proposed (Papadopoulos, et al., 2021) with GCN. The methods consist of various steps. The first step was designed for the estimation of 3D landmarks on human faces. In this step, 2D facial landmark detectors were applied to estimate 2D landmarks based on the dataset. The estimation is then employed for 3D faces and texture mapping on 3D meshes. The landmarks vary from 45 to 90 points that detect the facial features of eyes, eyebrows, nose, mouth, and chin, etc. A few landmarks were added based on the previously estimated landmarks.

The next step is to construct the patches of the landmarks based on 3D face meshes by using the neighborhoods of the landmarks. KD-tree is offered to find the closest points to each landmark and combine them to create feature maps. Lastly, the final features are applied to the proposed spatiotemporal GCN algorithm to implement face identification.

Another work with the Graph Convolutional Neural Network (GCNN) was also proposed (Xu, et al., 2020). In the work, Graph Convolution Neural Networks are utilized with the use of landmarking technique. The basic methodology consists of six steps. The first step is to get an image from a camera; secondly, the face is detected; thirdly, it is

resized to 128×128, the landmark detector is implemented, which is based on GCNN.

Pertaining to the model training and testing, various datasets are taken into consideration. The datasets are JAFFE, FER2013, and CK+. With the CK+ database, only the last frame of each sequence is employed for emotion classification. As we know, landmark detection is employed, DLIB is a toolbox that can be utilized for facial landmark detector and frontal face detector. For the landmark detection, it takes use of regression trees to estimate the positions of the landmarks. The result of the DLIB toolkit is facial landmarking based on all images with different classes of emotions.

The next step is the DGNN network. When it comes to facial landmarking, the landmark positions or coordinates are classified as the node while the connectivity is the edge which can be the labels of facial expressions of emotion. The GCNN classifier was employed with the utilization of Pytorch.

The network GCNN is grouped into three parts: Input, feature extraction, and emotion classification. The input layer is the first layer that takes an input of nodes and edges from the landmarks. The next step is feature extraction. The structure contains nine layers, seven of them are the convolutional layer, the remaining layers are the pooling layers. The classification layer contains two fully connected layers, the first layer contains 64 units while the second has 6 units. Each of the layers is followed by a RELU layer and for the fully connected layer, softmax is applied to output the labels of classes and probabilities. The training was conducted with various parameters, a number of epochs, loss, learning rate, batch size and weight decay. The number of epochs for training was 400 epochs, with employment of cross-entropy loss using SGD having 0.9 momentum. The learning rate was set to 0.1 with batch size of 64 and the weight decay of 0.0001.

In addition, data augmentation was applied to the dataset by using the methods such as rotating, shifting, scaling, noising, contrast adjusting, and color jittering. The result of the experiment is presented by using accuracy metric. With the confusion matrix, the classification result of each emotion corresponding to its emotions is above 90% accuracy with the highest one at 96.13% for the emotion Anger.

By comparing the overall accuracy with the other method and CK+ dataset, the highest accuracy is 95.85%. The algorithm is tested with the real-world data. The result

of the experiment has produced the highest accuracy 99% while classifying Anger emotion and the lowest one 82% for the emotion Disgust.

The facial landmarking method was proposed with the Spatial-Temporal Graph Convolutional Network (ST-GCN) (Chen, et al., 2019). A method consists of two steps. The first step is to combine pre-trained CNN networks. The CNN networks are 2D-CNN and 1D-CNN for extracting the spatial-temporal features from the videos and audio. The second is the Spatial-Temporal Graph Convolution Network (ST-GCN) for extracting facial landmarks. Video analysis for facial expressions of emotion is conducted. The 1D and 2D convolutional networks are employed for training and reducing the risk of overfitting. For the 2D-CNN, the ResNet-50 network was pre-trained by using FER+ dataset. The 2D-CNN was employed for feature representation.

ResNet-50 was employed to extract feature maps from a video. This results in feature vectors that represents the deep appearance spatial features. The weights and bias parameters in the network ResNet-50 are all distributed to all the frames. The 1D convolution is employed to the vectors. Each frame of the video is segmented in 5 frames with one label.

In order to reduce the computing time and complexity of the training, random selection of one single frame from each segment is employed which produces the training samples. The samples are fed into the ResNet-50 which results in a representation of the sample frames.

The representation of the sample frames is stacked and inputted into the 1D CNN. The output is spatial-temporal features. Furthermore, the ResNet-50 is employed to extract the feature maps. One of the five frames is used as an input for the 1D CNN. This process was repeated five times with the same method for extracting one frame out of the five frames of video segments. The final feature maps are extracted with the mean pooling.

Essentially, the process starts with image sequences that were fed into 2D CNN, then one of the five frames is selected to be imported to the 1D CNN, which was repeated five times, lastly it is supplied to the mean pooling to output the spatial-temporal appearance features.

Geometric features refer to the facial-landmarking features that allows the

differentiation of different facial expressions. The facial landmark in terms of graph structure and its temporal dynamic with the employment of the ST-GCN model is employed. As graph is denoted by **G** = (**V**, **E**) where **V** is the nodes an **E** is the edges, this is to classify facial landmarks as their nodes.

Regarding graph edges, two types of edges are taken into account. One is the spatial edges which represents the connection between the landmarks, the other is the temporal edges which connect the landmark across the sequence of images. In order to detect the face region, face alignment and the extraction of facial landmarks are based on the sequence of images, OPENFace toolkit is employed. The result of the implementation is the input of the ST-GCN which is the landmark vectors. An average pooling on the consecutive five frames is applied to match the number of labels. The audio and text features are treated as one part of this work; however, it does not relate to our FER. The model training includes the training on each of the method, spatial-temporal appearance, and spatial-temporal geometric features.

With regard to the spatial-temporal appearance, 2D ResNet-50 was pretrained on FER2013+ dataset with the learning rate 0.0001. The fully connected layer of the 2D ResNet-50 model at the last is replaced with the 1D CNN. Apropos training the 1D CNN, one fully connected layer is stacked on the top of the model. Then the fine tuning is implemented by using loss function. In addition, SGD optimizer is employed with the learning rate 0.01.

The STGCN is the model that is use of facial landmarks and geometric features for its input. With regard to the training, the fully connected layer was replaced to fit the features finally. In addition, ADAM optimizers were employed with the learning rate 0.0003. For the experiment, the dataset is with AVE2019 CES dataset. The dataset is based on SEWA dataset which consists of spontaneous emotions in the wild.

The subjects of datasets were from three countries: China, Germany, and Hungary. The Chinese subjects are employed for testing while the German and Hungarian dataset are used for training which consists of 874 segments. The experimental result shows a couple of findings. Firstly, when arousal and valence predictions were output, spatial-temporal appearance features (ST-App) have performed the best. This indicates that

appearance features fine-tuned with FER database can predict facial expressions states accurately. Secondly, the spatial-temporal geometric feature (ST-Geo) provides the second-best result for arousal prediction.

With the landmarking methods of Graph Convolutional Network (FERGCN) (Liao, et al., 2022), the proposed model framework is divided into three parts. The first is the feature extraction, then the GCN, and lastly the graph matching. The feature extraction network is built up on key point-guided attention and CNN branch.

ResNet-18 was designed with the purpose of extracting feature map from the images. Triplet attention network is also employed to process the feature maps. In the landmarking method, key-point-guided attention branch was developed. It utilizes the SAN method to detect 68 landmarks on facial images. There are 16 key points to be selected from the 68 landmarks which represent facial features of eyebrows, eyes, mouth, and nose.

In addition, two extra key points of the facial image are added which represents the cheeks. To get the middle point of the cheeks, triangular regions are employed that were created with three indexes of landmarks. This results in two extra key points from the right and left cheek, a total of 18 key points. The 18 key points were generated from 18 Gauss distribution heat maps and its feature vectors are treated as the output.

While classifying facial expressions of emotion requires more than one feature of the face and various datasets with varying conditions, a classification module is created. The structure of the module consists of two branches. The first branch is the local positions processing, this was done by multiplying the confidence of each key point to its feature vector. It is then combined with the employment of average pooling; the combined local features are fed into the fully connected layer. The second branch is the global feature vector. It is processed by the fully connected layer that contributes to global classification.

In summary, the feature extraction model is divided into two branches, the first one is the process that the local points and the second points are tackled with the global points. In the global points, ResNet-18 and Triplet attention network are employed which creates the three features. Both the local points and the global points are then fed into Z-pool operation and a fully connected layer to output a vector that can be used by the GCN model, which brings us to the next part of the proposed methodology. The GCN takes

local feature vectors as the nodes of graph and uses the relationship between the key points to extract information or labels for the facial expressions.

Again, the networks are divided into two branches: Global feature vectors and local feature vectors. On the first branch, the network structure takes use of the whole local feature vectors and the local feature points. In the global feature vector branch, multiple global feature vectors are combined to achieve enhanced feature information. This methodology is applied to the first branch; however, it takes parts and the entirety of the local feature vectors.

The result of the methodology is a graph representation of facial landmarks. The graph matching is employed to distinguish the difference in similar expressions and to extract further information. The method involves direct matching of corresponding points. It employs Cross-Graph Embedded-Alignment Layer (CGEA) for optimizing the results and finding the similarity in the corresponding points or the images.

In the experiments, the datasets include RAF-DB, SFEW, AffecNet, Occlusion-RAF-DB, and Pose-RAF-DB. The RAF-DB dataset contains 29,672 facial images with six basic expressions of emotions, 12,271 images are composed of the training set, and 3,068 images consist of the testing set. SFEW is a dataset that has been created by the selection of static frames from the AFEW dataset. It contains six basic emotions and neutral. The dataset contains 958 training images, 436 validation images, and 372 test images.

The next dataset AffectNet is the largest one among facial expression datasets. The Occlusion-RAF-DB and Pose-RAF-DB are testing database which were gathered from the RAF-DB dataset. It contains 735 occluded facial images. Pose-RAF-DB contains two types of images. The first type has 1,248 images with side face, the angles are greater than 30°, the other contains 558 images at which the view angle is greater than 45°. This dataset is only supplied for testing.

Before the experiment is conducted using these datasets, image pre-processing method was employed. The methods include image rotating, horizontal flipping, and random erasing. The image was rotated between -10° and 10° with horizontal flipping of 50%. The model training was conducted by using Pytorch in Python. The parameters such as batch size, training cycles, learning rate, learning rate decay, number of epochs and

optimizers were employed. The batch size was set to 64 with 80 training cycles at a learning rate 0.0035 to 0.1 at epochs 40 and 60. The optimizer was ADAM.

The result of this experiment was reported in accuracy as its metrics. AffectNet achieved an accuracy 62,03%, 56.15% on the SFEW set with the highest accuracy 88.23% on the RAF-DB set. With the Occlusion and Post RF-DB dataset, it was able to achieve 83.40% for occlusion, 87.89% for the pose with greater than $30^{o}$ and 86.74% for the pose with the angles which are greater than 45°.

# Chapter 3
# Methodology

*The main content of this chapter is to state the methodology that is employed for the process of FER. This includes the datasets, the pre-process of the dataset and the method used to show the result.*

## 3.1 Non-Facial Landmark

### 3.1.1 Database Pre-processing without Facial Landmark

Before we train our CNN model, we make use of image augmentation for our FER2013 dataset provided by Keras in Python. Image augmentation was employed to the original images by using various transformations, which produces copies of the original images. The copy of images is different from one another because of the purpose of model training. Image augmentation is one of the important methods to build a better performing model during its training. It allows the model to learn different presentation of the images which leads to better performance of the models.

In this thesis, we applied random rotations, random shifts, random flips, and random zooming to our dataset (Sarin, 2019) (Rosebrock, 2019). The image augmentation utilizes the ImageDataGenerator method that is provided within the Keras Python library. It takes use of the original image and randomly transforms them into new images according to the specified parameters. The dimensions of the new transformed images will be the same as the original images; the resolution is 48×48 pixels.

We firstly applied random rotation to our original images. Random rotation means that the images are rotated through a viewing angle from 0 to 360 degrees. The parameters of rotations decide the number of degrees. In this thesis, the images are rotated every 10 degrees.

Next, we applied random shifts. It is a transformation method due to uncentered images in the dataset, which is accomplished by shifting the images horizontally and/or vertically. In this thesis, we make use of both horizontal flips and vertical flips. The parameters can be taken as a percentage of flips. In this case, it is a 10% shift horizontally and vertically.

Random flipping is a method for data augmentation that was applied to flip images vertically and horizontally. The parameters are horizontal and vertical flip parameters. The parameters take a Boolean value TRUE or FALSE. If the value is TRUE, then the

images will flip according to the parameter, while FALSE will not flip according to the parameter. In this thesis, we only applied horizontal flipping. It means that our images will only be flipped horizontally. It is sensible to flip the face image horizontally; it is most likely be horizontally symmetrical compared to vertical.

Fig. 3.1: The examples after image augmentation

Random zooming is applied to zoom in or out of the images. The parameter is a zooming range, which takes in the form of floating-point numbers or decimal numbers. If the floating-point number is less than one, it will be zoomed in on the images. On the other hand, if the number is more significant than one, it will zoom out the images. In this thesis, we set the floating-point number as 0.10. Therefore, the images from our dataset are zoomed in. The samples of image augmentations are illustrated in Fig.3.1. All the parameters are employed for image augmentation.

With the ViT model, we were use of the same dataset of FER2013. The pre-processing of the dataset is similar however in ViT we have a two-step pre-processing method.

The first step of the method is to prepare the data for the ViT model. In this step, we employed the CSV file for the dataset and convert it into a format that can be loaded into ViT model. The format of ViT model is from an open-source library for Python called Hugging Face. Hugging face is a base library that can be applied to text, image and audio classification, segmentation, detection and more. The first step of our process is iterating through the CSV file to separate the image and its labels. In FER2013 csv file, pixels in

our images and labels are the encoded emotions where ‘0’ refers to “angry”, ‘1’ is “disgust”, ‘2’ is “fear”, ‘3’ is “happy”, ‘4’ is “sad”, ‘5’ is “surprised”, and ‘6’ is “neutral”. Once the dataset is separated into two classes of images, we then separated them into the types of usages, training, private test, and public test as illustrated in Fig.3.2 for the training dataset.

The next step of the pre-processing is the use of feature extraction method from the Hugging Face library. The feature extractor is a method that is employed to prepare our data to fit the ViT model, which includes extraction of features from images or in our case it is pixels of images. Within this step, we converted our images using the method mentioned and return the image accordingly to the extracted features. The next step of the process is to combine the pre-processed dataset into one dataset. The dataset includes its labels which is the seven types of emotions, images, and the pixel values of the images. The combined dataset contains 24,402 rows of training data, 3,589 for the validation data, and 3,589 for the test data. The dataset is then employed in the training and evaluation in our model.

| | img | label |
|---|---|---|
| 0 | [[[70, 70, 70], [80, 80, 80], [82, 82, 82], [7... | 0 |
| 1 | [[[151, 151, 151], [150, 150, 150], [147, 147,... | 0 |
| 2 | [[[231, 231, 231], [212, 212, 212], [156, 156,... | 2 |
| 3 | [[[24, 24, 24], [32, 32, 32], [36, 36, 36], [3... | 4 |
| 4 | [[[4, 4, 4], [0, 0, 0], [0, 0, 0], [0, 0, 0], ... | 6 |

Fig. 3.2: An example of FER 2013 train set for first step pre-processing

### 3.1.2 Algorithm without Facial Landmark

Haar cascade algorithm was employed in this thesis to detect frontal faces, which were employed in our real-time experiments with a webcam. Haar cascade classifier needs to be trained with positive images and negative images. The positive images, in this case, refers to the images having faces, while the negative images refers to the images without faces. The extraction of facial features from the trained classifier will occur.

