

DEEP LEARNING FOR IMAGE CLASSIFICATION

BY

KOI CHIN CHONG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

JUNE 2024

REPORT STATUS DECLARATION FORM

Title: DEEP LEARNING FOR IMAGE CLASSIFICATION

Academic Session: JUNE 2024

I KOI CHIN CHONG

(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in

Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,



(Author's signature)



(Supervisor's signature)

Address:

48, JALAN BERCHAM SURIA 1,

TAMAN BERCHAM SURIA

31400 IPOH, PERAK.

Date: 06/09/2024

DR ZANARIAH BINTI ZAINUDIN

Supervisor's name

Date: 6/9/24

Universiti Tunku Abdul Rahman			
Form Title : Sample of Submission Sheet for FYP/Dissertation/Thesis			
Form Number: FM-IAD-004	Rev No.: 0	Effective Date: 21 JUNE 2011	Page No.: 1 of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 06/09/2024

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that **Koi Chin Chong** (ID No: **20ACB05613**) has completed this final year project/ dissertation/ thesis* entitled “ *Deep Learning For Image Classification* ” under the supervision of Dr Zanariah binti Zainudin (Supervisor) from the Department of Digital Economy Technology, Faculty/Institute* of Information and Communication Technology .

I understand that University will upload softcopy of my final year project / ~~dissertation/ thesis*~~ in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.


Yours truly,



(*Koi Chin Chong*)

DECLARATION OF ORIGINALITY

I declare that this report entitled “**DEEP LEARNING FOR IMAGE CLASSIFICATION**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature :  _____

Name : Koi Chin Chong

Date : 06/09/2024

ACKNOWLEDGEMENTS

I would like to express thanks and appreciation to my supervisor, Dr Zanariah Binti Zainudin and my moderator, Dr Noraini Binti Ibrahim who have given me a golden opportunity to involve in the deep learning for image classification project. Besides that, they have given me a lot of guidance in order to complete this project. When I was facing problems in this project, the advice from them always assists me in overcoming the problems. Again, a million thanks to my supervisor and moderator.

Finally, I must say thanks again to my supervisor, Dr Zanariah Binti Zainudin for their guidance, support, and continuous encouragement throughout the course.

Abstract

Facial emotion detection plays a vital role in human communication, as facial expressions are key indicators of emotional states and intentions. This research investigates the application of Convolutional Neural Networks (CNNs), specifically VGG16 and ResNet50 models, for facial expression image classification using the AffectNet dataset. The primary objective is to enhance the accuracy of facial emotion recognition by employing various training strategies, including full training, fine-tuning with frozen and unfrozen layers, and leveraging pre-trained models. The Multi-task Cascaded Convolutional Networks (MTCNN) technique was integrated to improve face detection capabilities within these models. The results demonstrate that the VGG16 model achieved the highest accuracy. When fully trained from scratch on the AffectNet dataset, it attained an accuracy of 69.95%. This performance was further improved to 69.98% through fine-tuning with initially frozen layers, highlighting the effectiveness of leveraging pre-trained features while refining deeper layers for emotion recognition tasks. The ResNet50 model also showed significant improvement with fine-tuning, achieving an accuracy of 71.72% when layers were initially frozen and then fine-tuned. The CNN model performed moderately, with the best accuracy of 58.71% observed when layers were frozen during fine-tuning after full training. The integration of MTCNN with ResNet50 was particularly effective, allowing it to predict different sets of data with the most accuracy among the tested models. All findings contribute to the ongoing research in facial expression recognition and provide insights into the effectiveness of various CNN architectures and training strategies for improving emotion classification accuracy.

TABLE OF CONTENTS

TITLE PAGE	i
REPORT STATUS DECLARATION FORM	ii
FYP THESIS SUBMISSION FORM	iii
DECLARATION OF ORIGINALITY	iv
ACKNOWLEDGEMENTS	v
ABSTARCT	vi
TABLE OF CONTENTS	x
LIST OF FIGURES	xi
LIST OF TABLES	xiii
LIST OF ABBREVIATION	xiv
CHAPTER 1 INTRODUCTION	1
1.1 Problem Statement and Motivation	1
1.2 Objectives	2
1.3 Project Scope and Direction	3
1.4 Contributions	4
1.5 Report Organization	4
CHAPTER 2 LITERATURE REVIEW	6
2.1 Previous Researchers' Work on Different Models	6
2.2 Limitations of Previous Researchers	7
2.3 Summary	8
CHAPTER 3 RESEARCH METHODOLOGY/MODELLING	9
3.1 Research Methodology	9
3.2 Modelling and Hyperparameter Tuning	10
3.3 Dataset Requirement – AffectNet	12
3.4 Modelling	12
3.4.1 Convolutional Neural Networks (CNNs)	12

3.4.2 VGG16	13
3.4.3 ResNet50	14
3.5 Evaluation Methods	14
3.5.1 Accuracy	14
3.5.2 Precision	15
3.5.3 Recall	15
3.5.4 F1 Score	15
3.5.5 Multi-task Cascaded Convolutional Neural Network (MTCNN)	15
3.6 Software Requirement – Jupyter Notebook and Python	16
3.7 Summary	17

CHAPTER 4 DATA PRE-PROCESSING, MODELLING AND HYPERPARAMETER TUNING **18**

4.1 Data pre-processing	18
4.2 Modelling and Hyperparameter Tuning – VGG16, ResNet50, and CNN Model	22
4.3 Setting up Model Checkpoint and Early Stopping Callbacks	24
4.4 Training the Model	25
4.5 Hyperparameter Tuning	26
4.6 Train Model	28
4.7 Summary	28

CHAPTER 5 EXPERIMENTAL RESULT AND DISCUSSION **30**

5.1 Experimental Result – VGG16 (Fully connected layers)	30
5.2 MTCNN Experimental Results – VGG16	35
5.3 Experimental Result – ResNet50 (Fully connected layers)	38
5.4 MTCNN Experimental Results – ResNet50	42
5.5 Experimental Result – CNN Model (Fully connected layers)	47
5.6 MTCNN Experimental Results – CNN Model	51
5.7 Comparison Performance with Previous Researchers	55
5.8 Project Challenges	57
5.9 Summary	57

CHAPTER 6 CONCLUSION	59
6.1 Conclusion	59
6.2 Future Work	60
REFERENCES	61
WEEKLY LOG	63
POSTER	70
PLAGIARISM CHECK RESULT	78
FYP2 CHECKLIST	79

LIST OF FIGUERS

Figure Number	Title	Page
Figure 3.1.1	Research Methodology Flowchart	9
Figure 3.2.1	Hyperparameter Tuning Model Flowchart	11
Figure 3.4.1.1	Convolutional Neural Networks Architecture	13
Figure 3.4.2.1	VGG16 architecture	13
Figure 3.4.3.1	ResNet50 Architecture	14
Figure 3.5.6.1	Network structure of MTCNN	16
Figure 4.1.1	Define Directories	18
Figure 4.1.2	Define Class Names	18
Figure 4.1.3	Data Split	19
Figure 4.1.4	Data Split folders	19
Figure 4.1.5	Angry Data file	20
Figure 4.1.6	Data Augmentation	20
Figure 4.1.7	Data Generators	21
Figure 4.2.1	VGG16 Model Creation	22
Figure 4.2.2	ResNet50 Model Creation	23
Figure 4.2.3	CNN Model Creation	24
Figure 4.3.1	Model Checkpoint and Early Stopping Callbacks	24
Figure 4.4.1	Train Model	25
Figure 4.5.1	Hyperparameter Tuning Data Augmentation	26
Figure 4.5.2	VGG16 Layers Freeze	26
Figure 4.5.3	ResNet50 Layers Freeze	27
Figure 4.5.4	CNN Layers Freeze	27
Figure 4.6.1	Train Model	28
Figure 5.1.1	VGG16 Loss and Accuracy Line Graphs	30
Figure 5.1.2	VGG16 Classification Report	30
Figure 5.1.3	VGG16 Confusion Matrix	31
Figure 5.1.4	VGG16 Predicted Images Results	32
Figure 5.1.5	VGG16 Tuning Classification Report	32

Figure 5.1.6	VGG16 Tuning Confusion Matrix	33
Figure 5.1.7	VGG16 Tuning Predicted Images Results	34
Figure 5.2.1	MTCNN – VGG16 Happy Datasets Predicted Results	35
Figure 5.2.2	MTCNN – VGG16 Sad and Angry Images Predicted Results	36
Figure 5.2.3	MTCNN – VGG16 Group Image Predicted Results	36
Figure 5.2.4	MTCNN – VGG16 Group Image Predicted Results	37
Figure 5.2.5	MTCNN – VGG16 Group Image Predicted Results	37
Figure 5.3.1	ResNet50 Loss and Accuracy Line Graphs	38
Figure 5.3.2	ResNet50 Classification Report	38
Figure 5.3.3	ResNet50 Confusion Matrix	39
Figure 5.3.4	ResNet50 Predicted Images Results	40
Figure 5.3.5	ResNet50 Tuning Classification Report	40
Figure 5.3.6	ResNet50 Tuning Confusion Matrix	41
Figure 5.3.7	ResNet50 Tuning Predicted Images Results	41
Figure 5.4.1	MTCNN – ResNet50 Happy Datasets Predicted Results	43
Figure 5.4.2	MTCNN – ResNet50 Sad and Angry Images Predicted Results	44
Figure 5.4.3	MTCNN – ResNet50 Group Image Predicted Results	44
Figure 5.4.4	MTCNN – ResNet50 Group Image Predicted Results	45
Figure 5.4.5	MTCNN – ResNet50 Group Image Predicted Results	46
Figure 5.5.1	CNN Loss and Accuracy Line Graphs	47
Figure 5.5.2	CNN Classification Report	47
Figure 5.5.3	CNN Confusion Matrix	48
Figure 5.5.4	CNN Predicted Images Results	48
Figure 5.5.5	CNN Tuning Classification Report	49
Figure 5.5.6	CNN Tuning Confusion Matrix	50
Figure 5.5.7	CNN Tuning Predicted Images Results	51
Figure 5.6.1	MTCNN – CNN Happy Datasets Predicted Results	52
Figure 5.6.2	MTCNN – CNN Sad and Angry Images Predicted Results	52
Figure 5.6.3	MTCNN – CNN Group Image Predicted Results	53
Figure 5.6.4	MTCNN – CNN Group Image Predicted Results	53

LIST OF TABLES

Table Number	Title	Page
Table 1.1	Comparison Performance with Previous Researchers	55

LIST OF ABBREVIATIONS

<i>CNNs</i>	Convolutional Neural Networks
<i>AI</i>	Artificial Intelligent
<i>MTCNN</i>	Multi-task Cascaded Convolutional Networks
<i>FER</i>	Facial Emotion Recognition
<i>PSR</i>	Pyramid with Super-Resolution Model
<i>ViT</i>	Vision Transformer model
<i>SENet</i>	Squeeze-and-Excite Network
<i>RAF-DB</i>	Real-world Affective Faces Database
<i>TP</i>	True Positive
<i>TN</i>	True Negative
<i>FP</i>	False Positive
<i>FN</i>	False Negative

Chapter 1

Introduction

Facial expression image classification involves deep learning tasks where models are trained to recognize human emotions like happiness, anger, sadness, and others in visual data, specifically in photographs of faces. To accomplish this, convolutional neural networks (CNNs) and similar advanced methods in deep learning are often employed. Models such as VGG16 and ResNet50 are commonly used, forming the core of the method[1]. These architectures help in learning the subtle patterns and variations of facial expressions, though with some limitations. Transfer learning plays a role where pre-trained models like VGG16 and ResNet50, initially designed for broader picture classification tasks, are adjusted slightly using facial expression datasets, allowing for quicker training and better generalization on less data.

Several techniques are employed to evaluate how accurate and adaptable these models are. Usually, the dataset is divided into parts for training and testing so the model can learn from one set and be evaluated on another. Methods like k-fold cross-validation may provide an improved evaluation of the model's performance, especially when the available data is limited. Additionally, metrics like accuracy, recall, and F1-score, along with tools like confusion matrices and ROC curves, are used to measure how well the model identifies each emotion. Techniques like data augmentation and regularization are also used, helping the model become more accurate by ensuring it can handle different conditions like lighting changes, angles, and obstructions, thus making it more suitable for practical use[2].

1.1 Problem Statement and Motivation

The problem statement of this research is that current facial expression classification models such as CNNs, VGG16, and ResNet50 are often inaccurate since they were trained on unreliable facial emotion datasets of limited size or low variety. This limitation may have a negative impact on the usability of these models in real-world applications, particularly when facial expressions vary significantly between participants and circumstances. Adding more data or various types of facial expressions with this objective it will result in stronger models capable of accurately identifying different moods. It also aids in improving user experience, particularly with interactive software, surveillance, and behaviour analysis.

The research is needed to improve performance in facial expression classification, which leads to unpredictable results since it had insufficient training with certain layer configurations such as CNN VGG16 or ResNet50. These models may be more (or less) effective depending on whether layers are frozen or completely trainable during fine-tuning. This research is motivated by the need for a detailed comparative comparison of these two training approaches in order to determine which is more effective for boosting performance during facial emotion classification. This is important in sectors where accurate emotion detection is critical, such as targeted advertising, mental state identification, and enhanced user experience via human-like dialog with an artificial intelligence.

The purpose of this research is to investigate the issue of how CNN, VGG16, and ResNet50 models identify facial emotions when combined with the Multi-task Cascaded Convolutional Networks (MTCNN) method. When applied in a variety of applications, the models show lower accuracies for different face image dataset collections, which might lead to inaccurate findings when recognizing emotions. The motive for this particular effort is to manually test and compare those models to the MTCNN framework structure, which is utilized to successfully detect facial emotions in a wider range of images.

1.2 Objectives

1. To implement CNN, VGG16, and ResNet50 models with an expanded dataset and increased class diversity to enhance facial expression classification accuracy.
2. To analyse and compare the accuracy of CNN, VGG16, and ResNet50 models in accurately classifying a broad range of facial expressions, using both frozen layers with fine-tuning and fully trained layers with fine-tuning.
3. To evaluate and compare of CNN, VGG16, and ResNet50 models by using MTCNN(Multi-task Cascaded Convolutional Networks) algorithm to predict of these models against a different collection of face photos to discover which model detects facial emotions most accurately

The primary goal includes using convolutional neural networks (CNNs), VGG16, and ResNet50 models, trained on a larger and more varied dataset that contains a wider spectrum of facial emotions. Through this expansion in the number and diversity of classes, the models are adjusted to capture more detailed variations in facial features and surrounding conditions.

This is aimed at improving the accuracy, but also enhancing the robustness of the models when they classify emotions.

The second goal is about assessing and making a comparison of the precision of three kinds of deep learning models CNN, VGG16, and ResNet50 in the task of identifying different types of facial emotions. This comparison would be done across different conditions, particularly by freezing and unfreezing layers of these models. Freezing of layers involves keeping certain layer weights unchangeable during the entire training, while unfreezing permits those weights to change. This investigation aims at understanding how the changes in model architecture and training procedures influence the ability to accurately classify facial emotions, thereby offering some understanding into which emotion identification algorithms are more effective.

The final goal is to look into and compare how the CNN, VGG16, and ResNet50 models work when they are paired with the MTCNN (Multi-task Cascaded Convolutional Networks) method. The MTCNN is generally employed for facial recognition, and this objective focuses on exploring how well the three models can predict and categorize facial emotions after MTCNN has detected them. The evaluation will rely on how these models perform with another set of facial images, aiming to figure out which model shows better accuracy in recognizing and categorizing facial emotions under varied conditions.

1.3 Project Scope and Direction

The project scope of this project about categorizing facial expressions has a project scope which includes several key parts, of which some are more significant. The research is first focused on the task of gathering a dataset that is diverse and broad, involving a variety of facial emotions collected from different demographic groups, under various lighting and with potential occlusions; this is why the AffectNet dataset will be chosen. This dataset will go to train not one, but three deep learning models: CNN, VGG16, and ResNet50 each one tailored differently to handle the complexities that come with categorizing facial emotions. A thorough examination of the model's performances will be conducted by performing statistical analysis, comparing accuracy based on vague but important criteria such as precision, recall, and F1 score. Another set of facial images will be brought in to test how well each model can predict, to ensure that when new data is seen, the correct categorization can be achieved.

The project's primary goal will be to improve the accuracy of CNN, VGG16, and ResNet50 models so that they can better predict emotions. This will involve making repeated adjustments and improving model parameters after assessing performances against the broader dataset. The goal here is to make these models consistently able to identify a wide range of facial expressions and apply these abilities in real situations where subtle emotion detection might be necessary. This could potentially lead to significant advances in areas like interactive AI systems, personalized digital experiences, and clinical diagnostics, where knowing human emotions plays an important role.

1.4 Contributions

This project makes a substantial contribution to face expression picture classification by giving a thorough comparison of model performances (CNN, VGG16, ResNet50) versus current benchmarks on the AffectNet dataset. By explicitly comparing these findings with previous research, the project highlights the gains or boosts in accuracy and resilience produced by larger training datasets and better model architectures. This comparison project aids in finding the most successful ways and establishes a new level of accuracy in the area.

Furthermore, by assessing the prediction accuracy of CNN, VGG16, and ResNet50 models using the AffectNet dataset, this project provides important insights into the models' practical application. It evaluates how successfully these models can be incorporated into real-world scenarios requiring precise emotion identification, such as security systems, customer service, and interactive learning environments. This helps us understand the deployment obstacles and potential for emotional AI solutions.

1.5 Report Organization

The research is divided into six chapters: Introduction, Literature Review, Research Methodology, Data Pre-processing, Modelling and Hyperparameter Tuning, Experimental Results and Discussion, and Conclusion. The first chapter introduces the project, including the problem description, project history and motivation, project scope, project objectives, project contribution, project highlights, and report structure. The second chapter is a literature assessment of the several current different deep learning models employed in this research,

with a focus on data pre-processing, feature selection methods, and hyperparameter tuning methods. The third chapter examines the general flow and needs of the project. The fourth chapter goes into depth on how to implement pre-processing on the specified dataset and into the specifics of applying the modelling stages and hyperparameter improving on the models selected. The fifth chapter examined the outcomes from the fourth chapters. The sixth chapter ended the investigation and findings with a summary and the future work.

Chapter 2

Literature Review

2.1 Previous Researchers' Work on Different Models

Previous researchers have extensively utilized deep learning models, particularly Convolutional Neural Networks (CNNs) such as VGG16 and ResNet50, to evaluate the effectiveness of facial emotion recognition (FER) methods. These models are popular for their proven ability to extract and learn complex features from images. The studies have leveraged datasets like AffectNet, FER2013, and RAF-DB, which offer a diverse range of facial expressions, thereby enhancing model training and testing. CNN architectures like VGG16 and ResNet50 are robust due to their deep layers, enabling them to detect subtle variations in facial expressions, making them highly suitable for emotion recognition tasks. Consequently, these models have formed the foundation for numerous studies, significantly advancing the field of facial expression analysis.