In face detection using Haar cascades (Behera, 2020), the algorithm AdaBoost is employed to select the best features extracted from the training images. The name cascade

classifier was introduced for human face classification. The way it works is that the features are grouped into multiple stages. It selects them one by one. It will discard any windows that fail to their first stage and will no longer consider the remaining features on that window. The algorithm is from the Library of OpenCV.

Our CNN algorithm is composed of five blocks. Each block consists of two 2D convolutional layers, two batch normalization, one rectified linear unit (ReLU), one average pooling, and one drop out for the last block, as illustrated in Fig.3.3.

The convolutional layer is employed to detect features in the images. We applied Convo2DLayer, which is a 2D convolutional layer. The way convolution works are that it utilizes a filtering system with the parameters such as filter size and kernel size as mentioned by Smeda (2019), which will be explained further in this thesis.

Batch normalization is applied to standardize inputs to a network (Brownlee, 2019), its purpose is to coordinate the update of layers in the model. In this thesis, batch normalization is applied before the input and before the activation layer.

The activation function that we take in this thesis is ReLU. It is a linear function that takes use of positive and non-positive values as its inputs. If the input is positive, it will produce the output directly from the inputs; if the input is negative or non-positive, it will generate an output of zero (Brownlee, 2019). ReLU activation function has many advantages. One of them is the ease of training the CNN model. Dropout is applied to reduce over-fitting among the neurons in the model (Budhiraja, 2016). In this thesis, we set the rate as 0.5.

Average pooling is a pooling layer that makes use of the average operation from the block as it is downsampling. It keeps all the information or elements of the block. The application of average pooling is vital because the position of the objects is essential when it comes to facial features (Versloot, 2021)

Within each block, the parameters such as filter and kernel size will be changed. In this thesis, kernel size refers to the tuple of two integers that specify the height and width of the convolutional window. For the first block, the filters are set to 16 while the kernel size is 7×7. The filters for the rest of the blocks are then applied to the power of two till it reaches 256. For example, the second block for both the filter of the convolutional layers

is set to 32, the third block is set to 64, then 128, and lastly 256 for the last block. At the same time, the kernel size of each block will decrease from 7×7 to 3×3. For example, the second block will decrease from 7×7 to 5×5, and the third block will go to 3×3 and remain the same for the rest of the blocks.

In the last block, batch normalization is not applied before each convolutional layer but in the middle of the two. Instead of having average pooling, we have a global average to take the average of all the other blocks (Versloot, 2021). Another modification was replacing the activating function of ReLU with the softmax layer. The purpose of this operation is for the classification or prediction of emotions.

With the model training using our dataset, we set up an environment in Python that takes the following parameters: Batch size, epoch, input shape, validation split, number of classes, and CNN model.

Apropos our batch size, we set up the number as 64 with 20% training dataset. Batch size refers to the number of samples that the network will propagate throughout the model network. However, we have time constraints on training our network with the FER13 dataset which has seven classes of facial expressions of emotion. The small number of epochs could generate an underfitting or overfitting of the model.

Due to the resolution of the dataset, we have set the input shape to be 48×48. In addition, because our dataset contains grayscale images, it allows us to have one channel for the input shape. Therefore, our input shape has been set to 48×48×1. Our CNN algorithm will utilize this input shape for training and testing. In our validation experiment, the ratios are considered for the evaluation of our model. We have set 20% training dataset and 80% training set as it provides the best results and avoids overfitting (Gholamy, et al., 2018). In this thesis, we also make use of other parameters to compile our CNN model. The parameters we utilized are optimizers and metrics.

Optimizers are a method for altering the attributes of the CNN. In this thesis, we consider adaptive moment estimation, an optimization algorithm for training our model. It refers to a combination of two other algorithms RMSprop and Stochastic Gradient Descent (SGD). It makes use of squared gradients to scale the learning rate from the RMSprop and takes use of the momentum of moving average of the gradient from the

SGD (Bushaev, 2018). With the given metrics, the accuracy of classification has been taken into consideration. This measure aims to show the evaluation of our model in the form of an accuracy.

Before model training and testing, we apply callbacks to save our model. We exported our model into a Hierarchical Data Format 5 (HDF5) file for our experiment with the combined parameters from training. The format of the HDF5 file consists of the model, number of epochs, and accuracy. For example, the CNN model HDF5 format contains the number of training epochs that have been improved and the training accuracy for the epoch. We make use of the highest accuracy rate and epochs to ensure that the model is trained appropriately.

*Early stopping* is a method to stop the training of our model if the parameter has stopped being improved. In this thesis, we make use of the parameter *val_loss* to track loss accuracy that occurs during the validation of our model. The parameter is employed to record the number of epochs at which the model is no longer improved; at this point, the training will no longer be commenced. In this thesis, we assign the numerous values, we started with 300, then 400, 500 and 600, which means that the training procedure will be stopped if the model does not improve after using the respective number of epochs.

*ReduceLROnPlateau* is another callback method that we utilized before training our model that reduces the learning rate of our model if the monitoring metric has stopped improving. The ways we employed for this work are monitoring, factoring, and verbosing. The monitor parameter is as same as the parameters. The following parameter we employed is the factor, which refers to the learning rate that will be reduced. In this thesis, we set the factor as 0.10. Lastly, the parameter we employed is verbose. It is a parameter that takes in an integer. The value is one means that ReduceLROnPlateeau will update messages while staying quiet when it is at 0. In this thesis, we choose 1 to see if any improvement has been made. The improvement is recorded and outputted into a log file.

*ModelCheckPoint* is the last method of the callback that is found in the Keras library on Python that is employed to save our CNN model later from the state it was saved. There are various parameters within Modelcheckpoint. Lastly, *save_best_only* is the parameter that is utilized to save the best model. In our method, we only take into account

of the method that has improved from the previous epoch. The parameter is set to be True; therefore, it only saves the best model and the latest best model.

The next model that we built is Mini Xception. The purpose of this model is to compare our CNN method. It is to determine if another neural network can perform better than our model. We will explain the layers within the Mini Xception model. The result and comparison to our CNN model will be explained in the later chapter of this thesis.

Mini Xception model consists of five blocks similar to our CNN model as illustrated in Fig.3.4. The first of the block is the base layer that takes input from the dataset. It consists of two 2D convolutional layers with filter size of 8 and kernel size of 3×3. The size of the convolutional layer in our model is increased as the number of blocks increases. For example, on the second block, it is 16, then 32, then 64, and lastly 128. On the other hand, the kernel size stays the same throughout the blocks.

Each block is similar to CNN model. However, from the second block onwards, 2D convolutional layer is added after the first 2D convolutional layer. The 2D convolutional layer is the process of splitting a single convolution into two or more convolutions. In addition, we employed max pooling on the last layer of the block instead of average pooling.

On the last block, we applied the same formation however we added an extra convolutional layer that takes use of the number of classes within the dataset as the filter, global average pooling is necessary to take all blocks average for prediction. The activation layer will also be replaced with softmax layer which outputs our emotion prediction.

Owing to uprising in popularity of deep learning algorithms, especially Transformers, we employed a traditional model of Vision Transformer (ViT). ViT is a deep learning model that is pre-trained on ImageNet and ImageNet-21K datasets, which consists of numerous self-attention layers.

While comparing the ViT with CNN, ViT shows a weaker inductive bias, however which provides a better result with 4 times fewer computational resources for pre-training. One characteristic of ViT model is that it can output exceptional performance if it is trained with a big dataset compared to CNN. ViT model segments images into fixed-size

patches, it then embeds each of the images which includes positional embedding as the input of the encoder.

One of the key roles of the self-attention layer in the ViT model is that it allows the embedding of information across images. In addition, ViT model encodes the location of the image patches by using training data for the reconstruction of the image.

The structure of the encoder consists of Multi-Head Self Attention (MSP) layer, Multi-Layer Perceptron (MLP) layer, and Layer Norm (LN). Multi-Head Self Attention layer (MSP) combines all the attention output directly to the right dimensions which allows to train local and global dependencies of an image. Multi-Layer Perceptron (MLP) consists of two layers with Gaussian Error Linear Unit (GELU).

Lastly, Layer Norm (LN) is applied to each block for the purpose of improving training time and performance. The overall architecture of the ViT consists of multiple steps. The first step is to split an image into patches, then the image patches are flattened. From the flattened image patches, it creates lower-dimensional linear embeddings. The next is to include the positional embeddings which are then fed into the encoder, then pre-train the model with the image labels, and lastly fine tune the downstream dataset for classification.

The ViT model was pre-trained on ImageNet21k with 14 million images and 21,843 classes. It is a large collection database that has pixel resolution of 224×224 pixels. One drops out layer that was set to value of 0.1, linear classifier takes hidden size and the number of labels. Within our model, we included a forward pass and a loss computation function. The forward pass function allows us to pass through the architecture of the model and input the loss computation for the training of our model. The loss function we employed is Cross Entropy Loss.

While training our model, we used Trainer library within Hugging Face. The library can be modified to fit the needs of performance evaluation, this is referred to as training arguments. In our training arguments, we have *evaluation_strategy*, *logging_strategy*, *save_strategy*, *learning_rate*, *num_train_epochs*, *weight_decay*, *metric_ for_ best_ model*, *logging_dir*, *load_best_model_at_end*, the model and logs will be saved.

*evaluation_strategy* is the strategy parameter that the model will be employed during training our model.

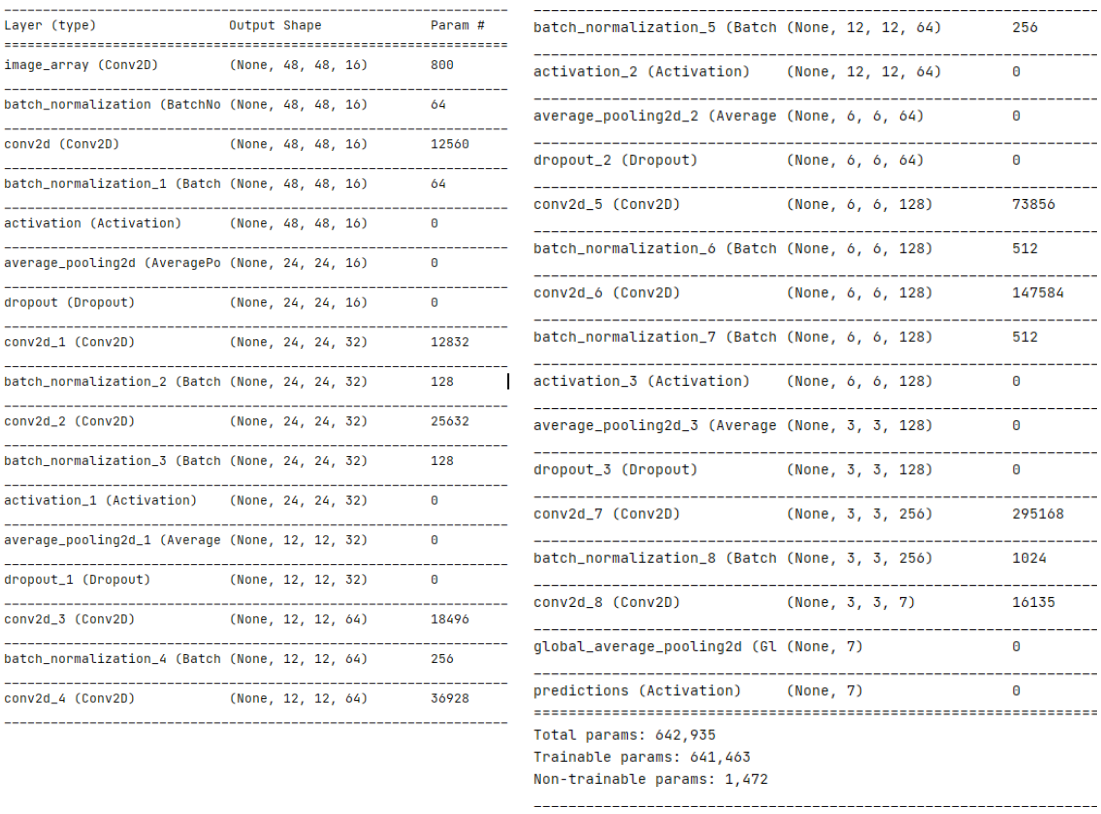
In our case, we employed “epoch”, our model is evaluated at the end of each epoch. The next parameter is the *learning_rate*, which refers to the initial learning rate for the ADAM optimizers. We set it as 0.00002. Then, it is *weight_decay*, which was set to 0.01. *num_train_epoch* is one of the arguments. The purpose of the argument is to set the number of training iteration that we want for the model to go through. Like other models, we use the number of epochs from 1 to 6. The number of epochs in this model is smaller compared to other models due to time and resource limitations. The time consists of the number of hours that it costs for the model to be trained. For one epoch, the model took approximately one hour and fifty minutes to train. With the time limitation to this project, we only trained the model through 6 number of epochs.

The remainders are related to the saving and fine tuning of the model, which includes the fined-tuned model, *load_best_model_at_end*, *metric_for_best_model* and the *logging_dir*. For the directory, we specified the folder that needs to be inputted in our project. The next argument is the *load_best_model_at_end* that is an argument or parameter which is shown as a Boolean logic. If the value is “TRUE”, it will load the best model during the training of the model at the end of training. To know which model to save, it requires a metric, the argument *metric_for_best_model* correlate with *load_best_model_at_end*. For our argument, we used “accuracy”. Therefore, the model has the best accuracy after training.

Another correlated argument is *logging_strategy* and *save_strategy*. Both the parameter takes “epoch” as the parameter input. *logging_strategy* is the factor that is used to save the outcome of the training, in our case, it is at the end of each epoch. Whereas *save_strategy* is the checkpoint that will save the model at the end of each epoch. Lastly, we save the training log. Training log will contain accuracy and losses. The argument that was employed is *logging_dir*. It is the directory or folder that the log file will go into. Once we have all the training arguments, we then load our training metrics of “accuracy”.

We then created a function that will be used by the Hugging Face trainer called *compute_metrics*. The *compute_metrics* function will include the predictions of probability and the labels of the datasets.

The next is the initialization of our model that can then be inputted as one of the arguments of the Hugging Face Trainer. Our Hugging Face Trainer consists of five arguments: model, *args* (training arguments), *train_dataset*, *eval_dataset*, and *compute_metrics*. The model arguments are the training parameters that we have initialized, *train_dataset* and *eval_dataset* is the result of our pre-processed method.



| Layer (type) | Output Shape | Param # |
|---|---|---|
| image_array (Conv2D) | (None, 48, 48, 16) | 800 |
| batch_normalization (BatchNo | (None, 48, 48, 16) | 64 |
| conv2d (Conv2D) | (None, 48, 48, 16) | 12560 |
| batch_normalization_1 (Batch | (None, 48, 48, 16) | 64 |
| activation (Activation) | (None, 48, 48, 16) | 0 |
| average_pooling2d (AveragePo | (None, 24, 24, 16) | 0 |
| dropout (Dropout) | (None, 24, 24, 16) | 0 |
| conv2d_1 (Conv2D) | (None, 24, 24, 32) | 12832 |
| batch_normalization_2 (Batch | (None, 24, 24, 32) | 128 |
| conv2d_2 (Conv2D) | (None, 24, 24, 32) | 25632 |
| batch_normalization_3 (Batch | (None, 24, 24, 32) | 128 |
| activation_1 (Activation) | (None, 24, 24, 32) | 0 |
| average_pooling2d_1 (Average | (None, 12, 12, 32) | 0 |
| dropout_1 (Dropout) | (None, 12, 12, 32) | 0 |
| conv2d_3 (Conv2D) | (None, 12, 12, 64) | 18496 |
| batch_normalization_4 (Batch | (None, 12, 12, 64) | 256 |
| conv2d_4 (Conv2D) | (None, 12, 12, 64) | 36928 |
| batch_normalization_5 (Batch | (None, 12, 12, 64) | 256 |
| activation_2 (Activation) | (None, 12, 12, 64) | 0 |
| average_pooling2d_2 (Average | (None, 6, 6, 64) | 0 |
| dropout_2 (Dropout) | (None, 6, 6, 64) | 0 |
| conv2d_5 (Conv2D) | (None, 6, 6, 128) | 73856 |
| batch_normalization_6 (Batch | (None, 6, 6, 128) | 512 |
| conv2d_6 (Conv2D) | (None, 6, 6, 128) | 147584 |
| batch_normalization_7 (Batch | (None, 6, 6, 128) | 512 |
| activation_3 (Activation) | (None, 6, 6, 128) | 0 |
| average_pooling2d_3 (Average | (None, 3, 3, 128) | 0 |
| dropout_3 (Dropout) | (None, 3, 3, 128) | 0 |
| conv2d_7 (Conv2D) | (None, 3, 3, 256) | 295168 |
| batch_normalization_8 (Batch | (None, 3, 3, 256) | 1024 |
| conv2d_8 (Conv2D) | (None, 3, 3, 7) | 16135 |
| global_average_pooling2d (Gl | (None, 7) | 0 |
| predictions (Activation) | (None, 7) | 0 |

Total params: 642,935
Trainable params: 641,463
Non-trainable params: 1,472

Fig. 3.3: The details of the proposed CNN model

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   Connected to
=================================================================
input_1 (InputLayer)         [(None, 64, 64, 1)]       0
_________________________________________________________________
conv2d (Conv2D)              (None, 62, 62, 8)         72        input_1[0][0]
_________________________________________________________________
batch_normalization (BatchNorma (None, 62, 62, 8)      32        conv2d[0][0]
_________________________________________________________________
activation (Activation)      (None, 62, 62, 8)         0         batch_normalization[0][0]
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 60, 60, 8)         576       activation[0][0]
_________________________________________________________________
batch_normalization_1 (BatchNor (None, 60, 60, 8)      32        conv2d_1[0][0]
_________________________________________________________________
activation_1 (Activation)    (None, 60, 60, 8)         0         batch_normalization_1[0][0]
_________________________________________________________________
separable_conv2d (SeparableConv (None, 60, 60, 16)     200       activation_1[0][0]
_________________________________________________________________
batch_normalization_3 (BatchNor (None, 60, 60, 16)     64        separable_conv2d[0][0]
_________________________________________________________________
activation_2 (Activation)    (None, 60, 60, 16)        0         batch_normalization_3[0][0]
_________________________________________________________________
separable_conv2d_1 (SeparableCo (None, 60, 60, 16)     400       activation_2[0][0]
_________________________________________________________________
batch_normalization_4 (BatchNor (None, 60, 60, 16)     64        separable_conv2d_1[0][0]
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 30, 30, 16)        128       activation_1[0][0]
_________________________________________________________________
max_pooling2d (MaxPooling2D)    (None, 30, 30, 16)     0         batch_normalization_4[0][0]
_________________________________________________________________
batch_normalization_2 (BatchNor (None, 30, 30, 16)     64        conv2d_2[0][0]
_________________________________________________________________
separable_conv2d_2 (SeparableCo (None, 30, 30, 32)     656       add[0][0]
_________________________________________________________________
batch_normalization_6 (BatchNor (None, 30, 30, 32)     128       separable_conv2d_2[0][0]
_________________________________________________________________
activation_3 (Activation)    (None, 30, 30, 32)        0         batch_normalization_6[0][0]
_________________________________________________________________
separable_conv2d_3 (SeparableCo (None, 30, 30, 32)     1312      activation_3[0][0]
_________________________________________________________________
batch_normalization_7 (BatchNor (None, 30, 30, 32)     128       separable_conv2d_3[0][0]
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 15, 15, 32)        512       add[0][0]
_________________________________________________________________
max_pooling2d_1 (MaxPooling2D)  (None, 15, 15, 32)     0         batch_normalization_7[0][0]
_________________________________________________________________
batch_normalization_5 (BatchNor (None, 15, 15, 32)     128       conv2d_3[0][0]
_________________________________________________________________
add_1 (Add)                  (None, 15, 15, 32)        0         max_pooling2d_1[0][0]
                                                                 batch_normalization_5[0][0]
_________________________________________________________________
separable_conv2d_4 (SeparableCo (None, 15, 15, 64)     2336      add_1[0][0]
_________________________________________________________________
batch_normalization_9 (BatchNor (None, 15, 15, 64)     256       separable_conv2d_4[0][0]
_________________________________________________________________
batch_normalization_11 (BatchNo (None, 4, 4, 128)      512       conv2d_5[0][0]
_________________________________________________________________
add_3 (Add)                  (None, 4, 4, 128)         0         max_pooling2d_3[0][0]
                                                                 batch_normalization_11[0][0]
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 4, 4, 7)           8071      add_3[0][0]
_________________________________________________________________
global_average_pooling2d (Globa (None, 7)              0         conv2d_6[0][0]
_________________________________________________________________
predictions (Activation)     (None, 7)                 0         global_average_pooling2d[0][0]
=================================================================
Total params: 58,423
Trainable params: 56,951
Non-trainable params: 1,472
_________________________________________________________________
```

Fig. 3.4: The details of Mini Xception model

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 272)]             0
_________________________________________________________________
dense (Dense)                (None, 128)               34944
_________________________________________________________________
dense_1 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_2 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_3 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_4 (Dense)              (None, 7)                 903
=================================================================
Total params: 85,383
Trainable params: 85,383
Non-trainable params: 0
_________________________________________________________________
```

Fig. 3.5: Our simple deep neural network (SDNN) model

## 3.2 Facial Landmark

### 3.2.1 Database Pre-processing for Facial Landmark

The difference in the pre-processing is that without facial landmarking, we were able to use the pixels within the database to train and test the non-facial landmarking model. While the facial landmark technique requires the dataset to be landmarked with the use of the Dlib toolkit. The purpose of separating the pre-processing is to investigate the different methodology as one requires further pre-processing than the other technique.

With the seven classes of facial expressions of emotion in the FER2013 dataset, we employed one-hot encoding, which is a method of converting numerical values into a new value of “1” or “0”. In FER2013, human expressions of emotion are sorted in the numerical values of 0 to 6, where the hot-encoding number “1” represents the TRUE value of the emotions. Each of the arrays contains 7 classes of emotions, the index of the arrays is the individual emotions as shown in Fig. 3.8.

In the face landmarking, we utilize three facial features: Left eye, right eye, and mouth. We believe that when emotions are shown on human faces, three features of the face provide the significant changes or information regarding the shown emotion. For example, with a base emotion of emotion neutral, we have our original faces with no

changes of the eyes, nor mouth, compared to when you are sad and happy. If a human face shows the happy emotion, our mouth displays a parabola curve or a “U” shape curve. On the other hand, if human face shows the sad emotion, our mouth creates an upside-down parabola or an upside down “U” shape.

Before we proceed further into the pre-processing, facial landmarking method had to be introduced to pre-process with the FER2013 dataset. The method involves the use of library called Dlib. It is a toolkit or library for frontal face detection and facial landmark predictor. In this thesis, we use Dlib to predict the locations of facial features in the position. The landmark algorithm then returns a list of locations that are can be implemented to the next step of pre-processing.

With the employment of the landmark algorithm, we were able to extract three features. With the employment of Dlib, we were able to get 68 points of landmarks. Our feature extraction method only extracts three features of frontal human face, left eye, right eye, and mouth. Each feature contains numerous coordinates which are also known as nodes. Left eye ranges from 37 to 42 landmarks and contains 6 nodes. Right eye ranges from 43 to 48 with 6 nodes. The last face feature, the mouth, has the highest number of nodes as it contains upper lips and lower lips. It ranges from 49 to 68 landmark points which contain 20 nodes.

To test the functionality of our pre-processing method, feature extraction, we experimented it on a picture. The result of our small experiment was arrays of coordinates as shown in Fig.3.6.

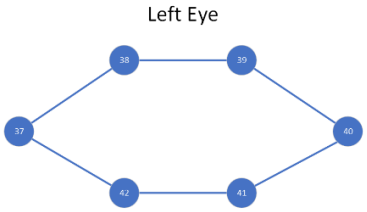
```
Mouth
[[2625.0, 1763.0, 283.3440355821878, -176.44889299665414, 2740.0, 1721.0, 178.05348213388035, -160.46089938802487,

Left Eye
[[3145.0, 1127.0, 115.83633089646612, 179.58780725859773, 3225.0, 1081.0, 57.654623019802784, -128.42700501352672,

Right Eye
[[2556.0, 1078.0, 109.07960905279735, 177.81086023942254, 2626.0, 1033.0, 56.465574566377214, -133.68446175848314,
```

Fig. 3.6: Examples of face landmarks for feature extraction

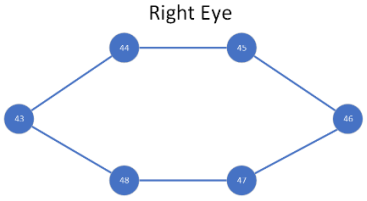
With the ranges between facial features, we represent the nodes connectivity using adjacency matrix. Adjacency matrix is a way of representing a graph as a matrix of

Booleans of ‘0’ and ‘1’. ‘0’ represents false connection, while ‘1’ shows true connection. Each coordinate of the eye corresponds to another coordinate of the eye. For example, the left eye, the coordinate of 37 are connected to 38 and 42 as illustrated in Fig.3.7. In adjacency matrix, we see on the right of the graph, it is represented as ‘1’ with true that the coordinate corresponds to another coordinate. This applies to all the other coordinates.

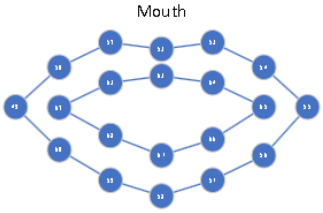
FER2013 dataset does not only contain numerical facial expressions of emotions but it also contains usages and pixels for the images. Regarding the type of usages, we grouped them into two categories: Training and testing sets. Within each of the categories, there are allocated pixels of images that we pre-processed with the use of the facial landmarking method. In order to ensure that our grayscale images provide the best quality images or pixels, contrast limited AHE (CLAHE) were used. It is a method that employs histogram equalization in small patches with high accuracy and contrast limiting (Senaratne, 2020).




| | 37 | 38 | 39 | 40 | 41 | 42 |
|---|---|---|---|---|---|---|
| 37 | 0 | 1 | 0 | 0 | 0 | 1 |
| 38 | 1 | 0 | 1 | 0 | 0 | 0 |
| 39 | 0 | 1 | 0 | 1 | 0 | 0 |
| 40 | 0 | 0 | 1 | 0 | 1 | 0 |
| 41 | 0 | 0 | 0 | 1 | 0 | 1 |
| 42 | 1 | 0 | 0 | 0 | 1 | 0 |




| | 43 | 44 | 45 | 46 | 47 | 48 |
|---|---|---|---|---|---|---|
| 43 | 0 | 1 | 0 | 0 | 0 | 1 |
| 44 | 1 | 0 | 1 | 0 | 0 | 0 |
| 45 | 0 | 1 | 0 | 1 | 0 | 0 |
| 46 | 0 | 0 | 1 | 0 | 1 | 0 |
| 47 | 0 | 0 | 0 | 1 | 0 | 1 |
| 48 | 1 | 0 | 0 | 0 | 1 | 0 |



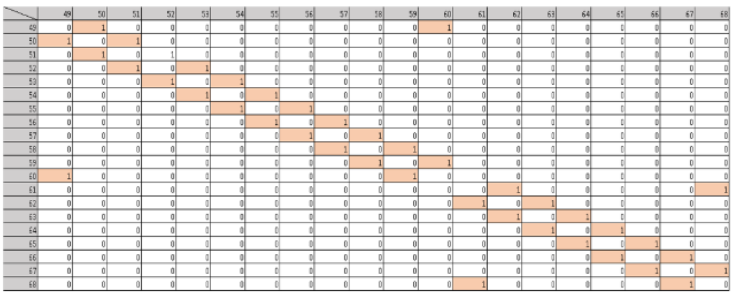



Fig. 3.7: Adjacency matrix representation of human facial features

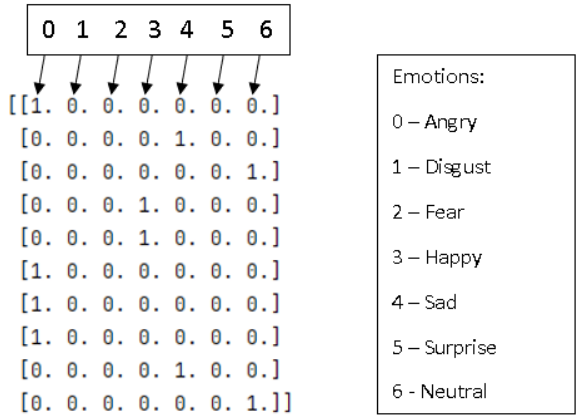



Fig. 3.8: One-hot encoding of emotions

### 3.2.2 Algorithm with Facial Landmark

Our Simple Deep Neural Network model consist of five dense layers. The parameters of the dense layers consist of unit, activation function, and input shape. Our first dense layer acts as input layer. Each of the dense layer within the network consists of unit size 128. The unit parameter defines the size of the output from the dense layer with activation function, it introduces the nonlinearity into the networks so that the networks can obtain the relationship between the input and output values. We set the next three dense layers with the rectified linear activation (ReLU) function as the input layer.

The last dense layer is different to the other layer as it is the prediction layer of the network. We set the unit parameter to the number of classes of emotions which is seven emotions. The activation function is set to softmax and outputs the emotion label and the probability of the predictions.