In this previous study, the focus was on identifying the optimal convolutional neural network (CNN) model for binary classification of normal contrast medical images enhanced by the CLAHE technique. The researchers compared the architectures of VGG16, VGG19, and ResNet50 based on their accuracy, F1 score, and recall using a selected set of brain images. Over ten experiments conducted in two modes—data augmentation and no data augmentation—VGG16 emerged as the superior architecture for medical image classification. In the data augmentation mode, the accuracies for normal contrast images were 0.78 for VGG16, 0.66 for VGG19, and 0.69 for ResNet50; for CLAHE-enhanced images, they were 0.78, 0.76, and 0.72 respectively. In the no data augmentation mode, the accuracies improved to 0.88 for VGG16, 0.79 for VGG19, and 0.75 for ResNet50 with normal contrast images, while for CLAHE-enhanced images, they were 0.86, 0.87, and 0.65 respectively[1].

Another study used a deep neural network (DNN) was employed for facial emotion recognition (FER), specifically utilizing a convolutional neural network (CNN) that combines a squeeze-and-excitation network with a residual neural network. The research aimed to identify critical facial features relevant to FER, using the AffectNet and Real-World Affective Faces Database (RAF-DB) as training datasets. Analysis of the feature maps extracted from the residual blocks revealed that the areas around the nose and mouth are significant for the model's performance.

Cross-database validation showed that the model trained on AffectNet achieved an accuracy of 77.37% when tested on RAF-DB, while transfer learning from AffectNet to RAF-DB improved validation accuracy to 83.37%. The findings contribute to a deeper understanding of neural networks and aim to enhance accuracy in computer vision applications[2].

Vo et. al addresses the challenging task of automatic facial expression recognition (FER) from single in-the-wild (ITW) images, which often face issues related to pose, direction, and input resolution. To tackle this, the authors propose a pyramid with super-resolution (PSR) network architecture and introduce a prior distribution label smoothing (PDLS) loss function that incorporates additional prior knowledge about the confusion among different expressions. Experiments conducted on three prominent ITW FER datasets demonstrate that the proposed approach outperforms existing state-of-the-art methods, highlighting its effectiveness in enhancing FER accuracy under real-world conditions[3].

Facial emotion recognition (FER) has gained significant attention in affective computing, yet it continues to face challenges such as sample data quality, effective feature extraction, model creation, and multi-feature fusion. This study explores five classic models—ResNet-50, Xception, EfficientNet-B0, Inception, and DenseNet121—through a series of experiments focusing on data preprocessing, training types, and multi-stage pretraining. The results indicated that using class weights was the most effective method for balancing data. Additionally, the freeze + fine-tuning training approach yielded higher accuracy across datasets, regardless of size. The proposed transfer learning method demonstrated superior accuracy compared to previous studies, achieving increases of 8.37%, 10.45%, 10.45%, 8.55%, and 5.47% for the respective models on the AffectNet dataset, and improvements of 5.72%, 2%, 10.45%, 5%, and 9% on the FER2013 dataset[4].

2.2 Limitations of Previous Researchers

Despite the significant progress made in these studies, several limitations persist. The first study comparing CNN architectures for medical image classification showed that while VGG16 outperformed other models like VGG19 and ResNet50, the accuracy improvements in no data augmentation mode were marginal, particularly for CLAHE-enhanced images, suggesting that the method might not generalize well across different enhancement techniques.

Furthermore, the experiments were conducted on a limited set of brain images, which could

affect the broader applicability of the findings to other types of medical images. In the domain of facial emotion recognition (FER), although the integration of squeeze-and-excitation and residual networks improved accuracy, challenges like data quality and feature extraction remain, as demonstrated by the reliance on transfer learning and the sensitivity of model performance to specific facial regions. Vo et al.'s proposed PSR network for FER addresses real-world conditions, but the method's dependency on prior knowledge, such as confusion patterns among expressions, raises concerns about its scalability to diverse datasets without such priors. Finally, the exploration of five classic models in another FER study revealed that balancing data through class weights and fine-tuning improved accuracy; however, these models may still struggle with variations in input quality and dataset size, indicating the need for more robust preprocessing techniques and model architectures. Overall, the reliance on limited datasets and transfer learning, as well as challenges with generalization and scalability, continue to hinder the broader application of these methods[1]–[4].

2.3 Summary

According to the literature, deep learning models, particularly CNN-based architectures like VGG16, ResNet50, and others, are commonly employed in facial emotion recognition (FER) tasks. These models, trained on datasets such as AffectNet, FER2013, and RAF-DB, are highly effective at extracting complex facial expression features, contributing significantly to advancements in the field. Numerous studies in deep learning have led to notable progress in FER. For example, [1] demonstrated how fine-tuning various deep learning models improved accuracy on the AffectNet and FER2013 datasets. The authors also proposed a fusion strategy that combines ResNet-50, VGG-19, and Inception-V3 models through ensemble learning, achieving impressive accuracy on both the RAF-DB and AffectNet datasets. Together, these studies emphasize the ongoing innovations and challenges in enhancing FER systems through advanced model architectures and fine-tuning techniques. However, the review also highlights some limitations in this research. While fine-tuning techniques yield slight improvements, models that perform well on smaller datasets often struggle with larger, more complex datasets like AffectNet. Factors such as intraclass variation, occlusions, and changes in posture further challenge model performance. Additionally, although sophisticated architectures can achieve high accuracy, they may not be practical for real-time or resource-constrained applications. The ongoing difficulty of scaling models for real-world scenarios underscores the persistent challenges in FER research

Chapter 3

Research Methodology/Approach

3.1 Research Methodology

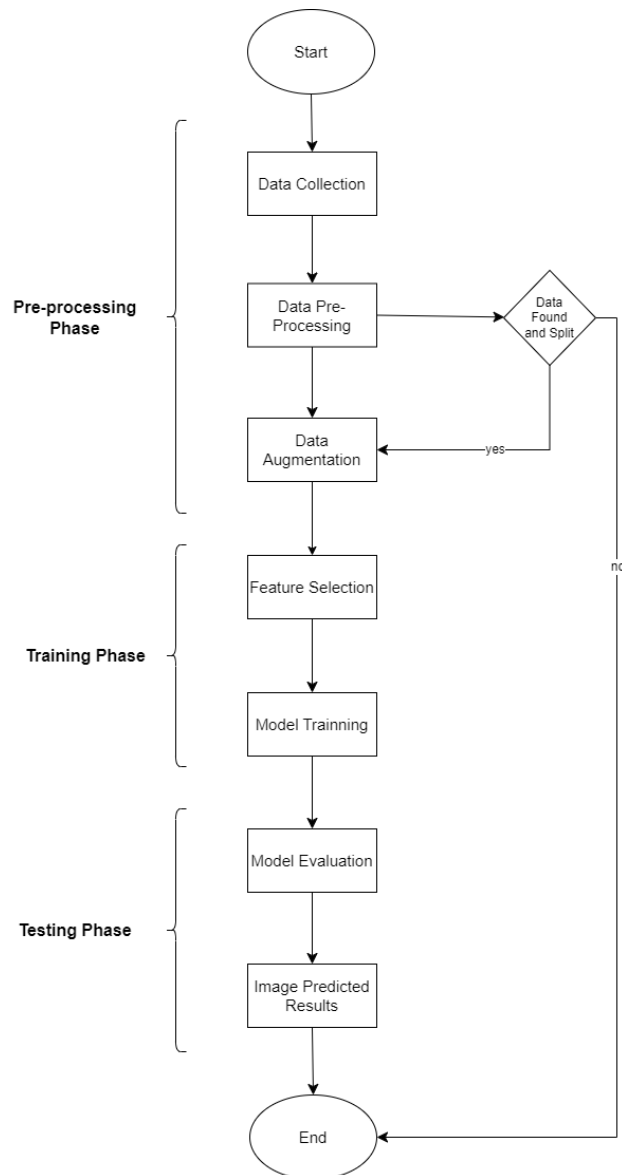


Figure 3.1.1 Research Methodology Flowchart

The research methodology will begin with the collection of data from the AffectNet dataset. This dataset will be imported into Jupyter Notebook for further processing, utilizing Python libraries designed for data importation and preprocessing. During the preprocessing phase, if suitable data is found, it will be divided into training and testing sets. If not, the process will

halt before moving on to the next step: data augmentation. Data augmentation will artificially enhance the dataset's diversity and size through techniques such as rotation, scaling, and flipping, which help improve model robustness and reduce overfitting. Simultaneously, a feature selection process will be conducted to identify and retain only the essential features necessary for optimal model performance.

The choice of architecture, such as CNN, VGG16, or ResNet50, will depend on the specific project being developed. The model training phase will commence at this stage, where the network learns from the training data by adjusting its weights and biases to minimize prediction errors. Once training is complete, the model will be thoroughly evaluated on the testing dataset. Performance metrics such as accuracy, precision, recall, and confusion matrix will be used to assess the model's efficiency and provide insights into its overall performance.

In addition, during the preprocessing and feature selection stages using the AffectNet dataset, performance metrics will help optimize the dataset for better training outcomes. The training, testing, and evaluation processes will be repeated in several iterations to optimize the models, with adjustments made through hyperparameter tuning. This will allow for a comparison of the model's performance with previous research results. Finally, the trained model will be applied to new datasets to predict outcomes, such as identifying emotional expressions in images and classifying them based on the AffectNet dataset.

3.2 Modelling and Hyperparameter Tuning

The initial stage of the process is Data Augmentation, where the project's hyperparameters are first set. This technique enhances the dataset by applying transformations to the images, such as rotation, flipping, and brightness adjustments. By generating variations of the existing data, the model gains more diverse training examples, helping it to better distinguish patterns and generalize across different scenarios, ultimately reducing the risk of overfitting.

The next phase involves determining which layers to freeze or unfreeze, referred to as the Freeze or Unfreeze Model Layers step. This stage defines which layers of the model remain unchanged and which ones are allowed to adapt during training. Once this decision is made, the model training phase can begin, where a suitable architecture, such as CNN, VGG16, or ResNet50, is selected based on the specific requirements and available resources. During

training, the model learns by adjusting the weights of both the frozen and unfrozen layers to minimize prediction errors.

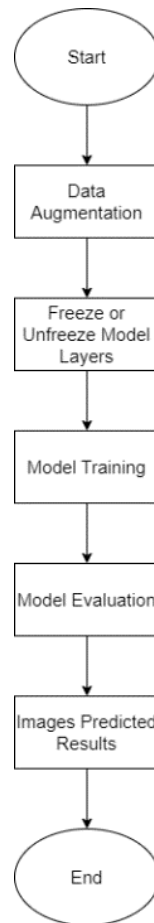


Figure 3.2.1 Hyperparameter Tuning Model Flowchart

After the training phase, the model is tested using a separate dataset to evaluate how well it applies the learned weights in both frozen and unfrozen layers. At this stage, the model's performance is assessed through various metrics, including accuracy, precision, recall, and F1 score, to ensure its effectiveness.

Finally, during the image prediction phase, the trained model is used to predict facial expressions in new images. It assigns a classification label to each image, indicating the detected emotion, along with confidence scores. This marks the final step of the process, where the model's predictions can be applied to various real-world applications, such as emotion recognition systems aimed at improving human-computer interactions.

3.3 Dataset Requirement – AffectNet

In this research, the AffectNet dataset was utilized to train and test the deep learning model. Introduced by Mollahosseini et al. in their, AffectNet is a large-scale facial expression dataset containing approximately 0.4 million images, each manually labeled with one of eight facial emotions (neutral, happy, angry, sad, fear, surprise, disgust, contempt) along with valence and arousal levels. These labels define facial expressions based on their impact on the opposite face. In this study, the AffectNet dataset will undergo preprocessing and feature selection to prepare it for model training and testing. [5].

3.4 Modelling

This research will explore three models that will be implemented in the research. These models are Convolutional Neural Networks (CNNs), VGG16, and ResNet50 as a transfer learning and will be implemented in .

3.4.1 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs), or ConvNets, represent a significant advancement in deep learning architectures, specifically designed to analyze visual data. By leveraging a hierarchical structure of layers, CNNs excel at identifying patterns and features within images, making them ideal for tasks such as object detection, classification, and segmentation. Their architecture allows them to automatically learn relevant features from raw data, reducing the need for manual feature extraction. This capability extends beyond images; CNNs are also applicable in domains like audio processing and time series analysis, showcasing their versatility in handling various types of data[6].

The core of a CNN lies in its convolutional layers, where filters are applied to the input data to create feature maps that highlight the presence of specific features. Initially, these filters capture basic characteristics such as edges and textures. As the data progresses through deeper layers, the network learns to recognize increasingly complex patterns and objects. This multi-layered approach enables CNNs to build a comprehensive understanding of the input data, enhancing their performance in tasks that require high accuracy and precision. The ability to learn filters during training rather than relying on predefined ones is a hallmark of CNNs, contributing to their effectiveness in diverse applications [7].

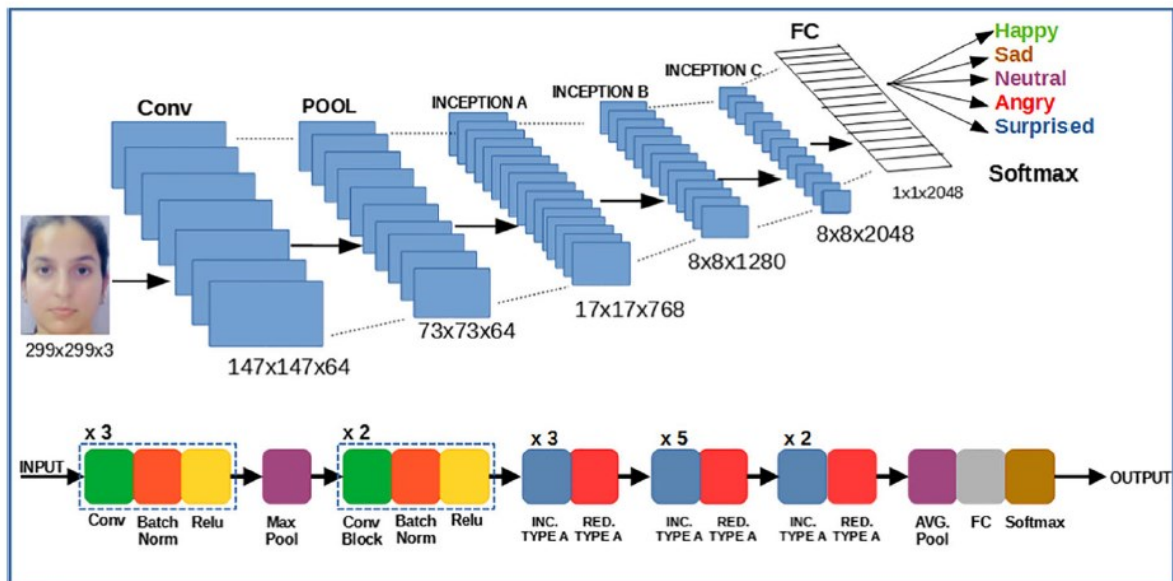


Figure 3.4.1.1 Convolutional Neural Networks Architecture[6]

3.4.2 VGG16

In 2014, Karen Simonyan and Andrew Zisserman presented VGG-16 which included a total of 16 Convolutions. The input image of the network is $(224 \times 224 \times 3)$, with an architecture using fixed size filter stylized as filter (3×3) , having a small window directly attached to the output of many set layers and five stages containing Max pooling dimensioned at (2×2) along different segments within this construction. The only part where they come together at the top is a common softmax output layer of both levels. The VGG16 model is a massive network, more than 138 million parameters. It integrates many convolutional layers to construct deep neural networks, which improves the ability to learn hidden information[7]. Figure 3.4.2.1 shows the architecture of the VGG-16 network.

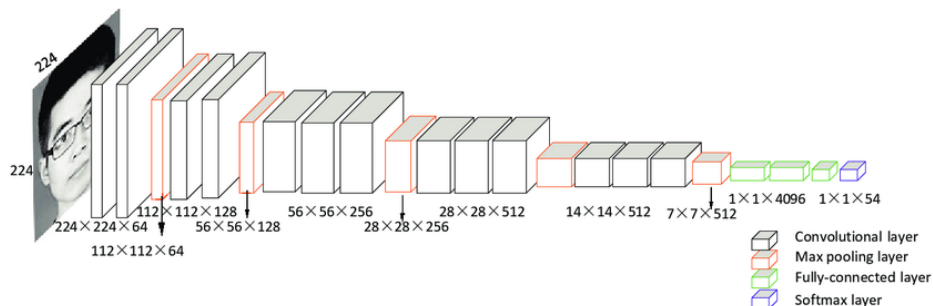


Figure 3.4.2.1 VGG16 architecture[7]

3.4.3 ResNet50

ResNet50 is a 50-layer residual network that has 26 million parameters. Indeed, is a deep convolution neural network model that Microsoft introduced in 2015. In the residual network, instead of learning features, we learn residuals, which are the learnt features removed from the input layers. ResNet links the “nth” layer's input directly to the (n+x)th layer, enabling the stacking of further layers and the construction of a deep network. In experiment, will train and improved a pre-trained ResNet50 model[10]. Figure 3.4.3.1 depicts the architecture of ResNet50.

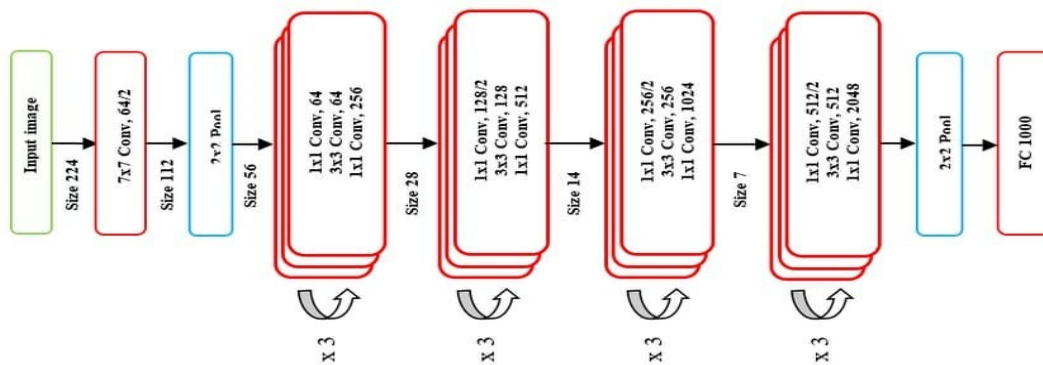


Figure 3.4.3.1 ResNet50 Architecture[7]

3.5 Evaluation Methods

The research project will use a confusion matrix such as precision, recall, and F1 score approach to assess the accuracy of each class after model selection and training. Following training, the MTCNN approach will be used to randomly choose or select particular photos to assess the accuracy of face emotion classes.

3.5.1 Accuracy

Accuracy is one of the most widely used metrics for evaluating the performance of classification models. It provides insight into the percentage of correctly classified facial expressions, indicating how well the model distinguishes between different expressions[8]–[10]. The following equation shows how to determine a model's accuracy:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

3.5.2 Precision

Precision in the context of facial expression classification is represented as the ratio of correctly classified images of a particular expression (True Positives, TP) to the total number of images predicted to have that expression (TP + False Positives, FP) [13].

$$\text{Precision} = \frac{TP}{TP + FP}$$

3.5.3 Recall

Recall, also known as sensitivity, is defined as the ratio of correctly classified images of a particular expression (TP) divided by the total number of images that actually belong to that expression (TP + False Negatives, FN) [13].

$$\text{Recall} = \frac{TP}{TP + FN}$$

3.5.4 F1 Score

The F1 score, also known as the F-measure, represents the balance between precision and recall[13]. It is particularly useful when the class distribution is imbalanced, as it provides a single metric that accounts for both false positives and false negatives.

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

3.5.5 Multi-task Cascaded Convolutional Neural Network (MTCNN)

MTCNN method is one of those technologies that revolutionized the field of face detection and recognition. Introduced in 2016, the MTCNN technique uses a cascading sequence of neural networks to accurately and quickly detect, align, and extract face features from digital photos. MTCNN is a deep learning approach toward face recognition and alignment that detects and locates faces in digital images or videos by cascading a series of convolutional neural networks. The system can identify faces of different sizes and angles as well as resist variations in lighting, facial expressions, and occlusions. The MTCNN technique comprises three steps, including the proposal network, the refinement network, and the output network.

The first step of the MTCNN system, the Proposal Network, produces potential bounding boxes to detect potential face areas in a photograph. Convolutional filters are used to create feature

maps, while fully connected layers are employed to predict not only the likelihood of the face but also the coordinates of the square about it. The candidate regions thus identified are the areas from where the dimensions are higher than the threshold level and have a facial potential, while the rest are discarded. The Refinement Network is the second step of the MTCNN method that refines the P-Net-produced candidate bounding boxes. It measures these regions to determine whether they are faces or non-faces and then refines the dimensions. Output Network (O-Net): The O-Net is MTCNN's last step, where the bounding boxes are refined and face landmarks extracted. It takes the areas improved by the R-Net and classifies them, refines the bounding box locations, and finds the coordinates of major facial features like the eyes, nose, and mouth.

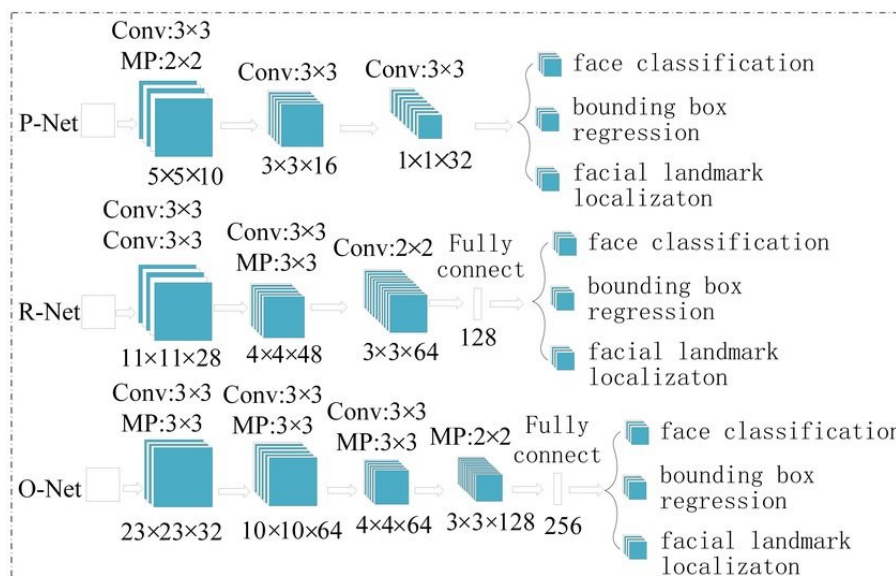


Figure 3.5.6.1 Network structure of MTCNN[10], [11]

3.6 Software Requirement – Jupyter Notebook and Python

Jupyter Notebook is software that provides a code-writing notebook style to facilitate data exploration and visualization. Python is the most popular language for deep learning and machine learning, and the software supports it. As a result, Jupyter Notebook will be utilized in this research paper to investigate data and view code snippets.

3.7 Summary

In Chapter 3, the research paper describes this project's methods and modeling techniques. We use deep learning methods to classify face expressions. The AffectNet dataset is first collected and pre-processed, the characteristics chosen, and the data divided into training / testing sets. Finally, CNN (VGG16 and ResNet50 structures in specific) are discussed by this chapter as our choices for a chapter based these three different needs: traditional Convolutional Neural Networks would not suffice.

This research's methodologies comprise a rigorous method of model training, evaluation, and fine-tuning by hyperparameter adjustment. Performance indices such as accuracy and recall all serve to measure model performance. For example, modifying data augmentation methods (unicity problem with mode changes) and the number of samples may be one way to improve the performance of a Fitting Model.

The AffectNet dataset consists of more than 0.4 million images categorised manually into eight facial emotion classes: neutral, happy, angry, sad, fear, surprise, disgust, and contempt. The paper also explores how deep learning models such as CNNs, VGG16, and ResNet50 can be used to classify emotions.

Including such standard model evaluation methods as model performance indicators (e.g., accuracy, precision, recall, F1 score and MTCNN algorithm and experimental techniques of freezing/unfreezing model layers. Additionally, it suggests Jupyter Notebook and Phyton as the software tools for data exploration, graph making at every stage, and model creation.

Chapter 4

Data Pre-processing, Modelling and Hyperparameter Tuning

In Chapter 4, we will examine in detail at how deep learning is used to emotion classification using the AffectNet dataset, demonstrating the building of a complex data preparation pipeline and the right implementation of transfer learning. The data is separately saved in directories, training and validation, with eight different emotional classes: angry, contempt, disgust, fear, happy, neutral, sad, and surprised. In this specific arrangement, splitting images 70%-30% between training and testing provides some protection against overfitting, as well as rescaling and flipping to help the model generalize better. We next deliver more advanced models, VGG16 and ResNet50 (both pretrained on ImageNet), as well as a custom CNN that has been pre-trained on ImageNet and fine-tuned for predict emotions. It not only simplifies the training process, but it also improves the reliability of classifying various emotional expressions with high accuracy and computational efficiency, ensuring that our models are strong enough to be effectively validated using a collected corpus of over 30,000 images.

4.1 Data pre-processing

```
: 1 # Define directories
  2 base_dir = 'data3'
  3 train_dir = os.path.join(base_dir, 'train')
  4 val_dir = os.path.join(base_dir, 'val')
```

Figure 4.1.1 Define Directories

In Figure 4.1.1 data pre-processing begins with the creation of directories for training and validation datasets. This stage converts the data to a structured format, making it simpler to access and handle throughout processing. Separating the images into various folders (train and val) allows for distinct processing ways for training and validation.

```
: 1 class_names = ['Angry', 'Contempt', 'Disgust', 'Fear', 'Happy', 'Neutral', 'Sad', 'Surprise']
```

Figure 4.1.2 Define Class Names

In Figure 4.1.1 class identification, the AffectNet dataset divides emotions into eight categories: 'Angry', 'Contempt', 'Disgust', 'Fear', 'Happy', 'Neutral', 'Sad', and 'Surprise'. This categorization helps in the labeling of images, which is required for supervised learning models to properly learn from annotated data.

```

1 for class_name in class_names:
2     class_dir = os.path.join(base_dir, class_name)
3     train_class_dir = os.path.join(train_dir, class_name)
4     val_class_dir = os.path.join(val_dir, class_name)
5
6     # Create class directories if they don't exist
7     os.makedirs(train_class_dir, exist_ok=True)
8     os.makedirs(val_class_dir, exist_ok=True)
9
10    images = os.listdir(class_dir)
11
12    if len(images) == 0:
13        print(f"No images found for class {class_name}. Skipping...")
14        continue
15
16    # Split images into 70% training and 30% validation
17    train_images, val_images = train_test_split(images, test_size=0.3, random_state=42)
18
19    for image in train_images:
20        shutil.move(os.path.join(class_dir, image), os.path.join(train_class_dir, image))
21    for image in val_images:
22        shutil.move(os.path.join(class_dir, image), os.path.join(val_class_dir, image))
23
24    print("Dataset split complete.")
25

```

Dataset split complete.

Figure 4.1.3 Data Split

Emotion_Classification > data3 >

Name	Date modified	Type
Angry	29/07/2024 12:42 AM	File folder
Contempt	29/07/2024 12:42 AM	File folder
Disgust	29/07/2024 12:42 AM	File folder
Fear	29/07/2024 12:42 AM	File folder
Happy	29/07/2024 12:42 AM	File folder
Neutral	29/07/2024 12:42 AM	File folder
Sad	29/07/2024 12:42 AM	File folder
Surprise	29/07/2024 12:42 AM	File folder
train	29/07/2024 12:42 AM	File folder
val	29/07/2024 12:42 AM	File folder

Figure 4.1.4 Data Split folders

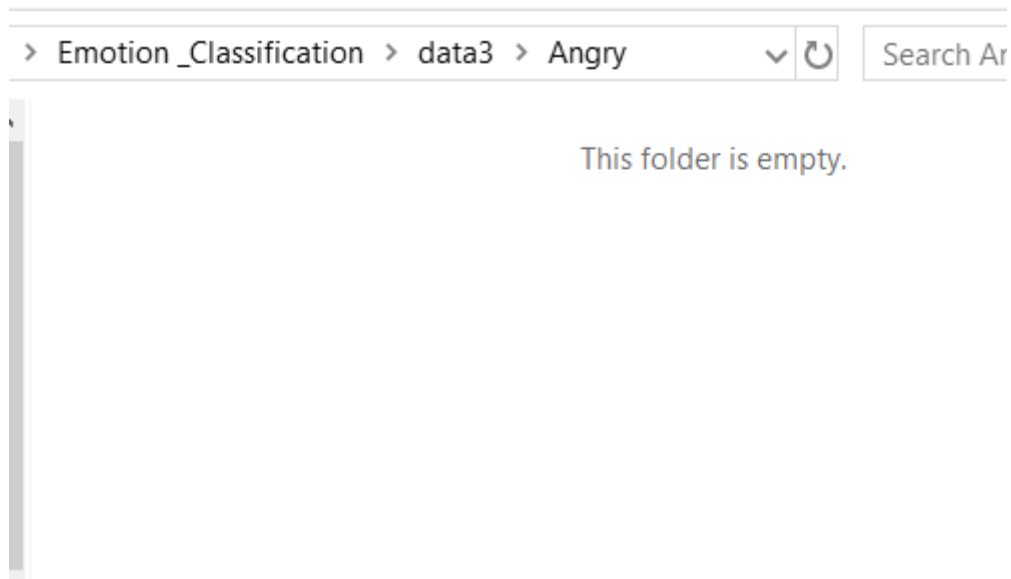


Figure 4.1.5 Angry Data file

In Figure 4.1.3 data splitting for each emotion category, directories are generated in both the training and validation folders. The images are then distributed, usually in a 70-30 ratio between training and validation. After splitting, in Figure 4.1.4 the data will be divided into two folders: train and val. Then, the other file in Figure 4.1.5 will be empty such as Angry file. This split helps to minimize overfitting by training on one set of data and validating on another, allowing to evaluate the model's performance on previously unknown data.

```
1 # Define data generators for train and validation
2 train_datagen = ImageDataGenerator(
3     rescale=1./255,
4     horizontal_flip=True,
5     fill_mode='nearest'
6 )
7 val_datagen = ImageDataGenerator(rescale=1./255)
```

Figure 4.1.6 Data Augmentation

In Figure 4.1.6 data augmentation and rescaling using ImageDataGenerator, images are rescaled by a factor of 1/255 to transform pixel values to a range of 0 to 1, increasing model training efficiency. Augmentations, such as horizontal and vertical flips, are used to the training data to improve its variety, simulate various situations, and strengthen the model.

```
1 # Setup data generators
2 train_generator = train_datagen.flow_from_directory(
3     train_dir,
4     target_size=(224, 224),
5     class_mode='categorical')
6
7 val_generator = val_datagen.flow_from_directory(
8     val_dir,
9     target_size=(224, 224),
10    class_mode='categorical')
```

Found 21001 images belonging to 8 classes.
Found 9005 images belonging to 8 classes.

Figure 4.1.7 Data Generators

Figure 4.1.7 the generators for both training and validation data are then configured to fetch images from their respective directories, resizing them to 224x224 pixels, and setting the class mode to categorical, which means that each image belongs to one of several classes, and the generator will return these labels in a one-hot encoded format. The images are then founded in the folders, with 21,001 in the train set and 9,005 in the test set, each representing one of 8 classes.

4.2 Modelling and Hyperparameter Tuning – VGG16, ResNet50, and CNN Model

VGG16

```
1 from tensorflow.keras.applications import VGG16
2 from tensorflow.keras import layers, Model, optimizers
3
4 # Load VGG16 model
5 vgg16 = VGG16(include_top=False, input_shape=(224, 224, 3), weights='imagenet')
6
7 # Add custom layers on top of VGG16 base
8 x = Flatten()(vgg16.output)
9 x = Dense(512, activation='relu')(x)
10 x = Dropout(0.5)(x)
11 x = Dense(256, activation='relu')(x)
12 x = Dropout(0.5)(x)
13 x = Dense(8, activation='softmax')(x)
14
15 model = Model(inputs=vgg16.input, outputs=x)
16 # Compile the model
17 model.compile(optimizer=optimizers.Adam(learning_rate=0.0001),
18               loss='categorical_crossentropy',
19               metrics=['accuracy'])
20 model.summary()
```

Figure 4.2.1 VGG16 Model Creation

In Figure 4.2.1 VGG16 model creation and compilation, a pre-trained VGG16 model is used, which has been customized for the purpose by deleting the top layer and adding custom dense layers. The model's input shape is configured to accept photos of size 224x224 pixels with three color channels (RGB). This model uses transfer learning, in which a network pre-trained on a large dataset (ImageNet) is fine-tuned for a particular objective (emotion classification). The model has dropout layers to prevent overfitting and a softmax layer for class probability predictions. The model includes an Adam optimizer and a categorical cross-entropy loss function, which are appropriate for multi-class classification applications.

ResNet50

```
1 from tensorflow.keras import layers, Model, optimizers
2
3 resnet50 = ResNet50(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
4
5
6 # Add custom layers on top of the base model
7 x = resnet50.output
8 x = GlobalAveragePooling2D()(x)
9 x = Dense(512, activation='relu')(x)
10 x = Dropout(0.5)(x)
11 x = Dense(256, activation='relu')(x)
12 x = Dropout(0.5)(x)
13 predictions = Dense(8, activation='softmax')(x)
14
15 # Compile the model
16 model = Model(inputs=resnet50.input, outputs=predictions)
17 model.compile(optimizer=Adam(learning_rate=0.0001),
18             loss='categorical_crossentropy',
19             metrics=['accuracy'])
20
21 # Model summary
22 model.summary()
```

Figure 4.2.2 ResNet50 Model Creation

The Figure 4.2.2 process for the loading of layers in the ResNet50 model is mostly the same as in the VGG16. The ResNet50 model is loaded with pre-trained weights from ImageNet but lacks the top (fully connected) layers. The model's input shape is configured to accept 224x224-pixel pictures with three color channels (RGB). The entire ResNet50 model typically has 50 deep layers, including convolutional layers, batch normalization layers, activation layers, pooling layers, and maybe more in its architecture. Then, various custom layers are added on top of the underlying model, including global average pooling, dense layers with ReLU activation, dropout layers for regularization, and a final dense layer with softmax activation for multi-class classification (8 classes). The Adam optimizer is used to build the complete model, using a learning rate of 0.0001 and categorical cross-entropy as the loss function. The performance measure is accuracy. 'model.summary()' generates a summary of the model architecture, outlining each layer, including the number of parameters (trainable and non-trainable), output shapes, and layer connections.

CNN Model

```
1 import tensorflow as tf
2 from tensorflow.keras.models import Sequential
3 from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, BatchNormalization
4
5 model = Sequential()
6
7 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(224,224,3)))
8
9 model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
10 model.add(MaxPooling2D(pool_size=(2, 2)))
11 model.add(Dropout(0.1))
12
13 model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
14 model.add(MaxPooling2D(pool_size=(2, 2)))
15 model.add(Dropout(0.1))
16
17 model.add(Conv2D(256, kernel_size=(3, 3), activation='relu'))
18 model.add(MaxPooling2D(pool_size=(2, 2)))
19 model.add(Dropout(0.1))
20
21 model.add(Flatten())
22 model.add(Dense(512, activation='relu'))
23 model.add(Dropout(0.2))
24
25 model.add(Dense(8, activation='softmax'))
26
27 model.compile(optimizer = 'adam', loss='categorical_crossentropy', metrics=['accuracy'])
28
29
30 model.summary()
31
```

Figure 4.2.3 CNN Model Creation

The Figure 4.2.3 CNN model It begins with an input layer that takes 224x224 color images, then moves on to two convolutional layers with 32 and 64 filters, respectively, followed by a max pooling layer to minimize spatial dimensions and a dropout layer to avoid overfitting. In the deeper levels, this pattern is repeated using ever more complex 128 and 256 filters. After flattening the output of the convolutional stacks, the model has a dense layer with 512 neurons, another dropout, and a softmax layer with 8 units for class predictions.

4.3 Setting up Model Checkpoint and Early Stopping Callbacks

```
1 # Set up callbacks for early stopping and saving the best model
2 checkpoint = ModelCheckpoint('vgg_retrain.h5', monitor='val_accuracy', save_best_only=True, mode='max')
3 early_stopping = EarlyStopping(monitor='val_accuracy', patience=10, restore_best_weights=True)
```

Figure 4.3.1 Model Checkpoint and Early Stopping Callbacks

Figure 4.3.1 above ModelCheckpoint and EarlyStopping callbacks are the same for CNN, VGG16, and ResNet50 when training a neural network. ModelCheckpoint saves the best model

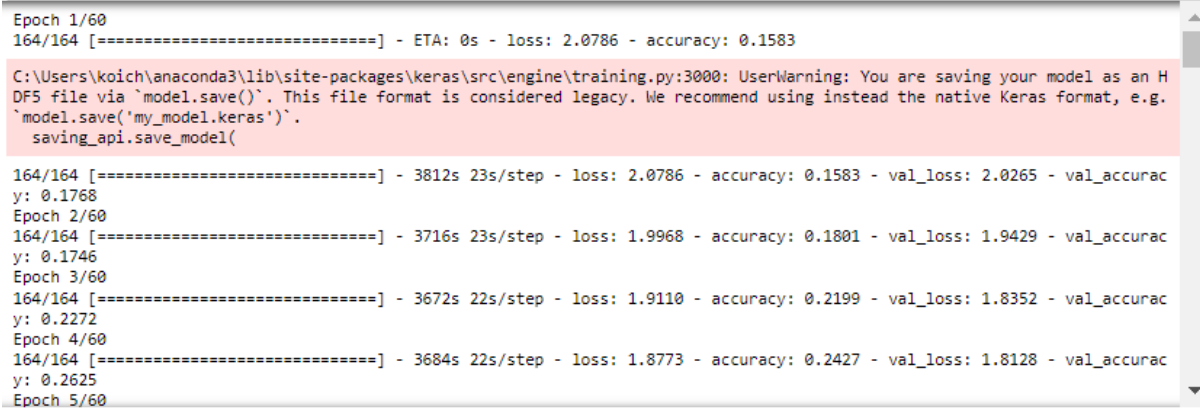
based on the greatest validation accuracy (`val_accuracy`), ensuring that only the best performing model is maintained in the file, depending on the model file name, such as `vgg_retrain.h5`. This is the file that will include the best version of the model. The `EarlyStopping` callback then monitors the same `val_accuracy` and stops the training process if there is no progress after 10 epochs, restoring the weights of the best performing epoch to avoid overfitting and computational waste.