The next algorithm that we employed is Graph Convolution Neural Network (GCN). As previously mentioned, Graph can be described as $\mathbf{G} = \{\mathbf{V}, \mathbf{E}\}$, where **V** represents the vertices also known as nodes, **E** shows the edges also known as the connection between nodes as illustrated in Fig. 3.10 where the edges can be directed or undirected. It is similar to the CNN network; however, it takes number of nodes and number of input features of each of the nodes as input. In addition, it takes adjacency matrix (A). Neural network

layer can be written as

$$H^{(l+1)} = f\left(H^{(l)}, A\right) \tag{3.1}$$

where $A$ is the adjacency matrix, $H^{(l+1)}$ is feature representation at layer $l$+1 and $H^{(l)}$is the feature representation at layer $l$. One of the operations of GCN is propagation or message passing, which is presented as

$$f\left(H^{(l)}, A\right) = \sigma\left(AH^{(l)}W^{(l)}\right) \tag{3.2}$$

where $W^{(l)}$ is a weight matrix for the $l$-th neural network layer, $\sigma$ is a nonlinear activation function. The size of the second dimension of the weight matrix will determine the number of features at the next layer. This is similar to the filtering mechanism of CNN. The input adjacency matrix is symbolized as **A**, which represents the edges between each node that enables the model to learn based on each node connectivity.

Our network consists of five layers of GCN, one layer of mean pooling and 2 fully connected layer which can be illustrated in Fig. 3.9. We utilize Python library called StellarGraph, which a library that provides algorithm for graph machine learning based on multiple types of graphs and different task of graph and/or node classification.

In our model, there are various parameters to be taken into account, which includes layer sizes, activation, drop out, and unit size. Similarly, to CNN, the layer sizes are the filters for the input from the extracted features of our data pre-processing. We set our all of our layers to be 128 with the nonlinear function of ReLU. To prevent over fitting, we set our drop out to be 0.5. The next is the mean pooling layer, similarly with CNN, the mean pooling layer takes average of all the other layer. In our model, we employed one mean pooling layer that consists of one dense layer. The dense layer has its own parameters such as unit and activation function. Unit parameters must contain positive integer. It is the number of output nodes that the dense layer should return as an output.

Lastly, we employed two fully connected layers. Similarly, to the pooling layer, the fully connected layer takes two parameters unit and activation function. Along with two layers, the first of the layer unit are set to 32 with the activation function of ReLU. The

last of the layer is set to unit of 7 with the activation function of softmax. The last layer of the fully connected layer is where the emotion predictions are made for the 7 types of emotions. The softmax layer is the probability calculator for the emotions that is predicted by our model.

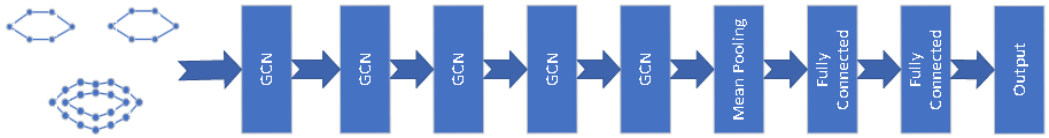



Fig. 3.9: The representation of GCN model

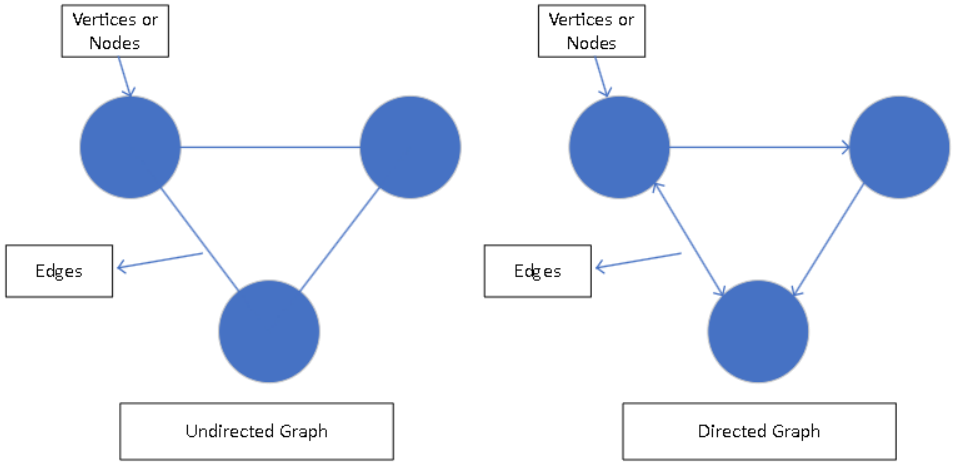



Fig. 3.10 The representation of directed and undirected graph

## 3.3 Time Series Analysis

Time series analysis is thought as a method of representing our accuracy result. Before outputting our accuracy from the training of our model, we firstly export the evaluation of our training model into a log file. It was fulfilled with the use of various methods mentioned before, such as Early Stopping, ReduceLROnPlateau, and Model Checkpoint. The log file contains the number of epochs, accuracy, validation accuracy and validation loss.

Our methods for time series analysis are decomposition, autocorrelation, and ARIMA (Auto Regressive Integrated Moving Average). Decomposition is a method of

time series analysis that is employed to visualize the trend in the data. In this project, it is employed to visualize the trend of accuracies across the number of epochs. At the same time, autocorrelation is employed to determine whether our data are correlated to the number of epochs if we want to increase them by a certain amount. We make use of multiplicative decomposition algorithm for determining the trend of our data. The algorithm of multiplicative decomposition is as follow:

$$y_t = T_t \times S_t \times R_t \tag{3.3}$$

where $y_t$ refers to the time series data, $T_t$ is the trend-cycle component, $S_t$ is the seasonal component, and $R_t$ is residual or remainders (Radecic, 2021) (Hyndman & Athanasopoulus, 2018) (Plummer, 2020). We utilize all the parameters while plotting our decomposition time series result. With the equation mentioned, we rearrange them to calculate for different parameters.

Autocorrelation outputs are measured in a unit called lag (Prabhakaran, 2019)( Peirre, 2021); it is a measure to show how correlated the data is. In our case, it is between epoch and accuracy. We utilize the autocorrelation function (ACF) with the utilization of lag (Brownlee, 2017)(Hyndman & Athanasopoulus, 2018). The ACF is set as eq. (3.4):

$$r_k = \frac{\sum_{t=k+1}^{T} (y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^{T} (y_t - \bar{y})^2} \tag{3.4}$$

where $r_k$ presents the autocorrelation value, $T$ for the length of the time series, and $y_t$ for the dataset; for example, $r_1$ measures the correlation between $y_t$ and $y_{t-1}$.-We applied the ACF function with the utilization of tsaplot library with the number of lags of 50 for determination of correlated data or partially correlated data. Prabhakaran (2021) states that ARIMA is a method that is used for forecasting future values. The equation of the ARIMA is set as eq. (3.5):

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_p Y_{t-p} \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + \cdots + \phi_q \epsilon_{t-q} \tag{3.5}$$

where $y_{t-1}$ refers to the first lag, $\phi_1$ is the coefficient of the first lag, $\alpha$ is the intercept the term. In another words, the predicted $y_t$ equals to constant plus linear combination Lags of $y$ plus linear combination of lagged forecast errors.

In this thesis, we took use of auto ARIMA known as SARIMAX. It makes use of a library within Python called pmdarima. Before applying the SARIMAX model, we tested the stationary of our dataset result with the employment of Augmented Dickey-Fuller (ADF). The stationary series works if the mean, variance, and covariance properties of the data does not vary with time. If a series is stationary, it should not exhibit a trend (Singh, 2018). We create the model to predict the future accuracy of our model. While creating the model, we take into consideration of the result of ADF test as its parameters. In this thesis, we apply the concept of Integrated (I) in the ARIMA which is represented by the parameter of 'd' with the value of NONE to set our non-stationary data.

The methods are from the stats model and pmdarima library in Python. The model is a Python package that provides statistical computing for statistical models such as time series analysis.

## 3.4 Real-Time Experiment

The experiment of our initial network with non-facial landmarking method is described in Algorithm (1). We start off with the initialization of our seven types of emotions, and Haar Cascade frontal face. Next, we take use of the OpenCV library that utilizes cv2 for camera detect faces from a webcam. We created a function that detects faces within the area of the webcam, if there is a frame within the camera, we extract the ROI within the frame, then resize and lastly place them into an array. Within each array of the ROI, we predict the emotions and output the probability of the emotions into a canvas.

The simple model is shown in Fig. 3.11 where we start with an input from a camera, then Haar Cascade classifier for frontal face detection, then non-landmarking algorithm of CNN and Mini Xception, the output of the prediction, and its accuracy.

Before the experiment using the landmarking method or method with Simple Deep Neural Network (SDNN), we had to initialize a couple of variables and functions that

allows the model and the experiment to be carried out. We start off with the initialization of our seven types of emotions, our CLAHE method, Dlib frontal face detector, and Dlib landmark predictor.

**Algorithm** 1 Real-Time Experiment

```
initialization;
while there is a frame from webcam do
    readFrame();
    if faceDetected then
        extractROI();
        resize();
        roitoArray();
    else continue
    end if
    for each ROI do
        predictEmotion()
        outputProbabilities(in percentage with 2 decimal places)
    save-probability-to-Canvas();
```

Result: Facial emotion recognition and predicted probability

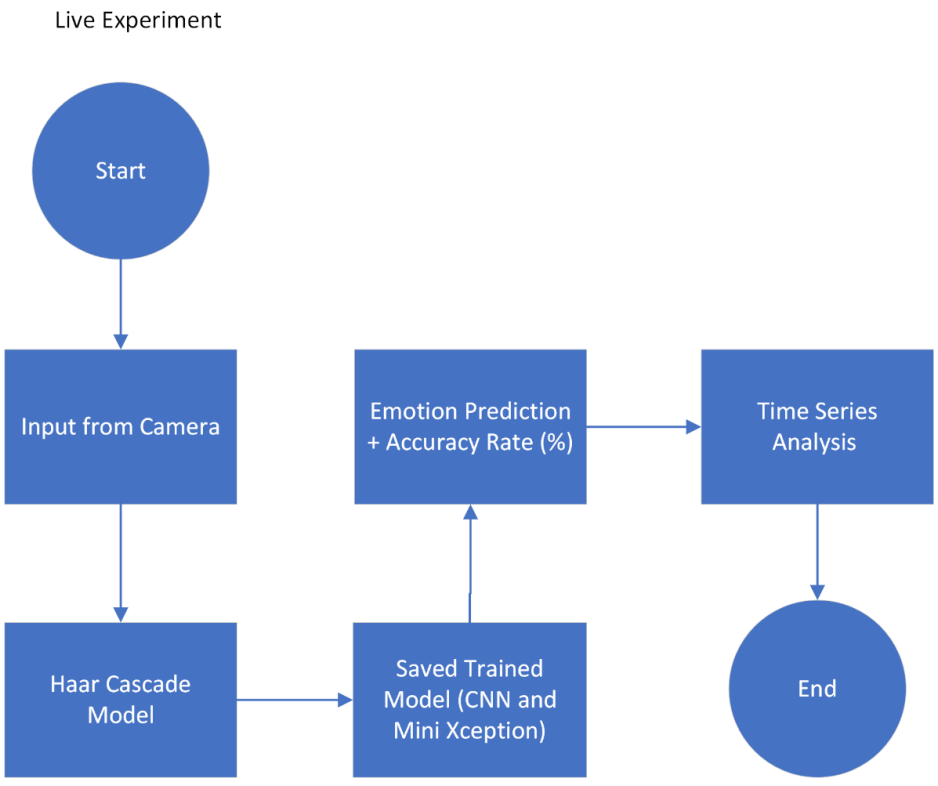



Fig. 3.11: The simple model of non-landmarking live experiment

Due to landmarking technique, we created multiple function or method to get and show the landmarking during live experiment. The methods are *rec_to_bb*, *shape_to_np* and *get_landmarks*. The *rect_to_bb* is to take the bounding predicted by using Dlib and converting it into the format of that can accept *x* and *y* coordinates. The *shape_to_np* is to convert *x* and *y* coordinates into a NumPy format, this takes use of the range of Dlib 68 points landmarks. It returns the coordinates in NumPy array format. Lastly, the *get_landmarks* make use of the *rect_to_bb* with the ranges of the 68 landmarks. The *get_landmarks* method will return vectorized landmarks positions in arrays. With the methods and variables initialized, we now carry out our real-time experiment.

The experiment of our following network with facial landmarking method is described in Algorithm (2). We make use of the webcam and OpenCV library for the real-time detection and prediction. The first step of the experiment was to initialize the OpenCV library for the webcam. While there is a frame within the camera, we applied the CLAHE to the face detected within the camera frame. The face detection is employed within the while loop and the employment of the Dlib frontal face detector. Within the frame that detects the face, we employed the Dlib landmark predictor, then converted the landmark points to the coordinates, then the landmark points are fed into our model that was trained based on the best result during training.

With the outputted emotions and its probability, we applied it to the ROI or bounding box of the OpenCV format. Then we created a loop that goes through all the coordinates and draw them on the face that is detected. Lastly, we outputted emotions and its probability on an OpenCV canvas. The basics of the model are seen in Fig. 3.12.

**Algorithm 2** Landmark Real-Time Experiment

```
initialization;
while there is a frame from webcam do
    readFrame();
    applyClahe();
    for faceDetected do
        faceLandmark();
        convertLandmarkToCoord();
        predictEmotion();
        outputProbability();
        convertDlibRectToROI();
        for eachCoord do
            DrawFaceOnBoundingBox();
            showEmotion();
            showProbability();
```

Result: Facial emotion recognition and predicted probability with Landmarks

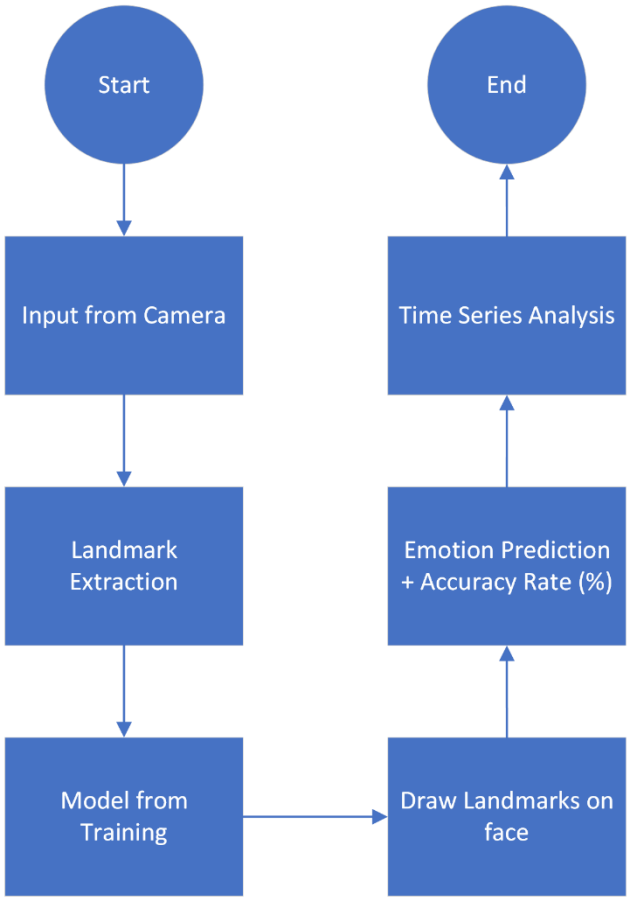



Fig. 3.12: Simple model of landmarking method in real-time experiment

## 3.5 Ablation Experiment

With our ablation experiment, we modified the models that will be the best model for our methodology: Xception and SDNN. With each model, we have increased and decreased the number of layers to determine which models with different layers would output a better performance. For the model implementation, we have applied a decrease of one layer and an increase by one layer. We trained each of our models with the epochs of 300, 400, 500, and 600. Our hypothesis is an increase in the number of layers would ramp up the performance of the model. The result of the ablation experiment is from the training of our model, which will be explained on further details in Chapter 4.