4.4 Training the Model

```

1 # Train the model
2 history = model.fit(
3     train_generator,
4     validation_data=val_generator,
5     steps_per_epoch=train_generator.samples // 128,
6     validation_steps=val_generator.samples // 128,
7     epochs=60,
8     callbacks=[checkpoint, early_stopping]
9 )
10

```



```

Epoch 1/60
164/164 [=====] - ETA: 0s - loss: 2.0786 - accuracy: 0.1583

C:\Users\koich\anaconda3\lib\site-packages\keras\src\engine\training.py:3000: UserWarning: You are saving your model as an H
DF5 file via `model.save()`. This file format is considered legacy. We recommend using instead the native Keras format, e.g.
`model.save('my_model.keras')`.
  saving_api.save_model(

164/164 [=====] - 3812s 23s/step - loss: 2.0786 - accuracy: 0.1583 - val_loss: 2.0265 - val_accurac
y: 0.1768
Epoch 2/60
164/164 [=====] - 3716s 23s/step - loss: 1.9968 - accuracy: 0.1801 - val_loss: 1.9429 - val_accurac
y: 0.1746
Epoch 3/60
164/164 [=====] - 3672s 22s/step - loss: 1.9110 - accuracy: 0.2199 - val_loss: 1.8352 - val_accurac
y: 0.2272
Epoch 4/60
164/164 [=====] - 3684s 22s/step - loss: 1.8773 - accuracy: 0.2427 - val_loss: 1.8128 - val_accurac
y: 0.2625
Epoch 5/60

```

Figure 4.4.1 Train Model

The Figure 4.4.1 approach is used to train the model, using separate data generators for training (`train_generator`) and validation (`val_generator`). It computes the number of steps per epoch and validation steps using a batch size of 128, thereby partitioning the dataset into manageable batches for each epoch. The training is set to run for a maximum of 60 epochs, but it includes the previously defined callbacks `ModelCheckpoint` and `EarlyStopping` to optimize performance by saving the best model and potentially stopping early if validation accuracy does not improve, streamlining the training process and improving model stability.

4.5 Hyperparameter Tuning

```
: 1 train_datagen = ImageDataGenerator(  
2     rescale=1./255,  
3     horizontal_flip=True,  
4     vertical_flip=True,  
5     rotation_range=20,  
6     width_shift_range=0.2,  
7     height_shift_range=0.2,  
8     shear_range=0.15,  
9     zoom_range=0.1,  
10    fill_mode='nearest'  
11 )  
12  
13 val_datagen = ImageDataGenerator(rescale=1./255)
```

Figure 4.5.1 Hyperparameter Tuning Data Augmentation

The Figure 4.5.1 starting point in hyperparameter tuning for CNN, VGG16, and ResNet50 is data augmentation, which involves defining two `ImageDataGenerator` instances for image preprocessing in a machine learning model that is especially designed for data augmentation and normalization. The `train_datagen` is set up for training data and includes several augmentation techniques such as horizontal and vertical flips, rotations up to 20 degrees, 20% width and height shifts, 15% shear transformations, 10% zoom, and filling any new pixels created during transformations with the nearest existing pixel values, all normalized by scaling pixel values to the range $[0, 1]$. In contrast, `val_datagen` for validation data only performs normalization, rescaling pixel values to the same range, and no extra augmentation, guaranteeing model validation on unmodified photos.

VGG16

```
1 # Freeze all layers to use VGG16 as a fixed feature extractor  
2 for layer in vgg16.layers:  
3     layer.trainable = False  
4  
5 # Compile the model with a smaller learning rate  
6 model.compile(optimizer=Adam(learning_rate=0.00001), # Smaller Learning rate for fine-tuning  
7               loss='categorical_crossentropy',  
8               metrics=['accuracy'])  
9 model.summary()
```

Figure 4.5.2 VGG16 Layers Freeze

ResNet50

```
1 # Freeze all but the last 33 layers
2 for layer in resnet50.layers[:-33]:
3     layer.trainable = False
4
5 # Compile the model with a smaller Learning rate
6 model.compile(optimizer=Adam(learning_rate=0.00001), # Smaller Learning rate for fine-tuning
7               loss='categorical_crossentropy',
8               metrics=['accuracy'])
9
10 # Model summary to check which layers are trainable
11 model.summary()
```

Figure 4.5.3 ResNet50 Layers Freeze

CNN Model

```
1 # freeze the first 4 layers
2 for layer in model.layers[:4]:
3     layer.trainable = False
4
5 model.compile(optimizer=Adam(learning_rate=0.00001), # Smaller Learning rate for fine-tuning
6               loss='categorical_crossentropy',
7               metrics=['accuracy'])
```

Figure 4.5.4 CNN Layers Freeze

According to the above Figure 4.5.2, Figure 4.5.3, and Figure 4.5.4, the VGG16, ResNet50, and CNN models will be used in hyperparameter tuning to freeze the layers, the only different is CNN is freezing only the first 4 layers, while ResNet50 is freeze all but the last 33 layers, and VGG16 model consists of 16 layers, 13 convolutional layers and 3 fully connected layers, while all layers are frozen, leaving just the newly added layers (the two dense layers, the dropout layers, and the final output layer) to train. With the learning rate set at 0.00001 for fine-tuning. Setting a lower learning rate, such as 0.00001, for fine-tuning of deep learning models like VGG16 and ResNet50 is critical since it allows for little alterations to the pre-trained weights. This strategy is especially useful when the model's layers are frozen for fine-tuning. Because these models have previously been trained on huge datasets, their weights have come close to optimum solutions for those particular tasks. A low learning rate guarantees that these precisely adjusted weights are not dramatically changed, which might lead to overfitting on the new, often smaller dataset. It also helps to retain the model's generalizability while allowing for subtle learning of additional characteristics relevant to the target task, resulting in increased performance without sacrificing the advantages of the pre-trained model.

4.6 Train Model

```
1 # Train the model
2 history = model.fit(
3     train_generator,
4     steps_per_epoch=train_generator.samples // 128,
5     validation_data=val_generator,
6     validation_steps=val_generator.samples // 128,
7     epochs=30,
8     callbacks=[checkpoint, early_stopping]
9 )
```

Epoch 1/30
164/164 [=====] - ETA: 0s - loss: 0.6951 - accuracy: 0.7534

C:\Users\koich\anaconda3\lib\site-packages\keras\src\engine\training.py:3000: UserWarning: You are saving your model as an file via `model.save()`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')`.
saving_api.save_model(
164/164 [=====] - 734s 4s/step - loss: 0.6951 - accuracy: 0.7534 - val_loss: 1.1690 - val_accuracy: 5996
Epoch 2/30
164/164 [=====] - 716s 4s/step - loss: 0.6984 - accuracy: 0.7508 - val_loss: 1.1769 - val_accuracy: 5826
Epoch 3/30
164/164 [=====] - 712s 4s/step - loss: 0.6915 - accuracy: 0.7541 - val_loss: 1.1997 - val_accuracy: 5857
Epoch 4/30
164/164 [=====] - 716s 4s/step - loss: 0.6819 - accuracy: 0.7578 - val_loss: 1.2233 - val_accuracy: 5790
Epoch 5/30
164/164 [=====] - 727s 4s/step - loss: 0.6871 - accuracy: 0.7633 - val_loss: 1.1397 - val_accuracy: 6170

Figure 4.6.1 Train Model

The Figure 4.6.1 approach used to train the model in hyperparameter tuning is same approach as before by using separate data generators for training (`train_generator`) and validation (`val_generator`). It computes the number of steps per epoch and validation steps using a batch size of 128, thereby partitioning the dataset into manageable batches for each epoch. While the different in training is set to run for a maximum only 30 epochs, but it includes the previously defined callbacks `ModelCheckpoint` and `EarlyStopping` to optimize performance by saving the best model and potentially stopping early if validation accuracy does not improve, streamlining the training process and improving model stability.

4.7 Summary

In chapter 4, the project will focus on the process of emotion categorization using the AffectNet dataset, which contains images labeled across eight emotion categories: Angry, Contempt, Disgust, Fear, Happy, Neutral, Sad, and Surprise. Data preprocessing begins by organizing images into training and validation directories, ensuring effective image management. The dataset is split, with 70% of the images allocated for training and 30% for validation, to avoid overfitting and validate the model effectively. Data augmentation techniques, such as image

rescaling and flipping, are employed to enhance the dataset's reliability and improve the model's training success.

The image generator is configured to resize images to 224x224 pixels and use categorical class mode, facilitating the model's analysis of pre-labeled images in one-hot encoded forms. The training dataset comprises 21,001 images, while the validation set contains 9,005 images, with all eight emotion categories represented. The chapter thoroughly covers each stage of data preprocessing, ensuring clarity and smooth operation during model training and validation.

In terms of model training, transfer learning is employed using pre-trained models like VGG16, ResNet50, and a custom CNN. These models leverage ImageNet weights, fine-tuning only the higher layers specific to the emotion detection task while freezing the lower layers. For instance, the VGG16 model replaces its original top layers with customized dense and dropout layers, concluding with a softmax layer for class predictions. Similarly, the ResNet50 model freezes all but the last 33 layers to allow for task-specific learning. The custom CNN freezes the first four layers, expediting training while adapting the deeper layers for emotion classification.

Training incorporates callbacks such as ModelCheckpoint and EarlyStopping to optimize model performance and prevent overfitting. Data generators manage image inputs and augmentations, ensuring consistency in model evaluation. Hyperparameter tuning plays a crucial role in balancing the layers' freezing and adjusting learning rates, particularly during fine-tuning. This approach enhances the model's ability to learn new insights while retaining the robust knowledge acquired from pre-trained datasets like ImageNet.

Overall, the project demonstrates the effectiveness of using pre-trained models in emotion detection, achieving high accuracy with reduced computational costs and complexity.

Chapter 5

Experimental Result and Discussion

5.1 Experimental Result – VGG16 (Fully connected layers)

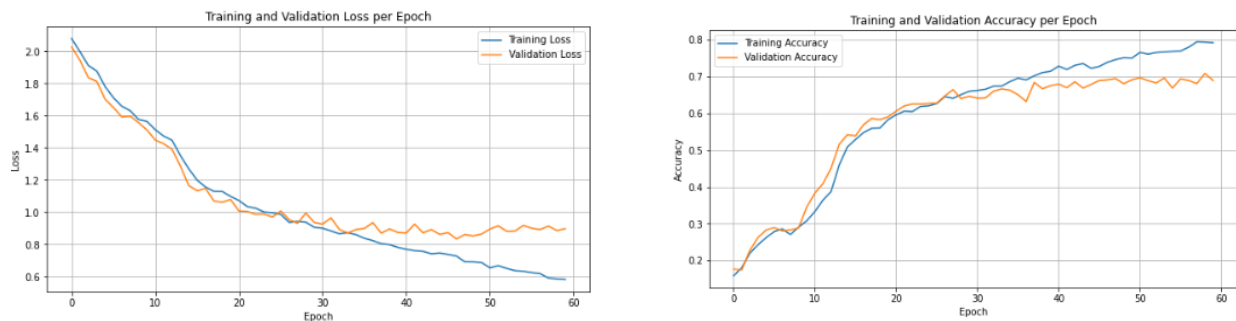


Figure 5.1.1 VGG16 Loss and Accuracy Line Graphs

Figure 5.1.1 above demonstrate the training and validation loss and accuracy for a VGG16 neural network model. The first graph shows a continuous reduction in both training and validation loss, which indicates that the model is learning from the training data. However, the validation loss varies somewhat, indicating that the model may be overfitting since it matches the training data better than the validation data. The second graph shows an improvement in both training and validation accuracy, with the training accuracy plateauing at about 69%. This mismatch in training and validation performance points to overfitting, in which the model performs better on training data than on unseen validation data.

	precision	recall	f1-score	support
Angry	0.54	0.55	0.54	1007
Contempt	0.71	0.68	0.69	862
Disgust	0.47	0.57	0.51	778
Fear	0.55	0.52	0.54	991
Happy	0.91	0.94	0.93	1590
Neutral	0.88	0.90	0.89	1575
Sad	0.56	0.56	0.56	951
Surprise	0.63	0.53	0.58	1251
accuracy			0.69	9005
macro avg	0.66	0.66	0.65	9005
weighted avg	0.69	0.69	0.69	9005

Figure 5.1.2 VGG16 Classification Report

The classification report in Figure 5.1.2 shows that the model performs well in identifying 'Happy' and 'Neutral' emotions, with high precision, recall, and F1-scores. However, it struggles with emotions like 'Disgust', 'Fear', and 'Angry' where the scores are lower, indicating less accurate predictions. The overall accuracy of the model is 69%, suggesting decent performance, but there is room for improvement, particularly in handling emotions with lower scores. The weighted and macro averages highlight the model's ability to maintain reasonable balance across all emotion classes despite varying sample sizes.

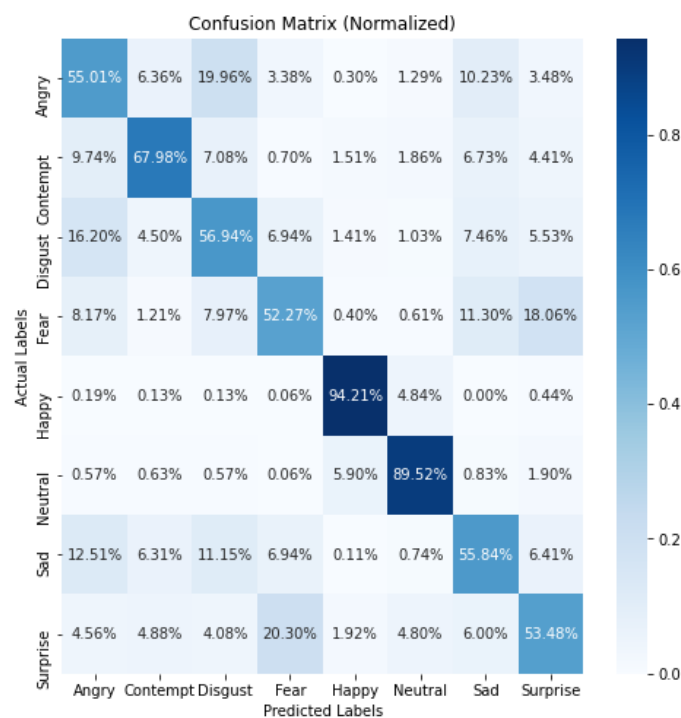


Figure 5.1.3 VGG16 Confusion Matrix

The confusion matrix provided in Figure 5.1.3 demonstrates the performance in identifying 'Happy' (94.21%) and 'Neutral' (89.52%), with relatively accurate predictions. However, it struggles with negative emotions, particularly 'Angry' (55.01%), 'Disgust' (56.94%), and 'Fear' (52.27%), which are often misclassified into each other. 'Sad' has moderate accuracy (55.84%) but is sometimes confused with 'Surprise'. Similarly, 'Surprise' is correctly identified (53.48%) of the time but tends to be mistaken for 'Fear' and 'Angry'. Overall, the model is highly effective for positive emotions but faces challenges in clearly distinguishing between negative emotions.



Figure 5.1.4 VGG16 Predicted Images Results

In the above Figure 5.1.4 randomly generated images, the images show examples where the model correctly predicted the emotional labels. For 'Contempt', 'Fear', 'Happy', and two instances of 'Neutral', the model's predictions match the original labels, demonstrating its ability to accurately classify these emotions in these specific cases. This reflects the model's strength in detecting both positive emotions like 'Happy' and more neutral or negative emotions such as 'Contempt' and 'Fear' when the facial expressions are clear and distinct.

VGG16 – Hyperparameter Tuning (Freeze all layers)

	precision	recall	f1-score	support
Angry	0.54	0.53	0.53	1007
Contempt	0.70	0.70	0.70	862
Disgust	0.52	0.51	0.52	778
Fear	0.55	0.54	0.55	991
Happy	0.93	0.93	0.93	1590
Neutral	0.88	0.90	0.89	1575
Sad	0.58	0.57	0.58	951
Surprise	0.61	0.64	0.63	1251
accuracy			0.70	9005
macro avg	0.67	0.66	0.66	9005
weighted avg	0.70	0.70	0.70	9005

Figure 5.1.5 VGG16 Tuning Classification Report

In Figure 5.1.5 classification report uses hyperparameter tuning to freeze all of the layers, the model performs best for 'Happy' and 'Neutral' emotions, with high precision, recall, and F1-scores around 0.93 and 0.89, respectively. For other emotions like 'Contempt', 'Fear', and 'Surprise', the model performs moderately. While 'Angry', 'Disgust', and 'Sad' have lower

precision and recall values, around 0.50 to 0.58, indicating some difficulties in accurately classifying these emotions. The overall accuracy of the model is 70%, with macro and weighted averages around 0.66 to 0.70, showing that the model has balanced but slightly better performance on certain emotions like 'Happy' and 'Neutral' while struggling with others.

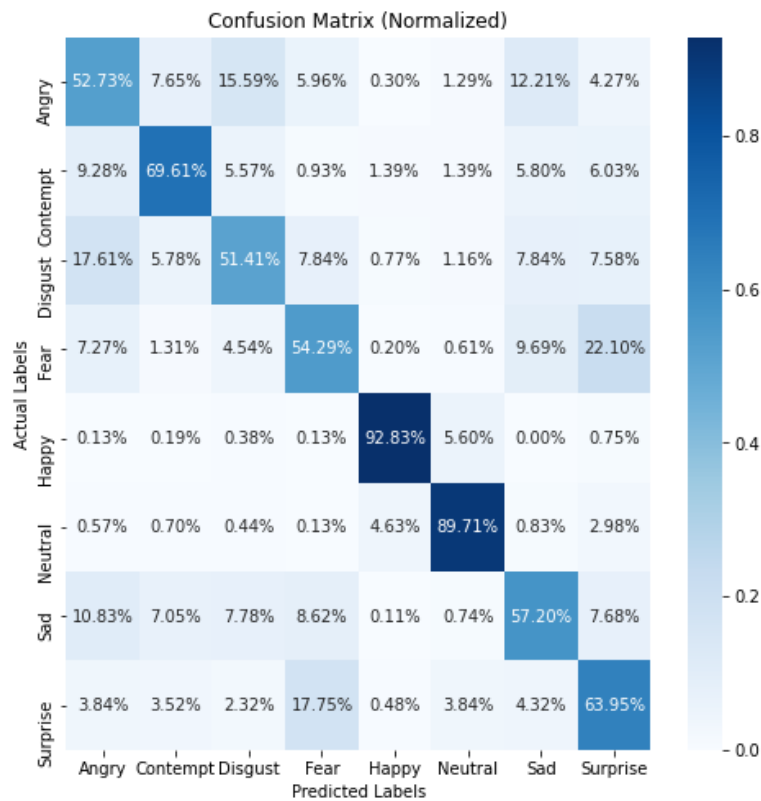


Figure 5.1.6 VGG16 Tuning Confusion Matrix

In Figure 5.1.6 confusion matrix shows the model performs well in classifying 'Happy' (92.83%) and 'Neutral' (89.71%) emotions with high accuracy. However, the model struggles with distinguishing between negative emotions like 'Angry' (52.73%), 'Disgust' (51.41%), and 'Fear' (54.29%), as there is significant misclassification between these categories. For example, 'Angry' is often confused with 'Contempt' (7.65%) and 'Disgust' (15.59%), while 'Fear' is frequently misclassified as 'Surprise' (22.10%). 'Sad' has moderate accuracy (57.20%) but is often confused with 'Surprise' (7.68%). 'Disgust', the model correctly classifies it (51.41%) of the time. However, there is a significant amount of misclassification, particularly with 'Fear' (11.67%), 'Angry' (15.59%), and 'Contempt' (5.57%). Overall, the model performs well for

positive and neutral emotions but faces challenges with more complex or overlapping negative emotions.