## 3.6 Database

The dataset for our experiment in this thesis is FER2013, which contains 35,887 grayscale facial images with seven classes of facial emotions having the resolution 48 × 48. In this thesis, we make use of 20% samples for model training. In this thesis, we take advantage of comma-separated values (CSV) file for the FER2013 dataset. The file contains facial expressions of emotion. This dataset contains 35,887 images, all of them are grayscale images. Within the 35,887 images, the dataset also has three types of usages: Training set, public test, and private test. The training set consists of 28,709 images, including 3,589 for public tests and 3,589 for private tests. We used

Human facial expressions of emotion are set from Class 0 to Class 6 to represent the seven classes of emotions: Angry (Class 0), Disgust (Class 1), Fear (Class 2), Happy (Class 3), Sad (Class 4), Surprise (Class 5), Neutral (Class 6). The dataset preprocessing method converted the pixels into integer format and created an array list to store the data type like a floating number. The list contains pixels of each emotion within the dataset.

Regarding the images of human facial emotions, we returned the number as a label that each image is classified as the emotions with the use of ‘0’ and ‘1’, ‘1’ represents the image that is classified as a facial expression of emotion.

# Chapter 4
# Results, Analysis and Discussion

*The main content of this chapter is to demonstrate and discuss the result of the training, testing, and real-time experimentation from our methodology. In addition, comparisons of the result with previous work are discussed.*

## 4.1 Introduction

In this thesis, we have conducted our experiments by using Python. Pertaining to our method for FER, we train our model by using the FER2013 dataset with various number of epochs for seven classes of emotions: Angry, disgust, scared, sad, happy, surprised, and neutral. The numbers of epochs we employed for our model training are 300, 400, 500 ,600. For our non-facial landmarking technique, we make use of Haar cascade for the frontal face detection and trained our first model, CNN, to classify the emotions.

To uplift our accuracy from our CNN model, we experimented two other models with the non-landmarking technique. The two models are Mini Xception and Visual Transformers (ViT).

In addition, we experimented with another method or technique of FER, with facial landmarking. We employed two models for the method, Simple Deep Neural Network (SDNN) and Graph Convolutional Network (GCN). With the landmarking method, we utilized Dlib for frontal face detection and facial landmark location prediction. For the training of our models that we employed, we presented the result in a bar graph representation and time series analysis. The real-time experiment result will also be presented with the metric of accuracy percentage. Our initial methodology of non-facial landmarking is with the utilization of CNN, Mini Xception and ViT model.

Our project is unique as we provide two types of methods and various algorithms or models to predict emotion and its probability or accuracy. In addition, our models have been modified with the employment of 5 blocks for CNN and Mini Xception and its parameters within the models, descending and ascending filters. Similarly, with our landmarking method and algorithm, we have modified our algorithm to have 5 layers or blocks. With the current progress of our project, we will implement a refined landmarking method which has been discussed in Section 3 and will be in further detail in the Future work section.

## 4.2 Training Result Analysis

One way of presenting our results is time series analysis. Time series analysis is a method to show the trend for time intervals, in our case, time intervals are the number of epochs. We employed various methods to show our result. They include Decomposition, Autocorrelation, and ARIMA. The time series analysis in our study is applied to represent the result of training and testing of our models.

Our hypothesis of the accuracy result is to be around 90.0% for the training if we increase the number of epochs with a downfall of longer time to train the model. In order to investigate our hypothesis and demonstrate our results, we make use of time series analysis methods.

Decomposition is a method of time series analysis that we applied to show the trend, seasonality, and residuals of our training result. We trained three of our algorithms with the use of different number of epochs, 300, 400, 500 and 600. From the training result illustrated in Fig.4.1, Fig.4.2 and Fig.4.3, we determined all the algorithms show Multiplicative Trend with Additive Seasonality with the residuals of each of the algorithm having more centered around one value and no distinct pattern. Multiplicative trend in this study refers to increase of accuracy gradually in the shape of a curve. On the other hand, Additive Seasonality refers to the linear season where the result is consistent over the number of epochs. The result shows that there is a trend of an increase in the number of epochs will increase the accurate rate.

Autocorrelation method is a method we applied to investigate our hypothesis and to further investigate our training trend. It is used as an initial step to confirm the trend that we discovered during training before using ARIMA. Autocorrelation is employed to determine the relationship of two variables, in our study, it is the accuracy rate and the number of epochs. We utilized the use of Autocorrelation Function for all our algorithms.

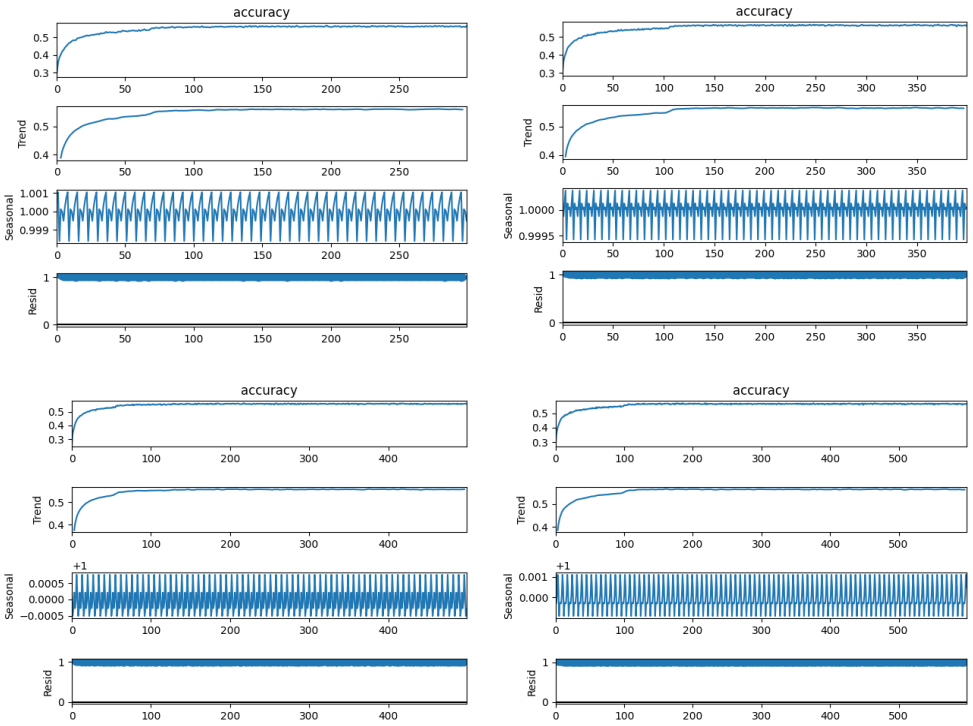



Fig. 4.1: The result of decomposition of CNN model with numerous epochs, 300 (top left), 400 (top right), 500 (bottom left) and 600 (bottom right).

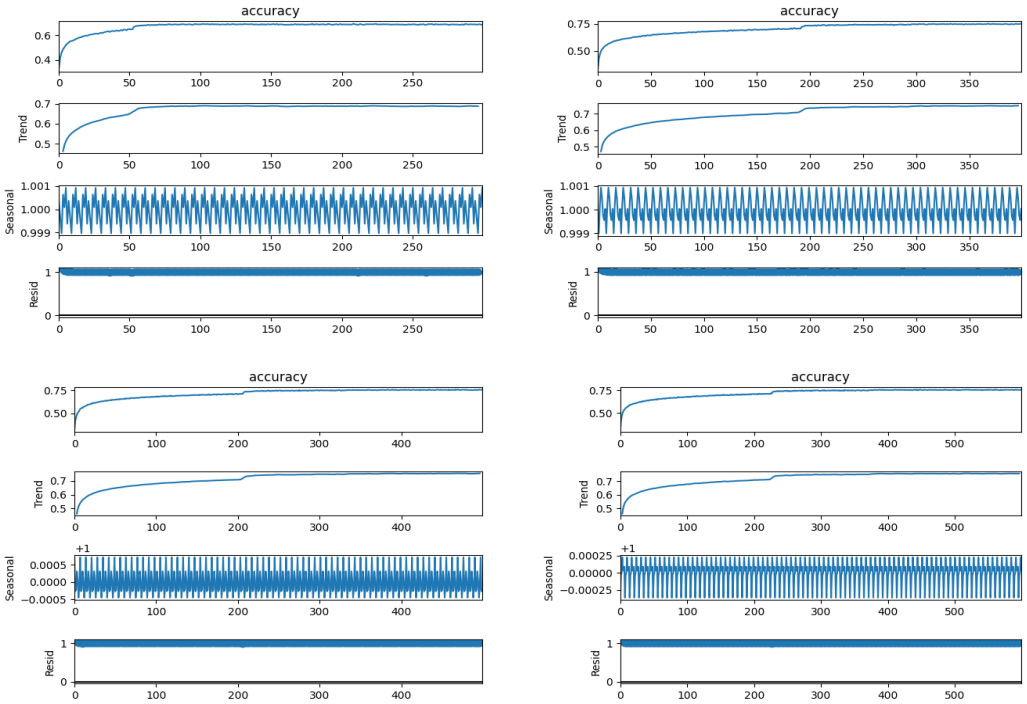



Fig. 4.2: The result of decomposition of Mini Xception model with numerous epochs, 300 (top left), 400 (top right), 500 (bottom left) and 600 (bottom right).

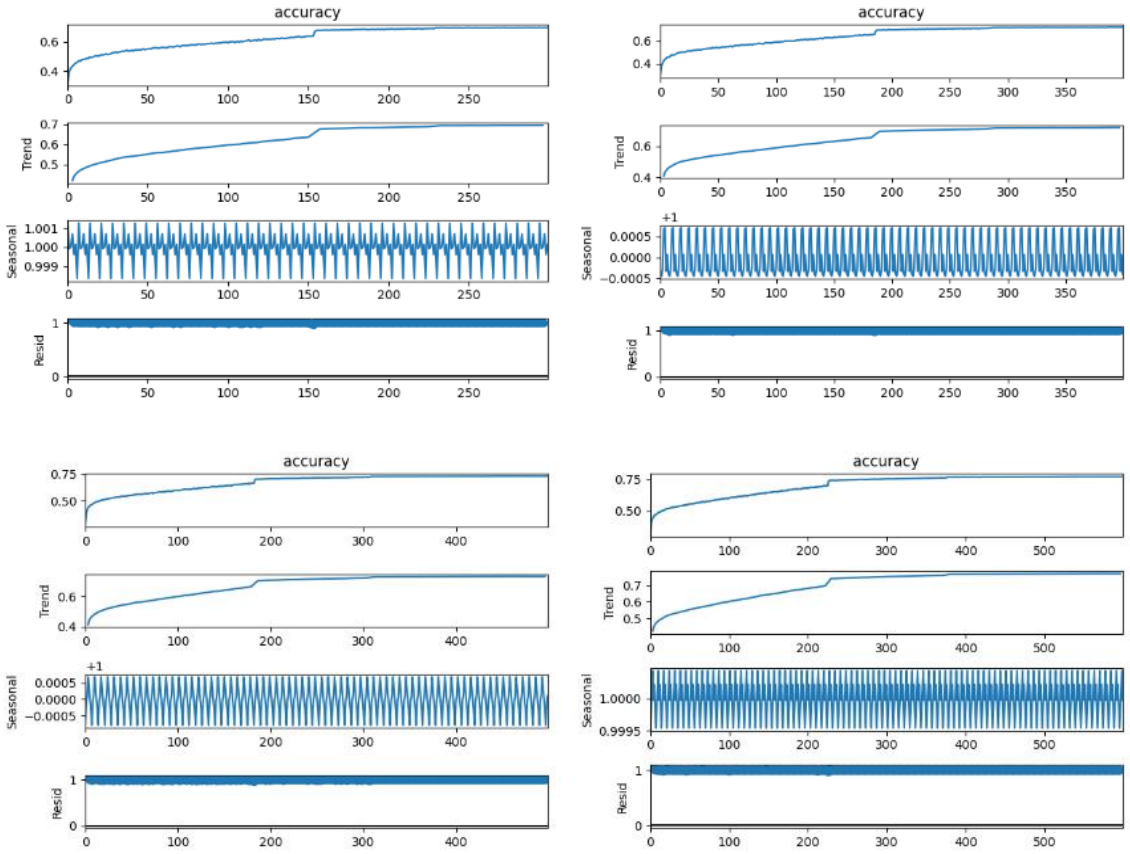



Fig. 4.3: The result of decomposition of SDNN model with numerous epochs, 300 (top left), 400 (top right), 500 (bottom left), and 600 (bottom right).

The result demonstrates that the data is autocorrelated. For each of the algorithms, we discovered that the number of epoch and the accuracy rate are correlated. It shows positive correlation and negative correlation. Positive correlation means that an increase in one predict the increase of the other, while negative correlation means that an increase in one value's predict the decrease of the other. Therefore, with the result of autocorrelation method, we determined that the number of epoch and the accuracy rate are correlated as the change of one will affect the other, in other word, the increasing number of epochs will affect the accuracy rate of our data.