Figure 5.1.7 VGG16 Tuning Predicted Images Results

The Figure 5.1.7 shows the images illustrate both correct and incorrect predictions by the model. The model correctly predicted 'Contempt', 'Neutral', and 'Disgust' in three of the cases, indicating its accuracy for these emotions in certain instances. However, it misclassified 'Sad' as 'Contempt' and 'Surprise' as 'Fear', showing some confusion between emotions with overlapping features. These misclassifications highlight the model's challenges in differentiating between similar or intense emotional expressions.

5.2 MTCNN Experimental Results – VGG16

The proposed face detection algorithm addresses two primary challenges: determining whether an image contains a face and extracting the face's positional information. Most face detection algorithms follow a two-step process. First, they identify all potential face-containing regions in the image, and second, they select the regions with the highest likelihood of containing a face. The MTCNN algorithm utilizes PNet to detect candidate face regions, and RNet and ONet to further refine these regions, selecting those with the greatest probability of containing a face. While this method excels in detection accuracy, it may not always be optimal in terms of speed. This research analyzes the time consumption of the MTCNN network at each stage. Moreover, the MTCNN model is combined with a previously trained VGG16 architecture to predict facial expressions—such as happiness, sadness, and anger—for improved detection performance.

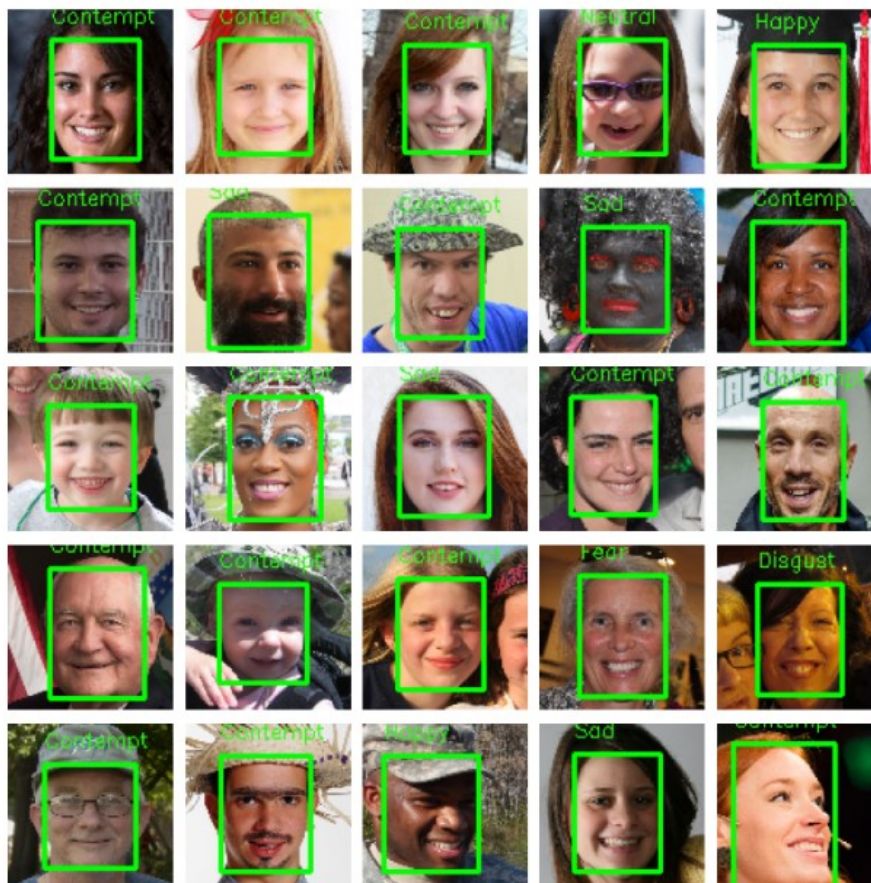


Figure 5.2.1 MTCNN – VGG16 Happy Datasets Predicted Results

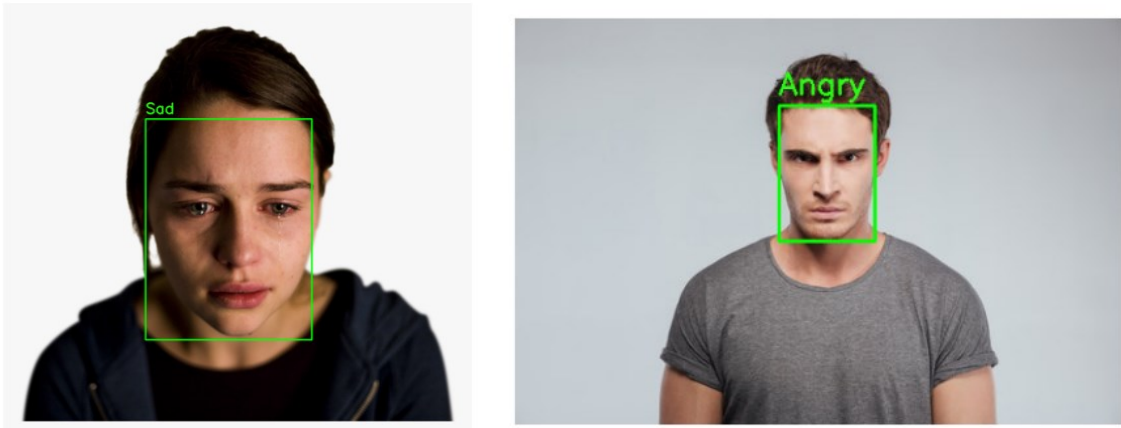


Figure 5.2.2 MTCNN – VGG16 Sad and Angry Images Predicted Results



Figure 5.2.3 MTCNN – VGG16 Group Image Predicted Results



Figure 5.2.4 MTCNN – VGG16 Group Image Predicted Results

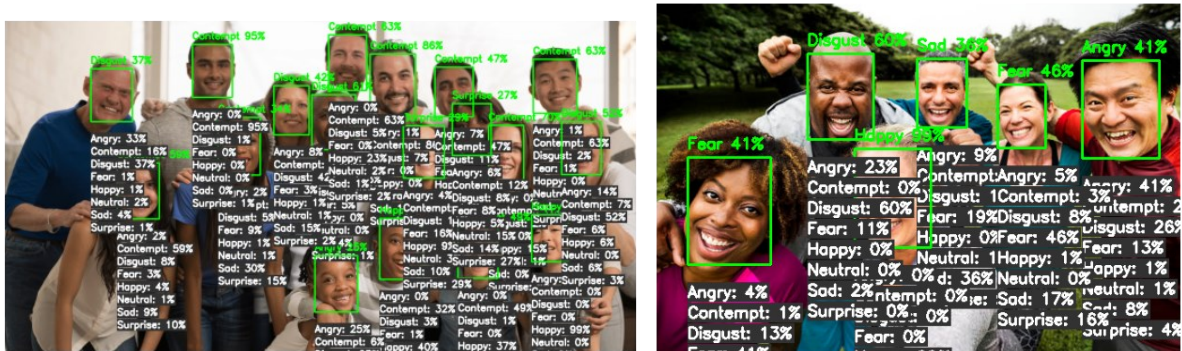


Figure 5.2.5 MTCNN – VGG16 Group Image Predicted Results

The experimental findings in Figure 5.2.1 until 5.2.5 above employ the MTCNN model in combination with the VGG16 architecture to categorize facial emotions. The answers mostly indicate emotions like 'contempt', 'angry', 'sad' and 'surprise'. However, a deeper look at the facial expressions represented in the images indicates that the majority of them are indications of 'happiness'. The difference between anticipated and real expressions suggests a possible

misalignment or inaccuracy in the model's capacity to appropriately categorize emotions. Such discrepancies indicate that, although the model can recognize some negative emotions, it may have difficulty recognizing and classifying signs of pleasure.

5.3 Experimental Result – ResNet50 (Fully connected layers)

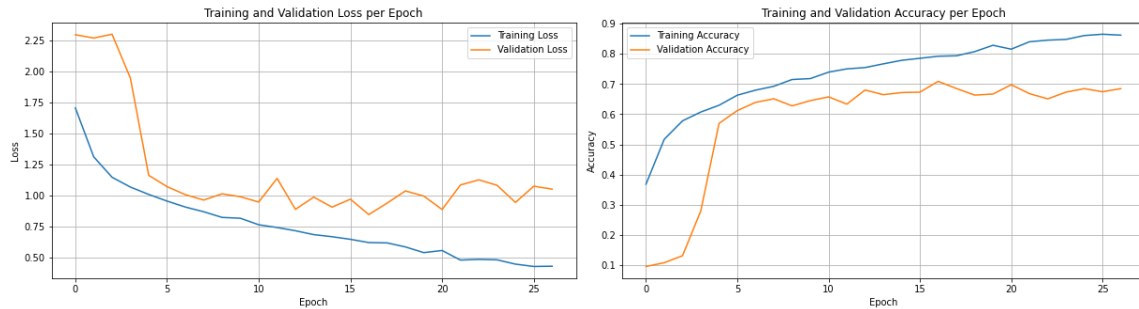


Figure 5.3.1 ResNet50 Loss and Accuracy Line Graphs

The Figure 5.3.1 shows the training and validation outcomes for a ResNet50 model. The first graph, the training loss steadily decreases, indicating that the model is learning and improving during training. However, the validation loss shows a plateau and slight fluctuation after an initial drop, suggesting some overfitting. The second graph, the training accuracy improves consistently, while the validation accuracy initially rises but then fluctuates, suggesting that while the model performs well on the training data, it may be struggling to generalize to the validation set.

	precision	recall	f1-score	support
Angry	0.54	0.58	0.56	1007
Contempt	0.68	0.70	0.69	862
Disgust	0.61	0.43	0.51	778
Fear	0.56	0.51	0.54	991
Happy	0.94	0.90	0.92	1590
Neutral	0.87	0.90	0.89	1575
Sad	0.50	0.66	0.57	951
Surprise	0.64	0.59	0.61	1251
accuracy			0.69	9005
macro avg	0.67	0.66	0.66	9005
weighted avg	0.70	0.69	0.69	9005

Figure 5.3.2 ResNet50 Classification Report

The Figure 5.3.2 classification report shows that the model performs best for 'Happy' and 'Neutral', with high precision, recall, and F1-scores above 0.87. 'Contempt' and 'Surprise' have

moderate performance, with F1-scores of 0.69 and 0.61, respectively. The model struggles with 'Disgust' and 'Fear', with lower recall and F1-scores around 0.51. 'Angry' and 'Sad' show average performance, with F1-scores of 0.56 and 0.57. The overall accuracy of the model is 69%, with a macro average F1-score of 0.66, indicating a balanced but moderate performance across all emotions.

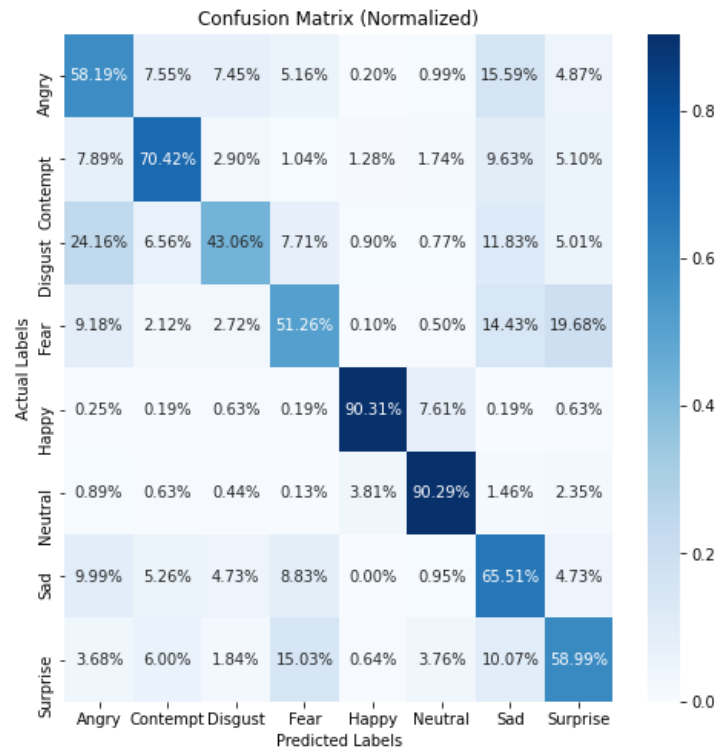


Figure 5.3.3 ResNet50 Confusion Matrix

In Figure 5.3.3 the confusion matrix shows that the model performs well in classifying 'Happy' and 'Neutral', with 90.31% and 90.29% accuracy, respectively. However, it struggles with emotions like 'Disgust' (43.06%) and 'Fear' (51.26%), which are often confused with 'Angry' or 'Surprise'. 'Angry' is correctly classified 58.19% of the time but is frequently misclassified as 'Sad' (15.59%). 'Sad' is identified 65.51% of the time but also confused with other emotions. 'Surprise' has moderate accuracy (58.99%) but shows confusion with other emotions too. Overall, the model performs best for positive and neutral emotions but has difficulty distinguishing between similar negative emotions.



Figure 5.3.4 ResNet50 Predicted Images Results

The Figure 5.3.4 shows the images illustrate both incorrect and correct predictions by the model. The model correctly predicted 'Neutral', 'Happy' and 'Surprise' in three of the cases, indicating its accuracy for these emotions in certain instances. However, it misclassified 'Surprise' as 'Fear' and 'Fear' as 'Surprise', showing some confusion between emotions with overlapping features. These misclassifications underscore the model's difficulties in distinguishing between similar or highly expressive emotions.

ResNet50 – Hyperparameter Tuning (Freeze all layers)

	precision	recall	f1-score	support
Angry	0.55	0.61	0.58	1007
Contempt	0.68	0.71	0.70	862
Disgust	0.59	0.53	0.56	778
Fear	0.55	0.60	0.58	991
Happy	0.94	0.94	0.94	1590
Neutral	0.90	0.90	0.90	1575
Sad	0.60	0.58	0.59	951
Surprise	0.67	0.61	0.64	1251
accuracy			0.72	9005
macro avg	0.69	0.69	0.68	9005
weighted avg	0.72	0.72	0.72	9005

Figure 5.3.5 ResNet50 Tuning Classification Report

The Figure 5.3.5 shows the classification metrics for a ResNet50 model that performs very well in detecting 'Happy' and 'Neutral' emotions, with high precision, recall, and F1-scores around 0.90 and above. 'Contempt' and 'Surprise' have moderate performance, with F1-scores of 0.70 and 0.64, respectively. 'Angry', 'Fear', and 'Sad' have lower scores, with F1-scores ranging from 0.58 to 0.60, indicating some difficulty in identifying these emotions. 'Disgust' is the most challenging emotion for the model, with an F1-score of 0.56. Overall, the model achieves an accuracy of 72%, with a macro average F1-score of 0.68, showing that while the model is

effective, especially with positive emotions, it struggles with negative or more subtle emotional distinctions.

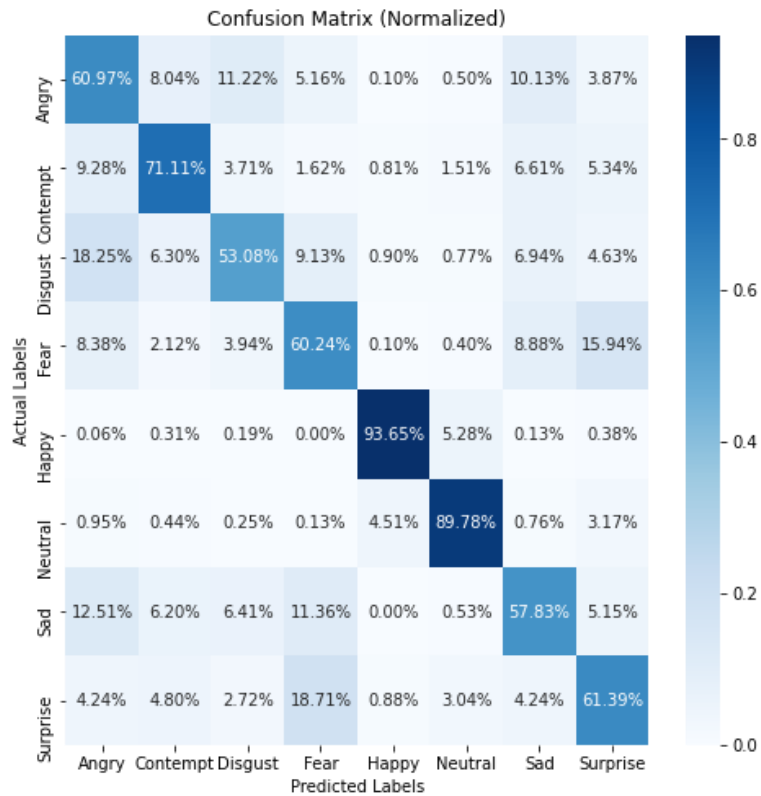


Figure 5.3.6 ResNet50 Tuning Confusion Matrix

The Figure 5.3.6 confusion matrix shows that the model performs well in classifying 'Happy' (93.65%) and 'Neutral' (89.78%) emotions with high accuracy. However, the model struggles with distinguishing between negative emotions like 'Angry' (60.97%) and 'Disgust' (53.08%), with considerable misclassification into other emotions. 'Sad' is correctly classified (57.83%) of the time but often confused with other emotions too. 'Surprise' itself has a moderate accuracy of (61.39%) but is frequently mistaken for 'Fear' (18.71%). Overall, the model excels at positive and neutral emotions but faces challenges in differentiating between negative emotions, leading to some misclassifications.

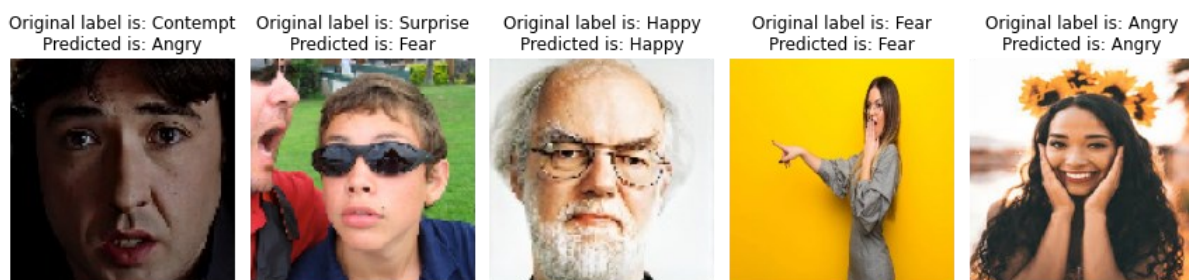


Figure 5.3.7 ResNet50 Tuning Predicted Images Results

The Figure 5.3.7 shows the images illustrate both incorrect and correct predictions by the model. The model correctly predicted 'Happy', 'Fear' and 'Angry' in three of the cases, indicating its accuracy for these emotions in certain instances. However, it misclassified 'Contempt' as 'Angry' and 'Surprise' as 'Fear', showing some confusion between emotions with overlapping features. These misclassifications highlight the model's challenges in differentiating between similar or intense emotional expressions.

5.4 MTCNN Experimental Results – ResNet50

The proposed face detection algorithm aims to address two main challenges: detecting whether an image contains a face and extracting the positional information of that face. Most face detection algorithms follow a two-step process. The first step is to identify all potential regions in the image that might contain faces, and the second step is to select the regions that have the highest probability of containing a face. The MTCNN algorithm approaches this by using PNet to detect candidate face regions, followed by RNet and ONet to refine and select the most likely face-containing areas. While this method tends to excel in detection accuracy, it may not always achieve optimal speed. In this research, the time consumption of MTCNN at each level is analyzed. Additionally, the MTCNN model is integrated with a previously trained ResNet50 architecture to predict facial expressions, such as happiness, sadness, and anger, for enhanced detection performance.

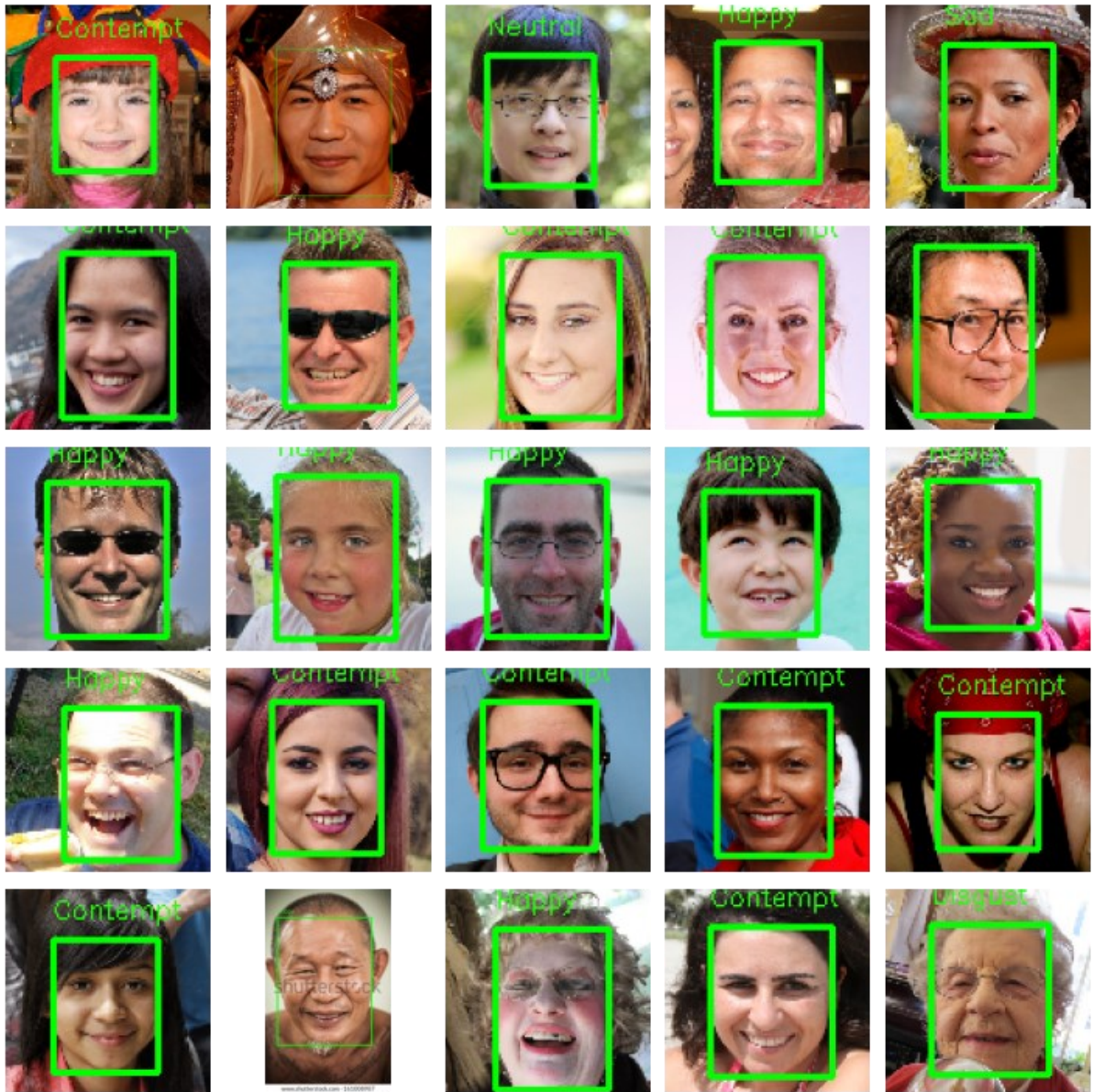


Figure 5.4.1 MTCNN – ResNet50 Happy Datasets Predicted Results

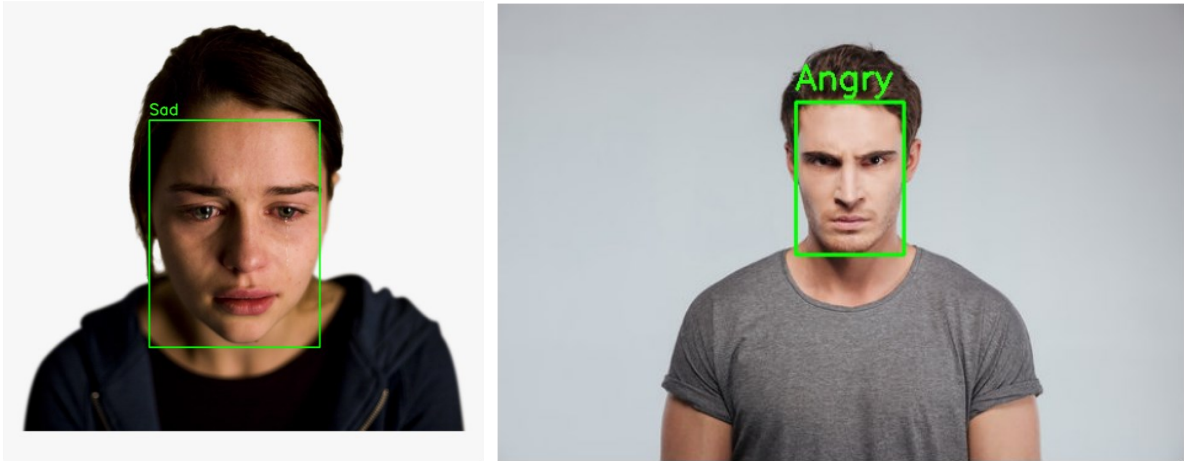


Figure 5.4.2 MTCNN – ResNet50 Sad and Angry Images Predicted Results



Figure 5.4.3 MTCNN – ResNet50 Group Image Predicted Results



Figure 5.4.4 MTCNN – ResNet50 Group Image Predicted Results



Figure 5.4.5 MTCNN – ResNet50 Group Image Predicted Results

The outcomes Figure 5.4.1 until 5.4.5 displayed above, generated from predictions made using the MTCNN model in ResNet50 architecture to categorize facial expressions, indicate that Facial Expression Recognition (FER) is highly accurate, especially in recognizing happy emotions. The bulk of the human facial expressions in the images are 'Happy', and as a result, the majority of MTCNN's forecasts are likewise pleased. This constancy highlights the model's ability to recognize 'Happy' emotions. Despite this general accuracy, the model does not always accurately categorize other emotional expressions. These mistakes suggest possible areas for development in the model's capacity to detect and classify all aspects of human emotions more consistently.

5.5 Experimental Result – CNN Model (Fully connected layers)

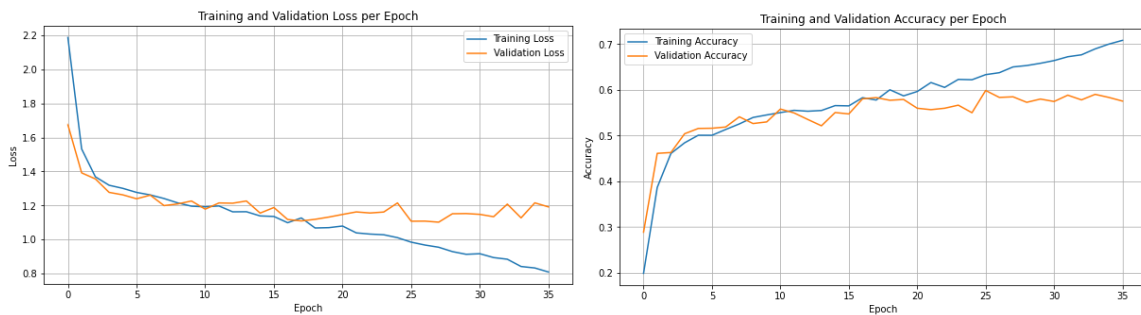


Figure 5.5.1 CNN Loss and Accuracy Line Graphs

The Figure 5.5.1 presents the training and validation loss and accuracy of a CNN model, revealing key aspects of its performance. Initially, both the training and validation losses decrease sharply, indicating rapid learning, and then they gradually converge with the validation loss slightly higher than the training loss, suggesting a decent fit with minimal overfitting. Meanwhile, the accuracy graph shows a quick improvement in training accuracy, plateauing around 60%, while validation accuracy follows a similar pattern but slightly lower, confirming that the model consistently performs well on both the training data and unseen validation data.

	precision	recall	f1-score	support
Angry	0.41	0.48	0.45	1007
Contempt	0.52	0.56	0.54	862
Disgust	0.40	0.27	0.32	778
Fear	0.46	0.27	0.34	991
Happy	0.92	0.83	0.87	1590
Neutral	0.77	0.89	0.83	1575
Sad	0.39	0.41	0.40	951
Surprise	0.46	0.56	0.50	1251
accuracy			0.58	9005
macro avg	0.54	0.53	0.53	9005
weighted avg	0.58	0.58	0.58	9005

Figure 5.5.2 CNN Classification Report

In Figure 5.5.2 above classification report performance of a model across various emotions. The overall accuracy of the model is 58%, with the weighted average F1-score also at 58%, indicating moderate performance. The model performs best for the "Happy" and "Neutral" classes, achieving high precision and recall, with F1-scores of 0.87 and 0.83, respectively. On the other hand, it struggles with classes like 'Disgust' and 'Fear', where precision, recall, and

F1-scores are notably lower, indicating difficulty in correctly identifying these emotions. The macro average for the F1-score is 0.53, reflecting balanced performance across all classes, but with room for improvement, particularly for the less dominant emotions.

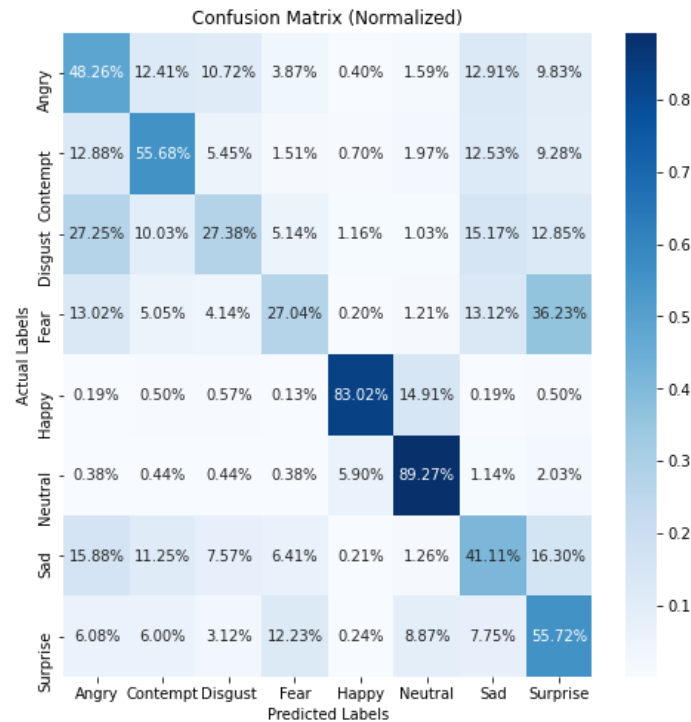


Figure 5.5.3 CNN Confusion Matrix

In Figure 5.5.3 above confusion matrix reveals that the model performs well in recognizing 'Happy' (89.27%) and 'Neutral' (83.02%) emotions, achieving high accuracy for these categories. However, it struggles more with negative emotions like 'Disgust' (27.38%) and 'Fear' (27.04%), where significant misclassification occurs with other emotions. While other emotions performs moderate accuracy. Overall, the model excels in classifying positive and neutral emotions, it has difficulty in distinguishing between several negative emotions, resulting in substantial misclassifications.

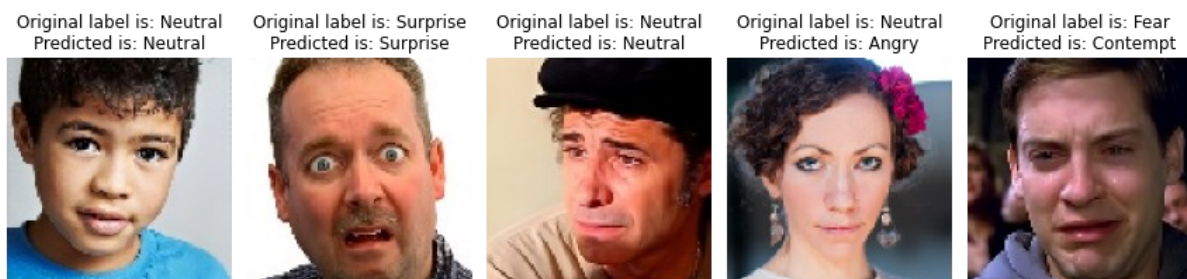


Figure 5.5.4 CNN Predicted Images Results

In Figure 5.5.4 shows the images illustrate both incorrect and correct predictions by the model. The model correctly predicted 'Neutral', 'Surprise' and 'Neutral' in three of the cases, indicating

its accuracy for these emotions in certain instances. However, it misclassified 'Neutral' as 'Angry' and 'Fear' as 'Contempt', showing some confusion between emotions with overlapping features. These misclassifications highlight the model's challenges in differentiating between similar or intense emotional expressions.

CNN Model – Hyperparameter Tuning (Freeze all layers)

	precision	recall	f1-score	support
Angry	0.42	0.46	0.44	1007
Contempt	0.52	0.56	0.54	862
Disgust	0.40	0.29	0.34	778
Fear	0.41	0.37	0.38	991
Happy	0.88	0.88	0.88	1590
Neutral	0.81	0.86	0.83	1575
Sad	0.41	0.42	0.41	951
Surprise	0.47	0.48	0.48	1251
accuracy			0.59	9005
macro avg	0.54	0.54	0.54	9005
weighted avg	0.58	0.59	0.58	9005

Figure 5.5.5 CNN Tuning Classification Report

The Figure 5.5.5 classification report shows the model's performance across various emotions using precision, recall, and F1-score metrics. The model performs best in recognizing 'Happy' and 'Neutral' emotions. However, it struggles with emotions like 'Fear', 'Disgust', and 'Sad', showing lower precision and recall for these categories. The overall accuracy of the model is 59%, with a macro average F1-score of 0.54, indicating moderate performance. The weighted average F1-score of 0.58 reflects that the model performs better with more frequently occurring emotions but has challenges with less frequent or more nuanced emotions, leading to lower precision and recall in some categories.

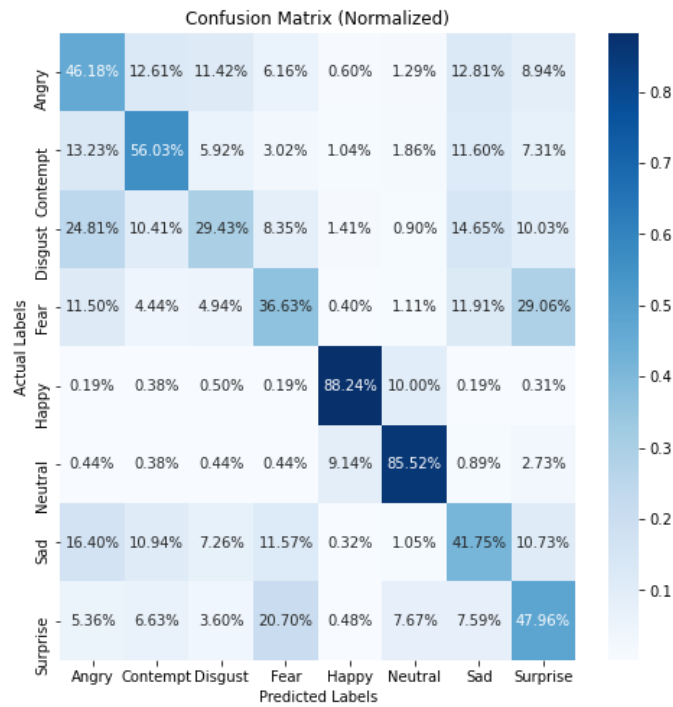


Figure 5.5.6 CNN Tuning Confusion Matrix

In Figure 5.5.6 confusion matrix shows the model's classification performance across various emotions. The model classifies 'Happy' (88.24%) and 'Neutral' (85.52%) with the highest accuracy. However, it struggles with emotions like 'Angry' (46.18%) and 'Disgust' (29.43%), which are often misclassified into other categories. For instance, 'Angry' is frequently confused with 'Disgust' (24.81%), and 'Sad' (41.75%) is often mistaken for 'Angry' (16.40%). 'Surprise' has moderate accuracy at (47.96%) but shows confusion with 'Fear' (20.70%). These misclassifications indicate that the model has difficulty distinguishing between similar or negative emotions, particularly for emotions with overlapping features, while performing well for more distinct or common emotions.