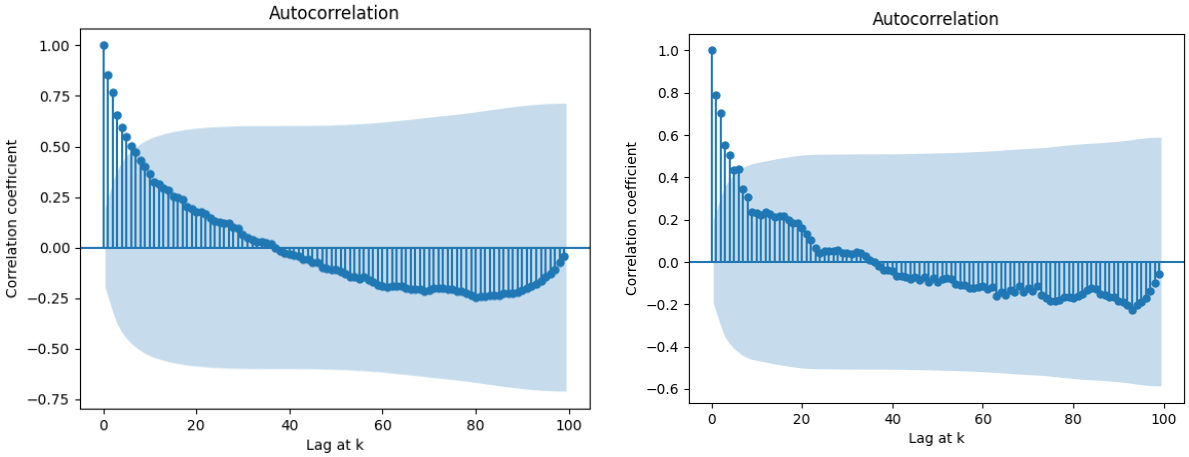



Fig. 4.4: The result of autocorrelation of non-facial landmarking method with 100 epochs, CNN (left) and Mini Xception (right)

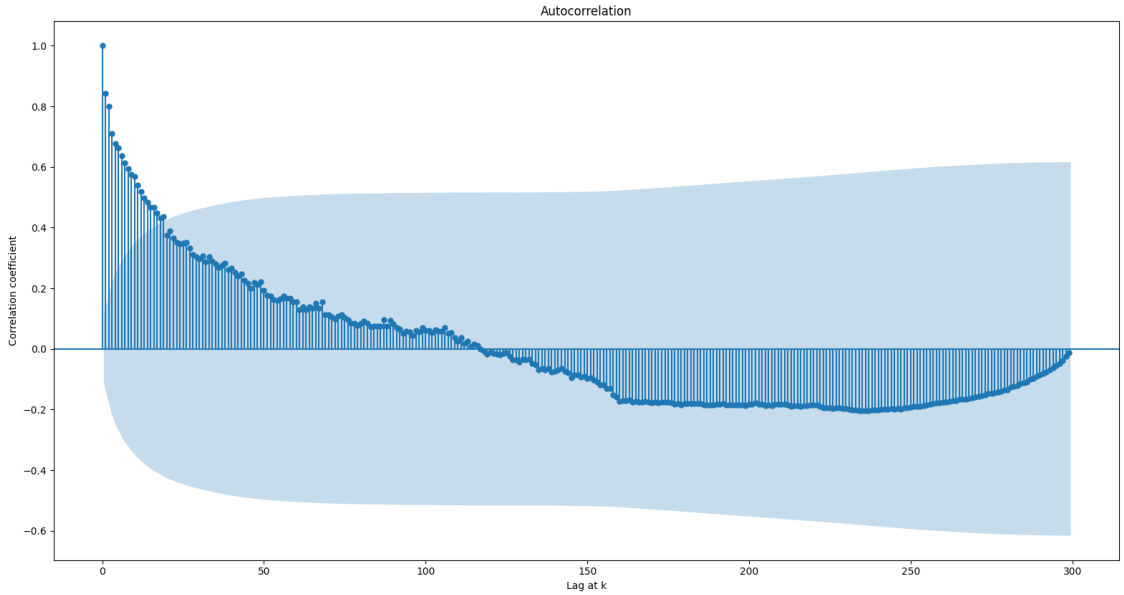



Fig. 4.5: The result of autocorrelation of facial landmarking method with 100 epochs

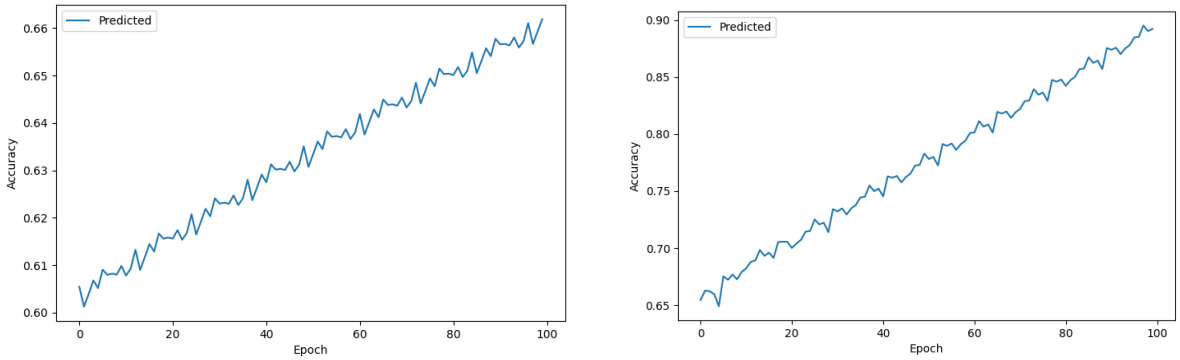



Fig. 4.6: The result of ARIMA of non-facial landmarking method with the next 100 epochs, CNN (left) and Mini Xception (right)

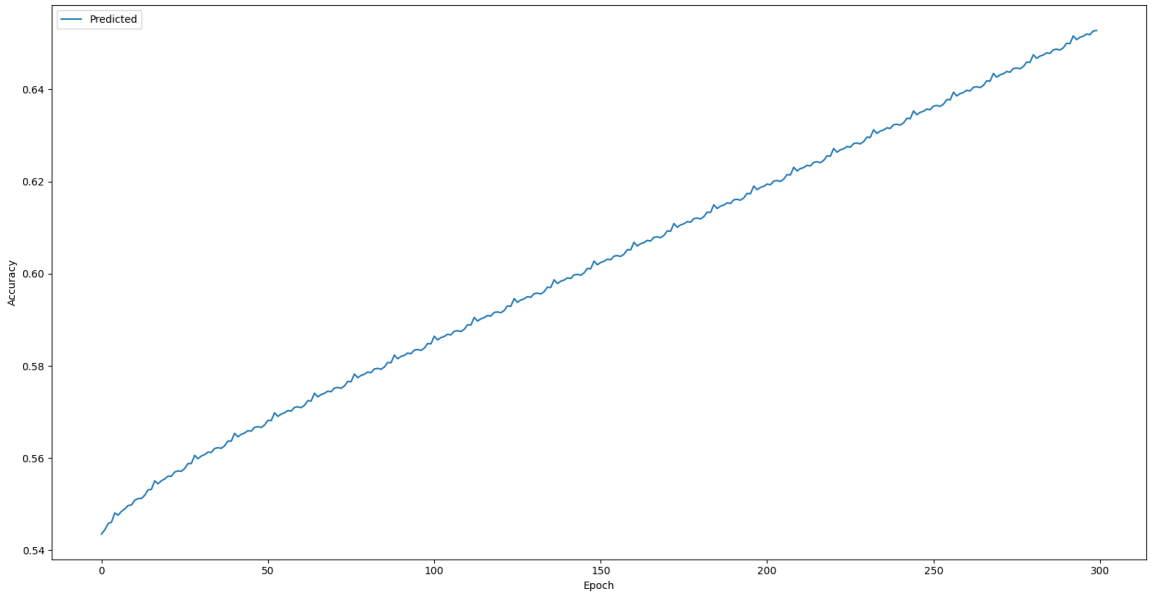



Fig. 4.7: The result of ARIMA of facial landmarking method with the next 100 epochs

With the knowledge of correlated number of epoch and accuracy rate, we further investigate the trend. We employed the ARIMA method. With the ADF test, we determined our dataset is non-stationary, therefore we used the value of 'd' in the ARIMA model. The result of the ARIMA method shows a positive increase of accuracy over the next 100 epoch. It again indicates that with an increase of the number of epochs will increase our accuracy up to approximately 90% from our original accuracy for our CNN, Mini Xception and the SDNN model. However, with the training result as shown in Fig.4.8, the CNN algorithm does not increase over the number of epochs, but an increase in validation accuracy can show that there is a trend. The result of SDNN is different to the CNN model, the algorithm has produced a similar result to its training of the next 300 epochs, where it has produced higher accuracy of 77% when trained with 600 epochs.


Fig. 4.8: The result of training CNN model with numerous epochs, 300 (top left), 400 (top right), 500 (bottom left), and 600 (bottom right).


Fig. 4.9: The result of training the Mini Xception model with numerous epochs, 300 (top left), 400 (top right), 500 (bottom left) and 600 (bottom right).


Fig. 4.10: The result of training SDNN model with numerous epochs, 300 (top left), 400 (top right), 500 (bottom left) and 600 (bottom right).

For the training of our ViT Model, we trained our model with various training arguments. The training is split into two parts: Training and evaluation of each epoch that the model is trained. When it comes to training, the number of epochs that we iterate through for the model is six epochs. The training dataset contains 24, 402 data, which our model was trained on. The result of the first training was exported in terms of loss, the model achieved an initial training loss of 1.12 for the first epoch.

The evaluation of the model is with the use of 3,589 data samples. While running the evaluation of six epochs, we were able to get the best results. The results from our evaluations are measured in loss and accuracy. The result of evaluation from the training of our model was 1.01 and 63% accuracy. As we see that there is an improvement of loss of 0.11 from the training of our model on the first epoch. The process continues till the model reaches epoch of six.

At the last epoch, the result of the training has outputted a loss of 0.21, this is an improvement of loss rate 0.91 from the first epoch. When it comes to evaluation, the model achieved a loss rate of 1 and accuracy of 69%. It is an increase in accuracy from

the first evaluation epoch by 6%.

After the training of our model, it is time to test the dataset. The test dataset contains 3,589 rows of data. While running the test after training, we were able to get a loss of 0.95 and an accuracy of 70%. From the training, evaluation, and testing processes of our model, we see that our accuracy rate has consistently stayed in the range of 60% to 70% with the lowest accuracy 63% on the first epoch of evaluation and the highest accuracy 70% for the testing. The loss for our training has decreased significantly from 1.12 to 0.21 by the end of the six epochs, however, there is a pattern for the evaluation loss. The pattern is similar to the one on the autocorrelation and ARIMA method of time series analysis where it dips to a point of decrease and increases in the rate when it reaches a certain number of epochs. The result of the testing can also be represented in a confusion matrix where predictions of facial expressions of emotion and its true classification score are presented.

Our test dataset contains 3,589 data. Based on the prediction on the test dataset, our confusion matrix has produced results as illustrated in Fig.4.11. The result of the confusion matrix shows the predicted emotions and the true emotions. The score of the matrix is the number of times that the predicted emotion is true positive. True positive in this case refers to when the algorithm has predicted emotions correctly. From our testing, we see that the highest score is when algorithm correctly predicted the Happy emotion with the score of 789. In comparison to the other emotions, emotion Neutral has produced a score of 420, followed by Sad emotion with the score of 347, then Surprise emotion with score of 324, then Anger emotion with 308 and Fear with 290. However, the algorithm has difficulty in predicting the Disgust emotion as it has only produced a score 31.


Fig. 4.11: Confusion matrix of ViT with the test set

Furthermore, false negative means that when the true emotion has been predicted wrong by our algorithm. For example, for the True Neutral emotion, it is seen that it has predicted wrongly with the emotion Sadness with the confusion matrix score of 112. It means that the test dataset, the model has wrongly classified neutral emotion with sadness 112 times.

In addition, we have a few emotions that have an exceptionally low score when it comes to false negative classification, which includes Happiness, surprise, disgust, and neutral. For emotion Happiness, it has classified false negative values of zero when it comes to classifying Disgust emotion, it also applies to Surprise when it was classifying Disgust. For Disgust, it has classified four other emotions with the score of 1, including emotions Fear, Happy or Happiness, Surprise and Neutral. Lastly, for emotion Neutral, it has false negative score of 2 while classifying emotion Disgust. As we see, there is a pattern of low values of false negative with the classification of emotion Disgust. There are other number of False Positive with the range of scores from three to seven.


Figure 4.12 Training and testing result of ViT

The training processes of our experiment are illustrated on Fig.4.8, Fig. 4.9, Fig.4.10, and Fig.4.11, we ran the training through numerous numbers of epochs. The result shows that if we increase the number of epochs, there is an increase of accuracy. The accuracy

of CNN models did not increase over the number of epochs however the validation accuracy increases from 60% to 62%. We also see that with the Mini Xception there is a significant increase from 68% accuracy to 76% accuracy. It also applies to the facial landmarking model, SDNN, where it increases from 69% accuracy to 77% accuracy. For the ViT, the result of the training had an initial accuracy of 63%, as we increased the number of epochs by one, the accuracy increased to around 69.4% at epoch 5, however a slight decrease of accuracy by 0.4% when increased to 6 epochs, the decrease of accuracy is by a small margin of percentage. The training loss of the ViT, the rate has decreased tremendously from 1.12 to 0.21.

## 4.3 Experimental Result

We tested the seven classes of facial expressions of emotion and presented the result using accuracy. Again, we have two methods of FER with various algorithms or models. For non-facial landmarking, we took use of CNN, Mini Xception and ViT. On the other hand, for facial landmarking, we used SDNN and GCN.

With the CNN model, we were able to predict three emotions with accuracy over 50%. The highest classification accuracy was 90%, for the emotion Happy, followed by 59% for emotion Sad and 56% for emotion Surprised. However, with three classes of emotions with one acceptable accuracy, misclassification occurs while determining emotion Angry and emotion Disgust. As illustrated in Table.4.1 the lowest accuracy is 20% for emotion Angry emotion and 2% for emotion Disgust.

In order to lift the accuracy, we tested the Mini Xception model in the live experiment. The Mini Xception model produced higher accuracy where six classes of the emotions have produced accuracies over 50%. These are three more emotions than the CNN model. As shown in Table 4.1, the model achieved the highest accuracy of 99% for emotion Happy, then followed by 95% for emotion Disgust, 93% for emotion Neutral, 71% for emotion Angry, 68% for emotion Fear, and 56% for emotion Surprised. While comparing the result to CNN, we see that we achieved an improvement of 51% while classifying emotion Angry, 93% for emotion Disgust, 37% for emotion Fear, 9% for

emotion Happy and 58% for emotion Neutral. The lowest accuracy of the experiments using this model was emotion Sad with accuracy 47%. Overall, compared to the lowest of CNN, it has provided a stable classification.

Table 4.1: Result of live experiments

| **Result of Live Experiment** | | |
|---|---|---|
| **Algorithms** | **Emotions** | **Accuracy %** |
| **CNN (Our Modified)** | Angry | 20.0 |
| | Disgust | 2.0 |
| | Fear | 31.0 |
| | Happy | **90.0** |
| | Sad | 59.0 |
| | Surprise | 56.0 |
| | Neutral | 35.0 |
| **Mini Xception (Our Modified)** | Angry | 71.0 |
| | Disgust | **95.0** |
| | Fear | 68.0 |
| | Happy | **99.0** |
| | Sad | 47.0 |
| | Surprise | 56.0 |
| | Neutral | **93.0** |
| **SDNN (Our Modified)** | Angry | **90.0** |
| | Disgust | 38.0 |
| | Fear | 43.0 |
| | Happy | **96.0** |
| | Sad | 51.0 |
| | Surprise | 72.0 |
| | Neutral | 53.0 |

Although the models with non-landmarking technique have provided a better experiment and training accuracies, we still encounter misclassification and low accuracy rate. We employed another methodology of facial landmarking to solve our encounters. The algorithm for this method is SDNN. The purpose of this model is to test our facial landmarking technique. While running the experiment with the saved model, it was able to achieve accuracy of 96% for emotion Happy, and 90% for emotion Angry as shown in Table 4.1. With the idea that the model can produce equivalent results to the other two models, we believe that with the employment of GCN model and the use of nodes, graph classification of facial features can uplift the results of our model.

## 4.4 Ablation Experimental Result

The results of our ablation experiments are from model training that will be presented in accuracy and loss as the metrics based on the training dataset and test dataset. We split the dataset into the ratio 80% for training dataset and 20% for test dataset from the pre-processed FER2013 dataset.

### 4.4.1 Mini Xception

Regarding our Xception model, with a decreasing number of layers of one from five to four layers, the model training with 300 epochs has produced an accuracy 72.7% with a loss of 0.75. For the testing, we achieved accuracy 63.7% with a test loss rate 1.07. With the last training epoch 600, we were able to get training accuracy 77% with a loss 0.63. While testing the model for the test set, we achieved accuracy 64.3% with a loss 1.1. For the training there is an increase of 4.3% in accuracy and decrease in loss of 0.12. However, the increase in testing accuracy is less by 0.6% and an increase in loss by 0.03.