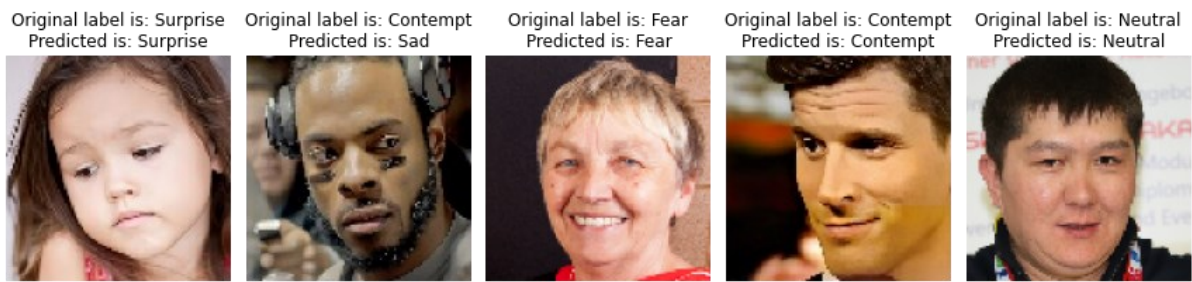


Figure 5.5.7 CNN Tuning Predicted Images Results

In Figure 5.5.7 above shows the images illustrate incorrect and correct predictions by the model. The model correctly predicted 'Surprise', 'Fear', 'Contempt' and 'Neutral' in four of the cases, indicating its accuracy for these emotions in certain instances. However, it misclassified 'Contempt' as 'Sad', showing some confusion between emotions with overlapping features.

5.6 MTCNN Experimental Results – CNN Model

The face detection algorithm is proposed to solve two major problems, one is to detect whether the image contains a face, and the other is how to extract the position information of the face. The process of most face detection algorithms can be divided into two parts. The first step is to find all candidate areas that may contain faces in the image, and the second step is to select the candidate regions containing the highest probability of faces from these candidate regions. The MTCNN algorithm uses PNet to find the candidate area of the face, and uses RNet and Onet to further select the face candidate area with the highest probability. Generally, this kind of algorithm will be outstanding in the detection rate, but the speed will not be too ideal. In the following, the time-consuming of MTCNN network at all levels will be analyzed. This research apply MTCNN model combined with a CNN models architecture which has been trained previously to predict happy face, sad and angry based on the image for better detection.

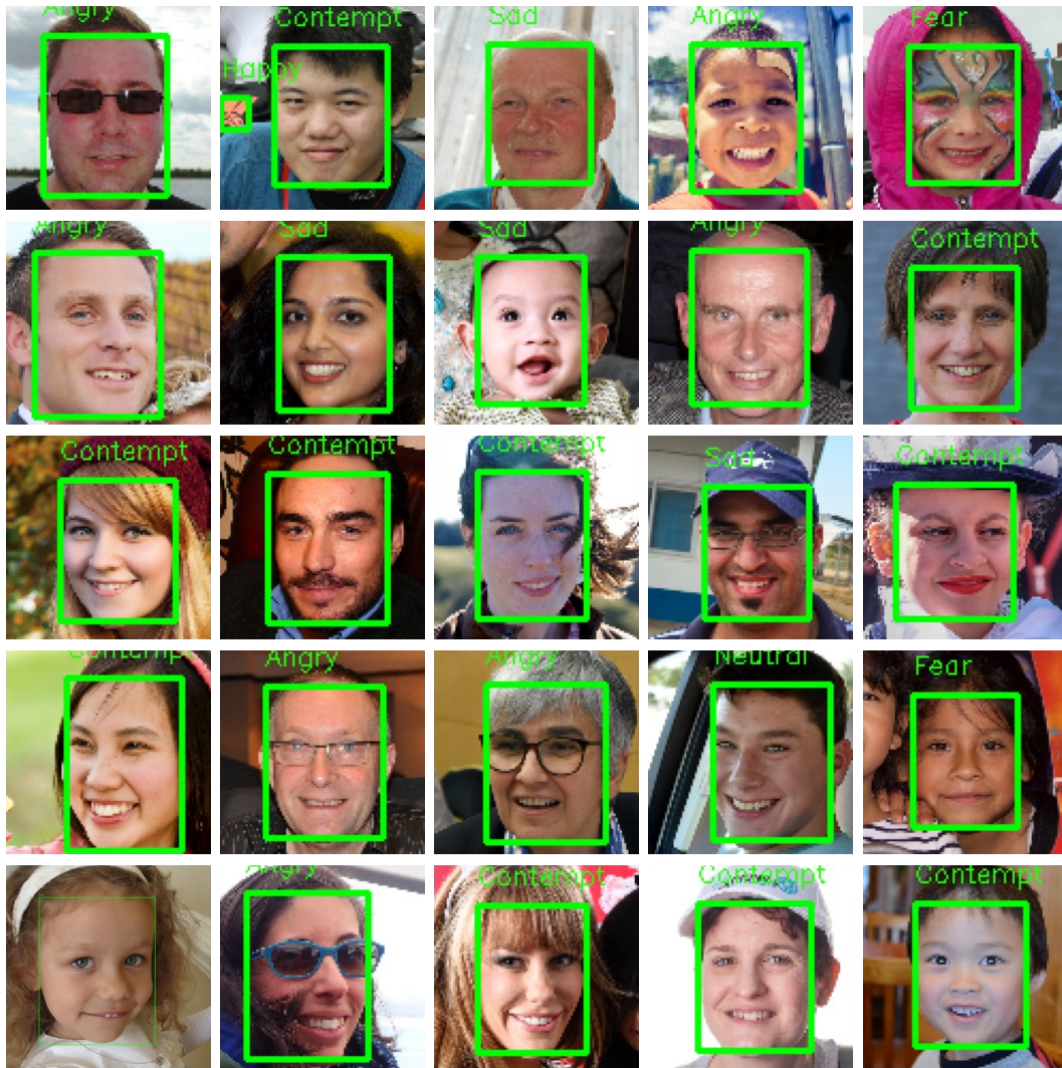


Figure 5.6.1 MTCNN – CNN Happy Datasets Predicted Results

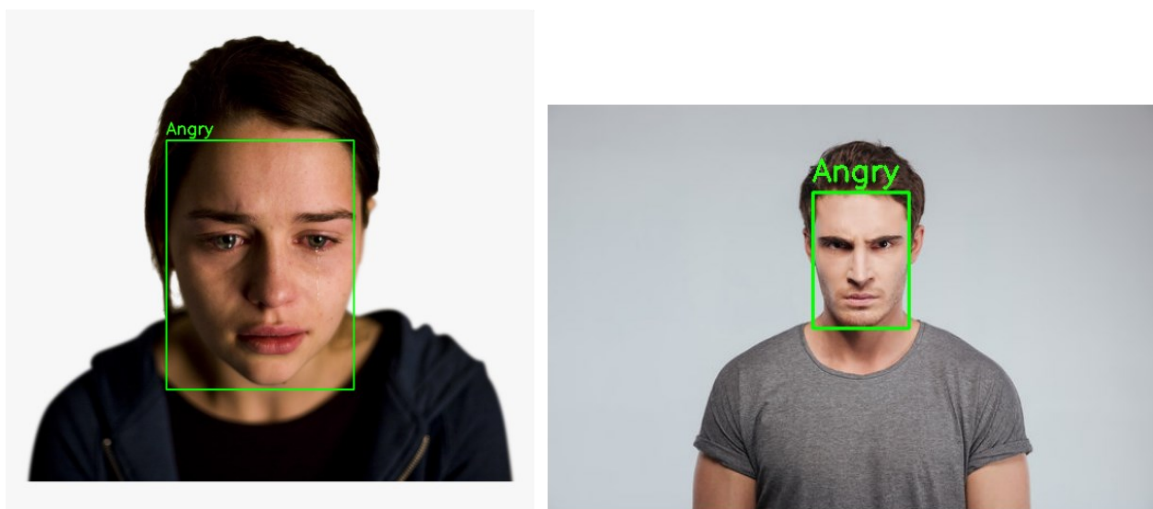


Figure 5.6.2 MTCNN – CNN Sad and Angry Images Predicted Results

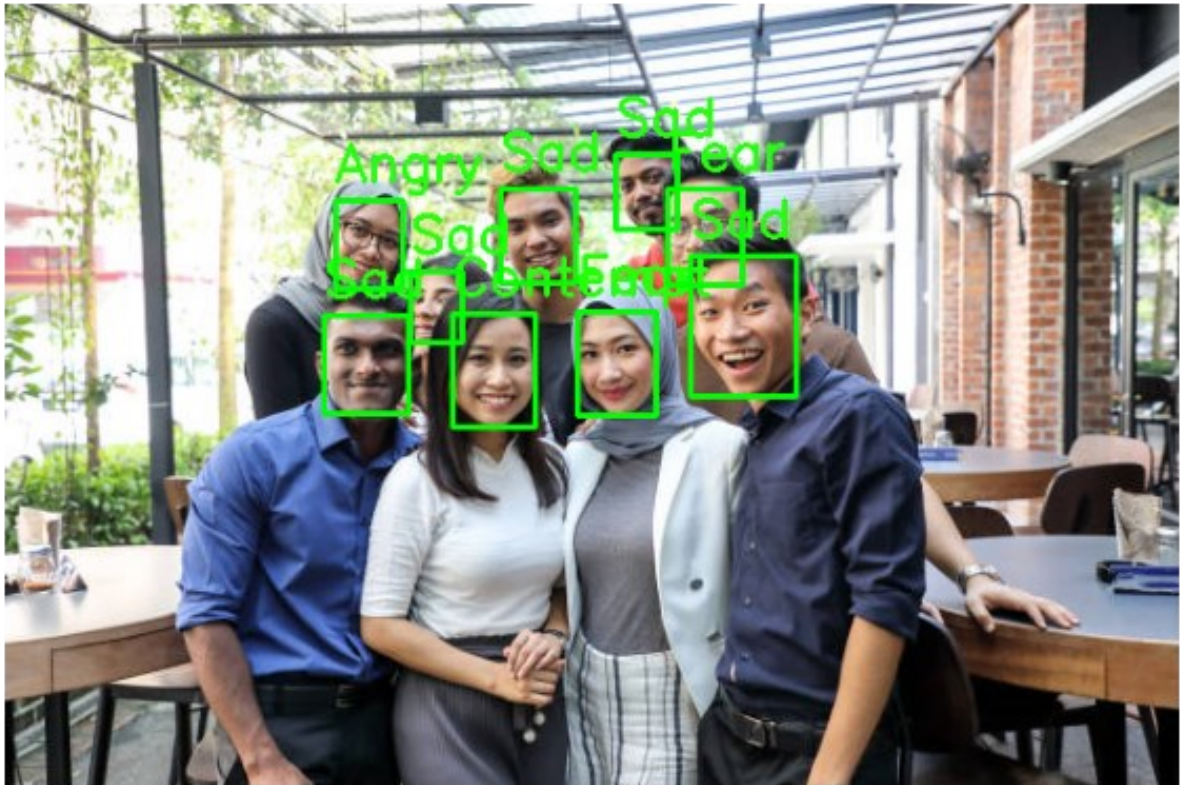


Figure 5.6.3 MTCNN – CNN Group Image Predicted Results



Figure 5.6.4 MTCNN – CNN Group Image Predicted Results

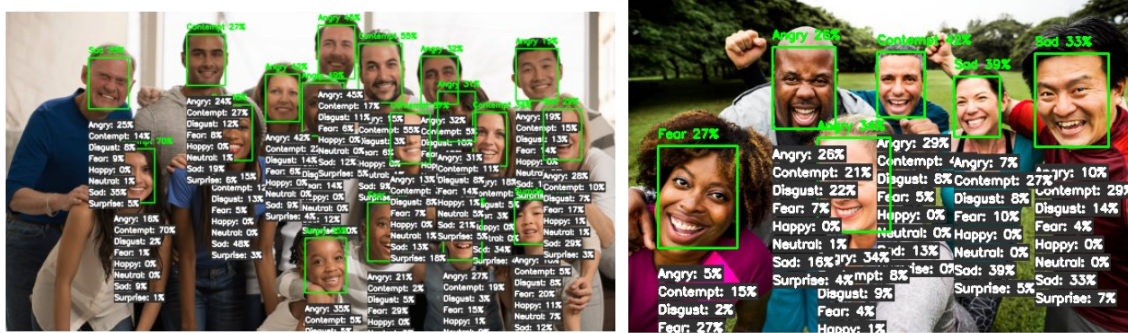


Figure 5.6.5 MTCNN – CNN Group Image Predicted Results

The above Figure 5.6.1 until 5.6.5 results which are generated from predictions made using the MTCNN model combined with a CNN architecture, reveal a serious problem with the accuracy of the Facial Expression Recognition (FER) system. Although the majority of the human face expressions in the photographs are 'Happy', the model commonly misidentifies them as emotions like 'Angry', 'Sad', 'Contempt', and 'Fear'. This continuous mismatch suggests an issue with the CNN model's ability to correctly understand and classify 'Happy' emotions. The problem might be due to flaws in the model's training dataset, insufficient feature extraction, or the model's failure to generalize from training data to real-world events.

5.7 Comparison Performance with Previous Researchers

Table 1.1 Comparison Performance with Previous Researchers

Model	Type	ACC(%)
Proposed Method VGG16	Fully	68.95
	Fine Tuning (Freeze)	69.98
	Freeze	51.39
	Fine Tuning (Unfreeze)	52.69
VGG16 [12]	-	51.11
VGG16 [13]	-	55.20
Proposed Method ResNet50	Fully	69.46
	Fine Tuning (Freeze)	71.72
	Freeze	43.00
	Fine Tuning (Unfreeze)	51.23
ResNet50 [14]	Mish + Accuracy Booster Plus + Weighted Loss	59.72
ResNet50 [15]	Freeze + Fine Tuning	58.00
Proposed Method CNN	Fully	58.42
	Fine Tuning (Freeze)	58.71
	Freeze	57.83
	Fine Tuning (Unfreeze)	58.61
CNN [16]	-	55.09
CNN [12]	-	47.00

Table 1.1 provides a comparative analysis of the performance of different models (VGG16, ResNet50, and CNN) using various training techniques. In the case of the Proposed Method using VGG16, the highest accuracy (69.98%) was achieved through fine-tuning with frozen layers. The fully trained version of the model also performed well with an accuracy of 68.95%. However, when only freezing the layers or unfreezing them for fine-tuning, the accuracy dropped significantly to 51.39% and 52.69%, respectively. These results show that the

proposed method with VGG16 and a fine-tuning approach (freeze) yields better accuracy compared to other strategies.

When compared to previous research using VGG16, the proposed method demonstrates superior performance. For instance, VGG16 from reference [12] achieved an accuracy of 51.11%, and the model in reference [13] reached 55.20%. This indicates that the proposed method's modifications improve the performance of VGG16, especially when using fine-tuning with frozen layers, which significantly surpasses these earlier results.

For the Proposed Method using ResNet50, the fine-tuning (freeze) technique also provided the highest accuracy, reaching 71.72%. The fully trained version of ResNet50 performed slightly lower at 69.46%, but it was still effective. The freeze and fine-tuning (unfreeze) options showed reduced accuracy, with 43.00% and 51.23%, respectively. Comparing this to previous ResNet50 models, such as the one using Mish and Accuracy Booster Plus from reference [14] which achieved 59.72%, and the model in reference [15] that obtained 58.00%, the proposed method shows a notable improvement in accuracy, particularly with the fine-tuning approach.

The Proposed Method using CNN performed slightly lower than VGG16 and ResNet50 but still showed improvements compared to previous research. The fully trained CNN achieved an accuracy of 58.42%, with fine-tuning (freeze) showing a small improvement at 58.71%. Other techniques, such as freezing or unfreezing the layers during fine-tuning, resulted in similar performances at 57.83% and 58.61%. When compared to earlier CNN models, such as the one from reference [16] with 55.09% accuracy, and the model from reference [12] that achieved 47.00%, the proposed method using CNN provides a clear advantage, particularly in its fine-tuning configurations.

These results collectively demonstrate the critical role of training strategies in the performance of neural network models, especially highlighting the benefits and limitations of full training, freezing, and fine-tuning in facial emotion recognition tasks.

5.8 Project Challenges

The project faced several challenges related to model training and accuracy. Time constraints were a significant issue, as the VGG16 and ResNet50 models required extensive training hours up to 71.25 and 23.7 hours respectively when employing various fine-tuning strategies. This lengthy training process underscores the need for more efficient training methods to optimize performance. Additionally, the use of the MTCNN approach in conjunction with VGG16 and CNN models revealed limitations in accurately predicting facial expressions. Despite the models achieving high accuracy in some cases, inconsistencies and misclassifications, particularly in identifying emotions like 'Contempt', 'Angry', 'Sad', and 'Disgust', while the image people facial expression are about 'Happy' and 'Neutral', indicated that the integration of MTCNN did not always enhance performance. These challenges highlight the need for further refinement of the models and training techniques to improve overall accuracy and efficiency in facial emotion recognition tasks.

5.9 Summary

In the chapter 5 summary, the VGG16 model was first analyzed, showing both training and validation loss reduction, indicating effective learning. However, validation loss varied, suggesting overfitting as the model performed better on training data than on unseen data. The classification report revealed that VGG16 excelled in recognizing 'Happy' and 'Neutral' emotions but struggled with emotions such as 'Angry', 'Disgust', and 'Fear', which had lower scores. After hyperparameter tuning with layer freezing, VGG16 achieved its highest accuracy of 69.98%, showing slight improvement. The confusion matrix highlighted the model's accuracy in classifying positive emotions but also its challenges with negative emotions, which were often misclassified into each other.

The ResNet50 model demonstrated a steady decrease in training loss and improved training accuracy, though the validation loss plateaued, with fluctuating accuracy indicating potential overfitting. Similar to VGG16, ResNet50 excelled in recognizing 'Happy' and 'Neutral' emotions with high precision, recall, and F1-scores but showed moderate difficulty with 'Disgust' and 'Fear'. Hyperparameter tuning allowed the model to achieve a slightly higher accuracy, with ResNet50 reaching its peak performance at 71.72%. Despite these improvements, the confusion matrix indicated significant misclassification among negative

emotions such as 'Angry' and 'Sad', revealing room for improvement in distinguishing these categories.

The CNN model displayed rapid learning, with both training and validation losses decreasing and converging, though validation accuracy remained slightly lower than training accuracy. The classification report showed moderate accuracy for the CNN model, with strong performance in identifying 'Happy' and 'Neutral' emotions but lower accuracy for 'Disgust' and 'Fear'. After fine-tuning, the CNN model's best accuracy was 58.71%, indicating moderate performance. The confusion matrix further highlighted the model's challenges with less frequent or overlapping negative emotions, resulting in frequent misclassifications, particularly among emotions like 'Angry', 'Sad', and 'Disgust'.

Overall, VGG16, ResNet50, and CNN models performed well in identifying positive and neutral emotions, but each model faced challenges with negative or subtle emotional distinctions. Fine-tuning and layer-freezing techniques proved beneficial, especially for ResNet50, which reached the highest accuracy among the three models. These results provide insights into the effectiveness and limitations of these neural network architectures for facial emotion recognition tasks.

Chapter 6

Conclusion

6.1 Conclusion

In conclusion, this research offers a comprehensive evaluation of three major deep learning models Convolutional Neural Networks (CNNs), VGG16, and ResNet50 utilizing the AffectNet dataset for the classification of facial expressions. Our detailed testing across various training setups was aimed at determining which model delivers the highest accuracy in recognizing a diverse range of human emotions effectively.