We then increased the layer of the model by one layer from five layers to six layers, with 300 epochs, we achieved a training accuracy 88.4% with loss 0.32. While testing the model with the test set, we achieved accuracy 65.1% and loss 1.39. With our last epoch of 600, we were able to achieve a training accuracy of 89.5% with a training loss of 0.29.

With the test set, we were able to achieve 64.9% accuracy and 1.43 loss.

### 4.4.2 Simple Deep Neural Network (SDNN)

For our SDNN model, while decreasing the number of layers from five to four layers at 300 epochs, we were able to get an accuracy 66% with a loss 0.91 for the training set, 53.4% for the testing set, and 1.33 for the loss. With 600 epochs, we were able to achieve the same accuracy of 66% with an increase of loss of 0.94. For the testing, we achieved a higher accuracy of 54% with is a slight decrease with a loss 1.32.

The following ablation method was increasing the number of layers, in our case, it is to six layers. For the training with 300 epochs, we were able to achieve a higher accuracy 72.5% and a loss 0.76. For the testing set, we were able to get accuracy of 54% which is similar to the previous method with the loss 1.41. For the 600 epochs, the results are significantly better by outputting training accuracy of 80% with a loss of 0.58 as illustrated in Fig. 4.13. For the test, the accuracy seems decrease in accuracy of 53% with loss rate 1.72.

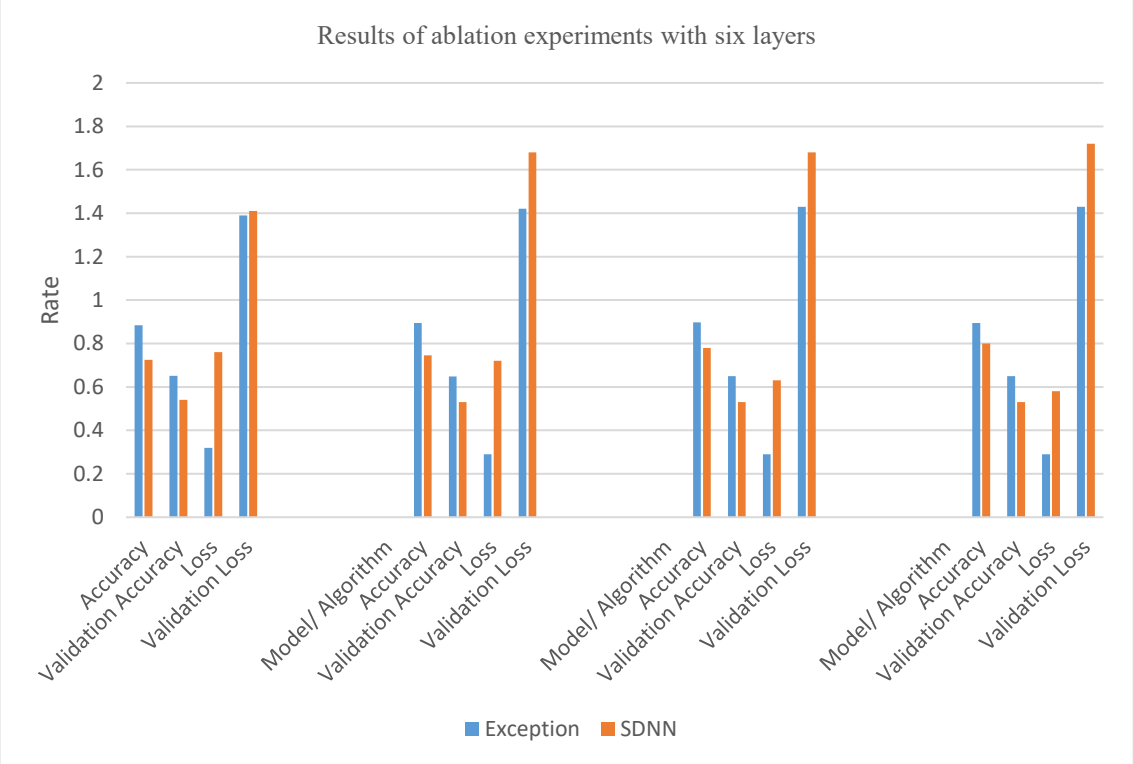



Fig. 4.13: The result of ablation experiments with 6 layers (Starting with 300 epochs on the left side and 600 epochs on right side)

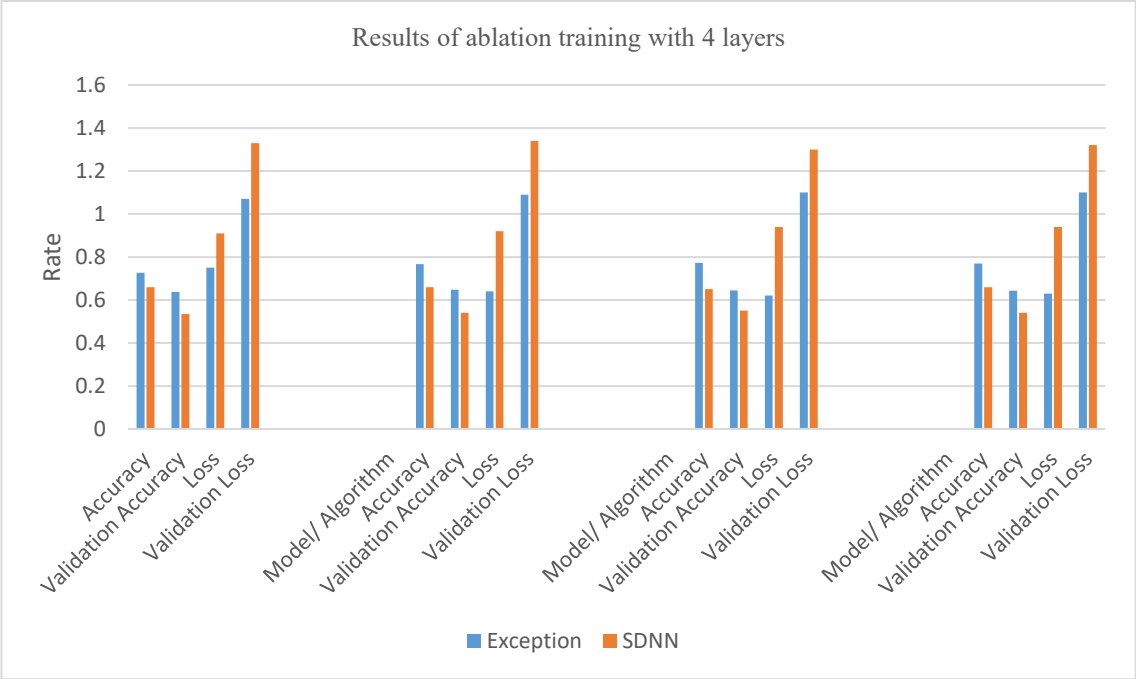



Fig. 4.14: The result of ablation experiment with 4 layers (Starting with 300 epochs on the left side and 600 epochs on right side)

## 4.5 Discussions

With the result that we have outputted, we believe that our CNN algorithm has produced lower accuracies in FER compared to other CNN algorithms (Mehendale, 2020). Our model is hard to distinguish between the facial expressions of emotions: Disgusted, scared and surprise. Due to the small number of epochs (Sharma, 2017), it is insufficient to achieve reasonable accuracy. Therefore, a higher number of epochs is necessary to reveal the capability of our proposed model to achieve higher accuracy. Another reason is the misclassification.

We employed nine layers of CNN with 35,887 images and image augmentations for training and testing to classify seven types of emotions. Our model is unique because our filters increase as the deeper the network goes. We employed Haar cascade and the modified CNN model for our experiments. While running the live experiment, we utilized various facial emotions to ensure that our model could predict emotions when the emotions are shown. If we show the emotion Happy, we achieve an accuracy rate of 90%.

With a low accuracy of training result, we explored other neural networks such as Mini Xception and SDNN with a goal of uplifting the training accuracy.

Owing to the popularity of Transformer models in deep learning, we employed the basic Transformer model, it is referred to as Visual Transformers (ViT). While training our ViT model, we again used the same dataset FER2013, but less epochs due to time and resource limitations. From the model training with 6 epochs as shown in Fig. 4.12, the first epoch of training has resulted 63% accuracy and a loss rate 1.01.

As the training epoch increases, we see that there are obvious improvements to the accuracy and loss. In epoch 2, we see that the accuracy has increased by approximately 4% from 63% to 67% and improvement in loss rate by approximately 0.1 from 1.01 to 0.91. For epoch 3, the model achieved accuracy of 68% and loss of 0.91. It is an increase of accuracy of 1% from previous epoch and no improvement of loss. With Further improvement in accuracy in epoch 4, we achieved accuracy of 69.2%. It is an increase of accuracy by 1.2%, however, the loss has increased from 0.91 to 0.93. It also applies to the other epochs, where the accuracy improves by approximately 1% each as we increase the epoch by 1.

We assume that if we train the model for 300 to 600 numbers of epochs, we would gain a better accuracy compared to CNN or Mini Xception, this is similar to the previous work where ViT tends to perform better than CNN or Mini Xception with less computational resources. With the epoch 5, we see only a slight increase in accuracy 0.2% in accuracy from 69.2% to 69.4% and an increase in loss 0.07 from 0.93 to 1. Although, there is an increase in accuracy, we see that in epoch 6, the accuracy seems to decrease by 0.4% from 69.4% to 69% with no change in loss. However, the increase and decrease of loss and accuracy is only by a small percentage or rate.

While evaluating our trained model with the test set, we see that our accuracy reached 70%, which is similar to the one produced by Mini Xception and CNN. The difference is that we only trained our ViT model with 6 epochs and on the other hand we trained CNN and Mini Xception with four epochs: 300, 400, 500 and 600. With only 6 epochs of training, we are able to get similar result with a smaller number of epochs and a small loss rate of training. We believe that an increase of number of epochs, the accuracy

outputted by our ViT can produce a higher accuracy compared to CNN and Mini Xception for the non-landmarking technique.

The CNN models (Mehendale, 2020) with Cohn-Kanade and additional datasets that contain around 10,000 images, were able to produce the FER accuracy up to 96.0% with two 4-layer CNN and kernel matrix of 3×3. While comparing our results with others, we employed one extra layer of CNN with 35,887 images and image augmentation.

With the idea of uplifting accuracy, we implemented an initial methodology of utilizing facial landmarking and graphs. With the current progression, we have implemented a Simple Deep Neural Network (SDNN) to test our facial landmarking methodology. We trained our SDNN model with several epochs. We started with 300 epochs, we were able to achieve accuracy of 69% and validation accuracy of 54%. It is again an increase of accuracy from our original methodology.

We further investigated and implemented the same algorithm however with increased number of epochs. We increased the number of epochs 400, 500 and 600. With the increase, we see that there is a trend of increasing accuracy rate as shown in Fig. 4.10. With the base line of 69% accuracy from 300 epochs to 77% accuracy from 600 epochs. This result correlates with the ARIMA model as it predicts that the accuracy will increase as the number of epochs increases however not as high accuracy as predicted.

We believe that the use of Simple Deep Neural Network is different. With the facial landmarking methods, we are able to capture various features of the face such as eyes, eyebrows, mouth, jawline, and nose. These features determine the changes in human emotions (Heaven, 2020). For example, with emotion Happy, the landmarks of mouth would be different from face sad.

Whilst looking at the results of the training and experiment of our model, we were able to achieve some high accuracies for all the models. There are a plenty of ways of increasing the accuracies of the model (Pawar, 2018), (Zita, 2022), (Dsouza, 2021) that includes adding more layers, changing the image sizes, increasing epochs, decreasing colour channels, underfitting and overfitting preventions.

In this thesis, for the CNN model, we have added a multilayer CNN model that includes resizing the images, increasing epochs from 300 to 600, various image

augmentation methods are shown in Fig.3.1, such as preventing the overfitting and underfitting by using training method. This also applies to the Mini Xception. For our SDNN model, we did not augment our dataset, which deter the training and the outcome accuracy of our model as shown in Fig. 4.10 and Table. 4.1.

In addition, our SDNN model does not contain many layers as our CNN and Mini Xception model. Although, we have not used the two methods of adding layers and data augmentation, we did use an increase in number of epochs from 300 to 600 and the prevention of overfitting with the use of early stopping. For the color channel, because our data samples are grayscale images, all the models already have taken advantage of this method.

In order to further investigate the need of improving accuracy, we employed ablation experiment on the two models, that provide the best model for each of the proposed methodology, Xception for non-facial landmarking and SDNN for facial landmarking.

As shown in Fig. 4.13 and Fig.4.14, if we increase the number of layers within then SDNN model, the training result has increased to 80% accuracy. It also applies to the Xception model where an increase in one layer has resulted in accuracy up to 89.5%. The increase of accuracy is due to the increase in the number of layers in the network. By increasing the number of layers in the network, it allows the network to distinguish the differences in the pixels of images for the different types of emotions.

# Chapter 5
# Conclusion and Future Work

*In this chapter, we conclude the findings of our project, discuss the limitations, and propose the future work of our research project.*

## 5.1 Conclusion

In this thesis, we proposed four types of models or algorithm that is used for FER. The initial model is CNN with the use of Haar cascade for frontal face detection for the live experiment. We modified our CNN models with five blocks. Each block consists of 2D convolutional layers, two batch normalizations, one rectified linear unit (ReLU), one average pooling, and one dropout for the last block. On the training of our CNN, various parameters of the proposed method have been considered. One of the critical parameters is the epoch. We used numerous numbers of epochs, starting with 300 epochs. The training time requires long hours for the algorithm and the numerous epochs to be trained.

While running the live experiment using our trained CNN model, we were able to get the prediction of emotions with accuracy. One of the highest detected emotions is up to 90.0% accuracy rate for the emotion Happy. While comparing our CNN algorithm to other CNN algorithms, our accuracy tends to be lower. Therefore, further improvements are needed. When we did the live experiment with the Mini Xception model, we were able to get a high accuracy 99.0% for emotion Happy, 95% for emotion Disgust and 93.0% emotion Neutral. Both models are with the use of the non-facial landmarking method. Mini Xception would be a better option to use when it comes to non-landmarking model.

Due to a popularity in deep learning model of Transformers, we employed ViT. Our current progress for the ViT model is the training of the network with the dataset. The ViT model training was different to the previous two methods due to the time costs. The ViT model training was conducted with a number of epochs. The result of the training was obtained in evaluation with accuracy and loss rates, while the training was measured using loss. With the training of six epochs, the loss rate significantly decreases from 1.12 to 0.21. While evaluating the model from training, the model was able to achieve accuracy from 63% to 69.4% at 5-th epoch. The accuracy of the sixth epoch decreased by a small percentage of 0.4%. This small margin is not significant enough to cause deviation of accuracy for future real-time experiment.

The last model that we implemented was with the use of facial-landmarking method. With the use of Dlib toolkit for frontal face detection and facial landmark predictor for

our live experiment. The model that we employed to test this technique is SDNN. We trained the model with 300, 400, 500 and 600 epochs, respectively. With the live experiment, we were able to achieve accuracy of 96.0% for emotion Happy and 90.0% for emotion Angry. With the high accuracy, we plan to further improve it with the use of Graph Convolutional Network (GCN) for facial landmarking technique as it provides a better representation of prediction of FER due to its facial features landmarking.

During the models training, we find a trend as the number of epochs increases, the accuracy rate of our algorithm increases, this was shown during the training of each model. With the initial model of CNN, we unfold that the accuracy rate does not increase however the validation accuracy changes slightly while the increase of epochs. On the other hand, both Mini Xception and SDNN algorithms have increased significantly. The Mini Xception has increased from accuracy 68% to 76% over the increase of 600 epochs, while SDNN has increased from accuracy 69% to 77%. Therefore, we believe that SDNN with facial landmarking method will provide the best method as it allows capturing of the important features during classification of human emotions.

With the progress of our proposed models, the test of three proposed models has been conducted: CNN, Mini Xception, and SDNN. In addition, we employed the training of the ViT model. We believe that with the training result of the ViT model it provides the best model for non-landmarking technique. However, with the experimental result conducted on the CNN and Mini Exception model with the non-landmarking technique, Mini Xception has performed more consistent than that of CNN. Therefore, based on the training with non-landmarking technique, we suggest using ViT model is he best for the non-landmarking. However, with the live experiment for the non-landmarking, we suggest using Mini Xception.

For the facial landmarking technique, with the employment of SDNN model. We believe that with the employment of facial landmarking, the performance would surpass the technique of the non-landmarking as it is able to distinguish more of the similar emotions such as Fear and Disgusted. We plan to implement the GCN model with training, testing and experimentation which will be discussed in the Future Work section.

## 5.2 Limitations

For our project, we encountered a few limitations that deter our progression and the outcome of the project, which include dataset, computing time, and knowledge. There are various public datasets that are available for our FER project, however, the public dataset requires access and permission from the manufacturer or the creator. The process of acquiring the permission is to fill in a form that was provided. The key issue with the process is that it takes time for the manufacturer or the creator to grant you access. With the assumptions, it will take a lot of time to gain access to the dataset. Therefore, we had to use the dataset that can be downloaded from Kaggle.

We ran this project with our personal device. Although our device was able to output results, it took longer time for our algorithms to be trained and tested. Lastly, due to lack of knowledge and understanding in how to pre-process and train graph structures, we were not able to get results for training and testing for our proposed Graph Convolutional Networks and the live experimentation of the ViT model.

## 5.3 Future Work

In this thesis, we only explored the proposed CNN, Mini Xception, and Simple Deep Neural Network. CNN is one of the deep learning models for FER. Due to our current progress not being experimented on the ViT and GCN models, we will implement the GCN algorithm in our future work. Due to the nature of GCN with the need to use the number of nodes, we will preprocess our visual data by using landmarks similarly implemented by multiple studies, with the ideas of getting geometrical features to distinguish the difference in similar emotions. This can be accomplished by taking the mean or the center of gravity of the landmarking of our features of the left eye, right eye, and mouth. For the ViT, since we already trained and tested the model, we plan to implement the model into a real-time experiment with a hypothesis of higher and more consistent accuracy compared to the CNN and Mini Xception.

The algorithm that we plan to implement in our future work is the GCN model. The

algorithm will be employed as a baseline for the facial landmarking technique. The model will consist of three building blocks. The first block contains five layers of GCN for filtering the pixels from our dataset, this includes nodes and classes of emotions. The next block is the average pooling layer and the last of the layers will be two fully connected layer that will then output accuracy and emotions for our testing.

In addition, we plan to implement more methods of increasing accuracy. The method includes increasing the number of layers, data augmentation, and image resizing. Due to our SDNN only having four dense layers that consist of the same number of filter size, we can decrease the filtering size as we increase the number of layers. By increasing the number of layers and filter, it then allows more filtration process of the image pixels which can lead to increasing accuracy of the model as it allows to distinguish the difference of emotions.

# References


Alvarez, V. M., Sanchez, C. N., Gutierrez, S., Dominguez-Soberanes, J., & Velazquez, R. (2018). Facial emotion recognition: A comparison of different landmark-based classifiers. *International Conference on Research in Intelligent and Computing in Engineering (RICE)*

An, N., Yan, W. (2021) Multitarget tracking using Siamese neural networks. *ACM Transactions on Multimedia Computing, Communications and Applications.*

Ashwini, U., Kalaivani, K., Ulagapriya, K., & Saritha, A. (2021). Time series analysis based Tamilnadu monsoon rainfall prediction using seasonal ARIMA. *International Conference on Inventive Computation Technologies (ICICT)*

Benitez-Quiroz, C. F., Srinivasan, R., & Martinez, A. M. (2016). EmotioNet: An accurate, real-time algorithm for the automatic annotation of a million facial expressions in the wild. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*

Behera, G.S. (2020) Face detection with Haar cascade. *Towards Data Science*

Behera, B., Prakash, A., Gupta, U., Semwal, V. B., Chauhan, A. (2021) Statistical prediction of facial emotions using Mini Xception CNN and time series analysis. *Transactions on Computer Systems and Networks*. 397–410.

Boesch, G. (2022). Vision Transformers (ViT) in image recognition – 2022 guide. *Viso AI.*

Brownlee, J. (2019) A gentle introduction to bath normalization for deep neural networks. *Machine Learning Mastery*

Brownlee, J. (2019) A gentle introduction to rectified linear unit (ReLU). *Machine Learning Mastery*

Brownlee, J. (2017) A gentle introduction to autocorrelation and partial correlation. *Machine Learning Mastery*

Budhiraja, A. (2016) *Dropout in (deep) machine learning*. *Medium*

Bushaev, V. (2018) Adam – Latest trends in deep learling optimization. *Towards Data Science.*

Chen, H., Deng, Y., Cheng, S., Wang, Y., Jiang, D., & Sahli, H. (2019). Efficient spatial temporal convolutional features for audiovisual continuous affect recognition. *International on Audio/Visual Emotion Challenge and Workshop* (pp. 19-26).

Choi, D. Y., Kim, D. H., & Song, B. C.. (2018). *Recognizing fine facial micro-expressions using two-dimensional landmark feature*. IEEE International Conference on Image Processing (ICIP)

Chouhayebi, H., Riffi, J., Mahraz, M. A., Yahyaouy, A., Tairi, H., & Alioua, N.. (2020). Facial expression recognition based on geometric features. *IEEE International Conference on Image Processing (ICIP)*

Cui, W., Yan, W. (2016) A scheme for face recognition in complex environments. *International Journal of Digital Crime and Forensics* (IJDCF) 8 (1), 26-36.

Doroftei, I., Adascalitei, F., Lefeber, D., Vanderborght, B., & Doroftei, I. A. (2016). Facial expressions recognition with an emotion expressive robotic head. *IOP Conference Series: Materials Science and Engineering* (Vol. 147, No. 1, p. 012086). IOP Publishing.

Duncan, D., Shine, G., & English, C. (2016). Facial emotion recognition in real time. *Semantic scholar, Computer Science.*

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv:2010.11929.*

Dsouza, J. (2021). How to improve the accuracy of your image recognition model. *freeCodeCamp.*

Fan, Y., Lu, X., Li, D., & Liu, Y. (2016). Video-based emotion recognition using CNN-RNN and C3D hybrid networks. *ACM International Conference on Multimodal Interaction* (pp. 445-450).

Gao, X., Nguyen, M., Yan, W. (2021) Face image inpainting based on generative adversarial network. *International Conference on Image and Vision Computing New Zealand.*

Gao, X., Nguyen, M., Yan, W. (2022) A face image inpainting method based on autoencoder and adversarial generative networks. *Pacific-Rim Symposium on Image and Video Technology*.

Gholamy, A., Kreinovich, V., & Kosheleva, O. (2018). Why 70/30 or 80/20 relation between training and testing sets: a pedagogical explanation.

Gupta, S.. (2018). Facial emotion recognition in real-time and static images. *International Conference on Inventive Systems and Control (ICISC)*

He, Z., Zhang, J., Kan, M., Shan, S., & Chen, X.. (2017). Robust FEC-CNN: A high accuracy facial landmark detection system. *IEEE Conference on Computer Vision and Pattern Recognition Workshops* (CVPRW)

Heaven, D. (2020) Why faces don't always tell the truth about feelings. *Nature*, 578, 502-504.

Hedge, S. (2021) An introduction to seperable convolutions. *Analytics Vidhya.*

Huynh, C. M., & Balas, B. (2014). Emotion recognition (sometimes) depends on horizontal orientations. *Attention, Perception, & Psychophysics*, 76(5), 1381-1392.

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: Principles and Practice.* OTexts.

Jain, D. K., Shamsolmoali, P., & Sehdev, P. (2019). Extended deep neural network for facial emotion recognition. *Pattern Recognition Letters*, *120*, 69-74.

Jepsen, T. S. (2018). How to do deep learning on graphs with graph convolutional Networks. *Towards Data Science.*

Lei, L., Chen, T., Li, S., & Li, J.. (2021). Micro-expression recognition based on facial graph representation learning and facial action unit fusion. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (CVPRW)

Liao, L., Zhu, Y., Zheng, B., Jiang, X., & Lin, J.. (2022). FERGCN: Facial expression recognition based on graph convolution network. *Machine Vision and Applications*, *33*(3).

Lini, R. (2021) Facial landmark detection. *Medium.*

Liu, M., Yan, W. (2022) Masked face recognition in real-time using MobileNetV2. *ACM ICCCV.*

Ma, F., Sun, B., & Li, S.. (2021). Facial Expression recognition with Visual Transformers and attentional selective fusion. *IEEE Transactions on Affective Computing*, 1–1.

Maeda, K., Kurebayashi, I., Komuro, N., & Hirai, K.. (2021). A study on time series analysis of environmental data for predicting emotional conditions. *IEEE Global Conference on Life Sciences and Technologies* (LifeTech)

Mayachita, I. (2020). Understanding graph convolutional networks for node classification. *Towards Data Science.*

Ma, Y., & Tang, J. (2021). *Deep Learning on Graphs*. Cambridge University Press.

Mehendale, N. (2020) Facial emotion recognition using convolutional neural networks (FERC). *SN Appl. Sci.* 2, 446.

Mellouk, W., Handouzi, W. (2020) Facial emotion recognition using deep learning: review and insights. *Procedia Computer Science,* 175, 689-694.

Ngoc, Q. T., Lee, S., Song, B. C. (2020) Facial landmark-based emotion recognition via directed graph neural network. *Electronics,* 9(5), 764.

Nguyen, M., Yan, W. Temporal colour-coded facial-expression recognition using convolutional neural network. *International Summit Smart City 360°: Science and Technologies for Smart Cities.*

Pan, C., Yan, W. (2018) A learning-based positive feedback in salient object detection. *International Conference on Image and Vision Computing New Zealand.*

Pan, C., Yan, W. (2020) Object detection based on saturation of visual perception. *Multimedia Tools and Applications, 79 (27-28), 19925-19944.*

Pan, C., Liu, J., Yan, W., Zhou, Y. (2021) Salient object detection based on visual perceptual saturation and two-stream hybrid networks. *IEEE Transactions on Image Processing.*

Papadopoulos, K., Kacem, A., Shabayek, A., Aouada, D. (2021) Face-GCN: A graph convolutional network for 3D dynamic face identification / recognition. *arxiv.org/pdf/2104.09145*

Pawar, Dipti. (2018). *Improving performance of convolutiona neural network*. Medium.

Plummer, A. (2020) Different types of time series decomposition. *Towards Data Science.*

Pierre, S. (2021) A guide to time series analysis in Python. *Builtin*.

Prabhakaran, S. (2019) Time series analysis in Python – A comprehensive guide with examples. *Machine Learning Plus.*

Prabhakaran, S. (2021) ARIMA Model– Complet guide to time series forecasting in Python. *Machine Learning Plus.*

Poulose, A., Kim, J. H., & Han, D. S.. (2021). Feature vector extraction technique for facial emotion recognition using facial landmarks. *International Conference on Information and Communication Technology Convergence* (ICTC)

Radecic, D. (2021) Time series from scartch – Decomposing time series data. *Towards Data Science.*

Radhakrishna, A., Yan, W., Kankanhalli, M. (2006) Modeling intent for home video repurposing. *IEEE MultiMedia 13 (1), 46-55.*

Rosebrock, A. (2019) Keras image data generator and data augmentation. *Pyimagesearch.*

Sarin, S. (2019) Exploring data augmentation with Keras and TensorFlow. *Towards Data Science.*

Senaratne, R. (2020) CLAHE and Thresholding in Python. *Towards Data Science.*

Sharma, S. (2017) Epoch vs batch size vs iterations. *Towards Data Science.*

Shen, D., Xin, C., Nguyen, M., Yan, W. (2018) Flame detection using deep learning. *International Conference on Control, Automation and Robotics.*

Singh, A. (2018) A gentle introduction to handling non-stationary time series in Python. *Machine Learning Plus.*

Smeda, K. (2019) Understanding the architecture of CNN. *Towards Data Science.*

Somepali, G. (2021). Vision Transformer – Paper summary. *Medium.*

Song, C., He, L., Yan, W., Nand, P. (2019) An improved selective facial extraction model for age estimation. *International Conference on Image and Vision Computing New Zealand.*

Taghi Zadeh, M. M., Imani, M., Majidi, B. (2019) Fast facial emotion recognition using convolutional neural networks and gabor filters. *KBEI,* 577-581.

Tautkute, I., Trzcinski, T., & Bielski, A.. (2018). I know how you feel: Emotion recognition with facial landmarks. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (CVPRW)

Versloot, C. (2021) What are max pooling, average pooling, global max pooling and global average poolling? *MachineCurve*.

Wang, H., Yan, W. (2022) Face detection and recognition from distance based on deep learning. *Aiding Forensic Investigation Through Deep Learning and Machine Learning Framework*. IGI Global.

Wang, J., Yan, W., Kankanhalli, M., Jain, R., Reinders, M. (2003) Adaptive monitoring for video surveillance. *International Conference on Information, Communications and Signal Processing.*

Xu, X., Ruan, Z., & Yang, L.. (2020). Facial expression recognition based on graph neural network. *IEEE International Conference on Image, Vision and Computing* (ICIVC)

Yan, W., Kankanhalli, M. (2015) Face search in encrypted domain. *Pacific-Rim Symposium on Image and Video Technology, 775-790.*

*Yan, W. (2019) Introduction to Intelligent Surveillance: Surveillance Data Capture, Transmission, and Analytics. Springer London.*

*Yan, W. (2021) Computational Methods for Deep Learning: Theoretic, Practice and Applications. Springer London.*

Zadeh, M. M. T., Imani, M., & Majidi, B. (2019, February). Fast facial emotion recognition using convolutional neural networks and Gabor filters. *IEEE Conference on Knowledge Based Engineering and Innovation* (KBEI) (pp. 577-581).

Zita, C. (2022). 5 Effective ways to improve the accuracy of your machine learning model. *Towards Data Science.*

Zhang, T., Zheng, W., Cui, Z., Zong, Y., Yan, J., & Yan, K.. (2016). A deep neural network-driven feature learning method for multi-view facial expression recognition. *IEEE Transactions on Multimedia*, *18*(12), 2528–2536.

Zhang, J., Sun, G., Zheng, K., Mazhar, S., Fu, X., & Yang, D.. (2021). Emotion recognition based on graph neural networks. *Communications in Computer and Information Science* (pp. 472–480).

Zheng, K., Yan, Q., Nand, P. (2017) Video dynamics detection using deep neural networks. *IEEE Transactions on Emerging Topics in Computational Intelligence.*