The project conclusively found that the ResNet50 model achieved the best overall performance, particularly when employed with a combination of full training followed by fine-tuning with layers initially frozen. This method, referred to as fully & fine tuning (freeze), led ResNet50 to attain an impressive accuracy of 71.72%, while VGG16 is 69.98% and CNN is only 58.71%. This approach effectively leveraged the robust feature extraction capabilities of ResNet50, initially trained on ImageNet, and then fine-tuned to adapt to the subtleties of emotion classification presented by the AffectNet dataset.

When examining the integration of models with the MTCNN technique, which is designed to enhance facial detection capabilities within the emotion classification task, the results varied. However, the ResNet50 model demonstrated the most consistent improvement in performance when combined with MTCNN. This combination was particularly effective in enhancing the model's ability to detect and classify facial expressions accurately, indicating a promising avenue for applications requiring high precision in emotion recognition.

These findings highlight the importance of selecting the right combination of model architecture and training methodology to tackle the complexities of facial expression classification. The superior performance of the ResNet50 model using the fully & fine tuning (freeze) approach underscores the effectiveness of adapting pre-trained models through fine-tuning for specific tasks, providing a valuable strategy for improving the accuracy and efficiency of models in real-world applications.

In conclusion, this research not only advances our understanding of the capabilities of different neural network architectures in the domain of emotional AI but also provides a clear benchmark for future studies aiming to optimize deep learning models for emotion recognition. By
Bachelor of Computer Science (Honours)
Faculty of Information and Communication Technology (Kampar Campus), UTAR

demonstrating that tailored training approaches can significantly enhance model performance, this project contributes to the broader field of human-computer interaction, opening up new possibilities for sensitive and accurate technology-driven emotional engagement.

6.2 Future Work

Given the restrictions noted, such as time limits and the requirement for sophisticated technology, it is advised to invest in high-performance GPUs to speed up the training of deep learning models on the AffectNet data. The completely and fine tuning (frozen) technique has demonstrated to be the most successful, resulting in better validation accuracy, and should be used as the major training strategy. In addition, using more efficient training approaches, such as distributed computing or improved batch processing, may save time while improving model performance. Further, combining the ResNet50 model with MTCNN has been proven to provide the most accurate results in predicting facial emotions. Afterward, in the future, different models and datasets may be used with MTCNN to predict facial emotions.

REFERENCES

- [1] Kamal and H. Ez-zahraouy, “A comparison between the VGG16 , VGG19 and ResNet50 architecture frameworks for classification of normal and CLAHE processed medical images,” *Res. Sq.*, pp. 0–16, 2023.
- [2] Z. Y. Huang, C. C. Chiang, J. H. Chen, Y. C. Chen, and H. L. Chung, “OPEN A study on computer vision for facial emotion recognition,” *Sci. Rep.*, pp. 1–13, 2023, doi: 10.1038/s41598-023-35446-4.
- [3] T. Vo and G. Lee, “Pyramid With Super Resolution for In-The-Wild Facial Expression Recognition,” vol. 8, 2020, doi: 10.1109/ACCESS.2020.3010018.
- [4] K. Li, “Discussions of Different Deep Transfer Learning Models for Emotion Recognitions,” *IEEE Access*, vol. 10, no. August, pp. 102860–102875, 2022, doi: 10.1109/ACCESS.2022.3209813.
- [5] A. Mollahosseini, S. Member, B. Hasani, and S. Member, “AffectNet : A Database for Facial Expression , Valence , and Arousal Computing in the Wild,” pp. 1–18.
- [6] M. Gupta and S. Vaikole, “Recognition of Human Mental Stress Using Machine Learning Paradigms,” *SSRN Electron. J.*, 2020, doi: 10.2139/ssrn.3571754.
- [7] Z. Pei, H. Xu, Y. Zhang, M. Guo, and Y. Yee-Hong, “Face recognition via deep learning using data augmentation based on orthogonal experiments,” *Electron.*, vol. 8, no. 10, pp. 1–16, 2019, doi: 10.3390/electronics8101088.
- [8] T. Ensari and M. Gunay, “Comparison of face recognition algorithms,” vol. 4, no. 2, pp. 1–4, 2017, doi: 10.1109/siu.2017.7960469.
- [9] J. Cai, Z. Meng, A. S. Khan, Z. Li, J. Oreilly, and Y. Tong, “Island loss for learning discriminative features in facial expression recognition,” *Proc. - 13th IEEE Int. Conf. Autom. Face Gesture Recognition, FG 2018*, no. October, pp. 302–309, 2018, doi: 10.1109/FG.2018.00051.
- [10] L. Zhang, H. Wang, and Z. Chen, “A Multi-task Cascaded Algorithm with Optimized Convolution Neural Network for Face Detection,” *Proc. - 2021 Asia-Pacific Conf. Commun. Technol. Comput. Sci. ACCTCS 2021*, pp. 242–245, 2021, doi: 10.1109/ACCTCS52002.2021.00054.
- [11] S. Jia and Y. Tian, “Face Detection Based on Improved Multi-task Cascaded Convolutional Neural Networks,” vol. 51, no. 2, pp. 67–74, 2024.
- [12] Z. Ullah *et al.*, “Emotion Recognition from Occluded Facial Images Using Deep Ensemble Model,” 2022, doi: 10.32604/cmc.2022.029101.

- [13] S. O. Avcı and O. Akay, "Employment and Investigation of Various CNN Models and Datasets for Facial Expression Recognition and Classification," *14th Int. Conf. Electr. Electron. Eng. ELECO 2023 - Proc.*, no. February, 2023, doi: 10.1109/ELECO60389.2023.10415947.
- [14] F. Y. Rahadika, N. Yudistira, and Y. A. Sari, "Facial Expression Recognition using Residual Convnet with Image Augmentations," *J. Ilmu Komput. dan Inf.*, vol. 14, no. 2, pp. 127–135, 2021, doi: 10.21609/jiki.v14i2.968.
- [15] C. T. Yen and K. H. Li, "Discussions of Different Deep Transfer Learning Models for Emotion Recognitions," *IEEE Access*, vol. 10, pp. 102860–102875, 2022, doi: 10.1109/ACCESS.2022.3209813.
- [16] N. Siddiqui, T. Reither, R. Dave, D. Black, T. Bauer, and M. Hanson, "A Robust Framework for Deep Learning Approaches to Facial Emotion Recognition and Evaluation," *Proc. - 2022 Asia Conf. Algorithms, Comput. Mach. Learn. CACML 2022*, pp. 68–73, 2022, doi: 10.1109/CACML55074.2022.00020.

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y4S3	Study week no.: 1& 2
Student Name & ID: Koi Chin Chong, 20ACB05613	
Supervisor: Dr Zanariah Binti Zainudin	
Project Title: Deep Learning for Image Classification	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- To research the coding techniques using various transfer learning models.

2. WORK TO BE DONE

- Train the VGG model

3. PROBLEMS ENCOUNTERED

- Find out the models and tuning method to be done.

4. SELF EVALUATION OF THE PROGRESS

- Should start to train the model earlier



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y4S3	Study week no.: 3,4&5
Student Name & ID: Koi Chin Chong, 20ACB05613	
Supervisor: Dr Zanariah Binti Zainudin	
Project Title: Deep Learning for Image Classification	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- The VGG16 model under training.

2. WORK TO BE DONE


- At least should finish the VGG16 model training.

3. PROBLEMS ENCOUNTERED


- While during VGG16 model training, the validation accuracy is not high as what I expected.

4. SELF EVALUATION OF THE PROGRESS

- The training process performance not as good. I will try to find out the VGG16 model to find out more possible ways to train the model to achieve high validation accuracy.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y4S3	Study week no.: 6,7&8
Student Name & ID: Koi Chin Chong, 20ACB05613	
Supervisor: Dr Zanariah Binti Zainudin	
Project Title: Deep Learning for Image Classification	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- VGG16 model training done in fully layer & hyperparameter tuning, and able to achieved high accuracy.

2. WORK TO BE DONE


- Complete trained the ResNet50 & CNN model

3. PROBLEMS ENCOUNTERED

- Time constraint while training the VGG16 model.

4. SELF EVALUATION OF THE PROGRESS

- Get recommendation from project supervisor, what feature should be added inside the project.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y4S3	Study week no.: 9&10
Student Name & ID: Koi Chin Chong, 20ACB05613	
Supervisor: Dr Zanariah Binti Zainudin	
Project Title: Deep Learning for Image Classification	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Both models trained done in ResNet50 & CNN model.

2. WORK TO BE DONE


- Evaluate which model can be achieved high accuracy in AffectNet dataset.
- Add MTCNN algorithm feature
- Write the Chapter 1,2 & 3 in the report


3. PROBLEMS ENCOUNTERED

- None.

4. SELF EVALUATION OF THE PROGRESS

- Get recommendation from project supervisor, what should be done or added in the report.


Supervisor's signature


Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y4S3	Study week no.: 11
Student Name & ID: Koi Chin Chong, 20ACB05613	
Supervisor: Dr Zanariah Binti Zainudin	
Project Title: Deep Learning for Image Classification	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Completed Chapter 1,2 & 3.
- Added MTCNN algorithm feature to predict facial expression

2. WORK TO BE DONE

- Write Chapter 4,5 & 6.

3. PROBLEMS ENCOUNTERED

- Difficult in literature review to find the previous researcher model training type.

4. SELF EVALUATION OF THE PROGRESS

- Will try to finish the chapter 4,5 & 6 in next week.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y4S3	Study week no.: 12
Student Name & ID: Koi Chin Chong, 20ACB05613	
Supervisor: Dr Zanariah Binti Zainudin	
Project Title: Deep Learning for Image Classification	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Completed Chapter 4,5 & 6.

2. WORK TO BE DONE


- Write Chapter 7.

3. PROBLEMS ENCOUNTERED


- None.

4. SELF EVALUATION OF THE PROGRESS

- Will finish the chapter 7 by the end of this week 12, and after finished the report in Chapter 1 until Chapter 7, then will be send it to the project supervisor for review in this week.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y4S3	Study week no.: 13
Student Name & ID: Koi Chin Chong, 20ACB05613	
Supervisor: Dr Zanariah Binti Zainudin	
Project Title: Deep Learning for Image Classification	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Completed report
- Completed poster

2. WORK TO BE DONE


- Prepare the presentation slides.

3. PROBLEMS ENCOUNTERED


- None.

4. SELF EVALUATION OF THE PROGRESS

- Prepare well for the presentation.



Supervisor's signature



Student's signature

POSTER



Faculty of Information Communication and Technology

Deep Learning for Image Classification

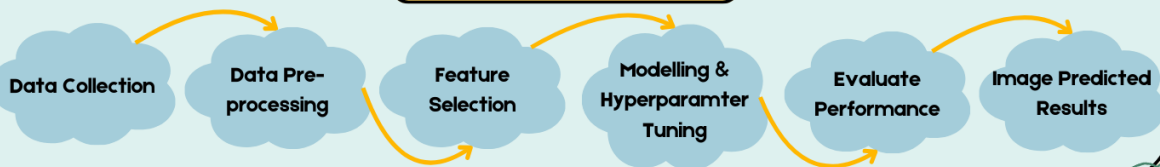
INTRODUCTION

Facial expression image classification leverages deep learning models like convolutional neural networks (CNN) to identify human emotions from photographs. Techniques such as transfer learning VGG16 & ResNet50 and method like k-fold cross-validation to evaluate model's performance..

OBJECTIVES

1. To implement CNN, VGG16, and ResNet50 models with an expanded dataset and increased class diversity to enhance facial expression classification accuracy.
2. To analyse and compare the accuracy of CNN, VGG16, and ResNet50 models in accurately classifying a broad range of facial expressions, using both frozen layers with fine-tuning and fully trained layers with fine-tuning.
3. To evaluate and compare of CNN, VGG16, and ResNet50 models by using MTCNN(Multi-task Cascaded Convolutional Networks) algorithm to predict of these models against a different collection of face photos to discover which model detects facial emotions most accurately.

PROPOSED METHOD



RESULT AFTER HYPERPARAMETER TUNED

MODEL	TYPE	ACC(%)
Proposed Method VGG16	Fully + Fine Tuning (Freeze)	68.95
	Freeze + Fine Tuning (Unfreeze)	69.98
Proposed Method ResNet50	Fully + Fine Tuning (Freeze)	69.46
	Freeze + Fine Tuning (Unfreeze)	71.72
Proposed Method CNN	Fully + Fine Tuning (Freeze)	58.42
	Freeze + Fine Tuning (Unfreeze)	58.71

CONCLUSION

This research evaluates the performance of deep learning models, specifically CNNs, VGG16, and ResNet50, on the AffectNet dataset for facial expression classification. The study found that the ResNet50 model, utilizing a method of full training followed by fine-tuning with initially frozen layers, achieved the highest accuracy at 71.72%, while VGG16 is 69.98% and CNN is only 58.71%. This approach effectively harnessed ResNet50's robust feature extraction capabilities, initially trained on ImageNet, adapted to the nuances of emotion classification. However, ResNet50 is also the most consistent improvement in performance when integrated with the MTCNN technique.

Research Student: Koi Chin Chong
Research Supervisor: Cik Zanariah Binti Zainudin

PLAGIARISM CHECK RESULT

Deep learning for image classification chp1-5.docx

ORIGINALITY REPORT

8% SIMILARITY INDEX	6% INTERNET SOURCES	5% PUBLICATIONS	3% STUDENT PAPERS
-------------------------------	-------------------------------	---------------------------	-----------------------------

PRIMARY SOURCES

1	assets.researchsquare.com Internet Source	1%
2	Submitted to Liverpool John Moores University Student Paper	<1%
3	Kshitiza Vasudeva, Akshat Dubey, Saravanan Chandran. "SCL-FExR: supervised contrastive learning approach for facial expression Recognition", Multimedia Tools and Applications, 2023 Publication	<1%
4	www.mdpi.com Internet Source	<1%
5	"Advanced Network Technologies and Intelligent Computing", Springer Science and Business Media LLC, 2024 Publication	<1%
6	www.nature.com Internet Source	<1%
7	www.techscience.com Internet Source	

		<1 %
8	Submitted to Konsorsium Turnitin Relawan Jurnal Indonesia Student Paper	<1 %
9	www.ijraset.com Internet Source	<1 %
10	Submitted to University of Northumbria at Newcastle Student Paper	<1 %
11	Arvind Dagur, Karan Singh, Pawan Singh Mehra, Dharendra Kumar Shukla. "Artificial Intelligence, Blockchain, Computing and Security", CRC Press, 2023 Publication	<1 %
12	Submitted to UNICAF Student Paper	<1 %
13	Submitted to University of Warwick Student Paper	<1 %
14	Chih-Ta Yen, Kang-Hua Li. "Discussions of different deep transfer learning models for emotion recognitions", IEEE Access, 2022 Publication	<1 %
15	webmail.inb.uni-luebeck.de Internet Source	<1 %

16	Student Paper	<1 %
17	downloads.hindawi.com Internet Source	<1 %
18	elibrary.buse.ac.zw:8080 Internet Source	<1 %
19	Submitted to Glasgow Caledonian University Student Paper	<1 %
20	Yuchen Huang, Jingwen Liu, Xuanyi Xiang, Pan Wen, Shiyuan Wen, Yanru Chen, Liangyin Chen, Yuanyuan Zhang. "Malware Identification Method in Industrial Control Systems Based on Opcode2vec and CVAE-GAN", Sensors, 2024 Publication	<1 %
21	Submitted to Addis Ababa University Student Paper	<1 %
22	Thanh-Hung Vo, Guee-Sang Lee, Hyung-Jeong Yang, Soo-Hyung Kim. "Pyramid With Super Resolution for In-the-Wild Facial Expression Recognition", IEEE Access, 2020 Publication	<1 %
23	link.springer.com Internet Source	<1 %
24	bura.brunel.ac.uk Internet Source	<1 %

25	paperswithcode.com Internet Source	<1 %
26	Zia Ullah, Muhammad Ismail Mohmand, Sadaqat ur Rehman, Muhammad Zubair, Maha Driss, Wadii Boulila, Rayan Sheikh, Ibrahim Alwawi. "Emotion Recognition from Occluded Facial Images Using Deep Ensemble Model", Computers, Materials & Continua, 2022 Publication	<1 %
27	formative.jmir.org Internet Source	<1 %
28	www.aimodels.fyi Internet Source	<1 %
29	Submitted to Griffith College Dublin Student Paper	<1 %
30	Isha Gupta, Swati Singh, Sheifali Gupta, Soumya Ranjan Nayak. "Classification of Brain Tumours in MRI Images using a Convolutional Neural Network", Current Medical Imaging Formerly Current Medical Imaging Reviews, 2023 Publication	<1 %
31	S. N. Kumar, Sherin Zafar, Eduard Babulak, M. Afshar Alam, Farheen Siddiqui. "Artificial Intelligence in Telemedicine - Processing of	<1 %

Biosignals and Medical Images", CRC Press,
2023
Publication

32	pdfslide.net Internet Source	<1 %
33	quarxiv.authorea.com Internet Source	<1 %
34	Submitted to The University of the West of Scotland Student Paper	<1 %
35	data.mendeley.com Internet Source	<1 %
36	www.ijisae.org Internet Source	<1 %
37	Elena Paya Bosch. "Deep Learning for the Automation of Embryo Selection in an In Vitro Fertilization Laboratory", Universitat Politecnica de Valencia, 2024 Publication	<1 %
38	rosap.ntl.bts.gov Internet Source	<1 %
39	www.cse.cuhk.edu.hk Internet Source	<1 %
40	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1 %

41	iieta.org Internet Source	<1 %
42	www.scribd.com Internet Source	<1 %
43	Ashraf Darwish, Dalia Ezzat, Aboul Ella Hassanien. "An optimized model based on convolutional neural networks and orthogonal learning particle swarm optimization algorithm for plant diseases diagnosis", Swarm and Evolutionary Computation, 2020 Publication	<1 %
44	Bozkurt, Alican. "Deep Representation Learning for Complex Medical Images.", Northeastern University, 2020 Publication	<1 %
45	arxiv.org Internet Source	<1 %
46	digitalcommons.njit.edu Internet Source	<1 %
47	dspace.bracu.ac.bd Internet Source	<1 %
48	ebuah.uah.es Internet Source	<1 %
49	escholarship.org Internet Source	<1 %

50	idus.us.es Internet Source	<1 %
51	ijsrcseit.com Internet Source	<1 %
52	pure.tue.nl Internet Source	<1 %
53	uwspace.uwaterloo.ca Internet Source	<1 %
54	www.sensorsportal.com Internet Source	<1 %
55	Abi, Beza Alemu. "Web SQL Injection Attack Detection Algorithm Using Deep Learning", Marymount University, 2024 Publication	<1 %
56	Aboubacar Sidiki Magassouba, Abdourahmane Diallo, Armel Nkurunziza, Ali Issakou Malam Tchole et al. "Implementing a Machine Learning Model to Predict Continuation of Contraception Among Women Aged 15-49: Secondary Analysis of the Last 4 Demographic and Health Surveys in West African Country", SocArXiv, 2024 Publication	<1 %

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1




**FACULTY OF INFORMATION AND COMMUNICATION
TECHNOLOGY**

Full Name(s) of Candidate(s)	Koi Chin Chong
ID Number(s)	20ACB05613
Programme / Course	Bachelor of Computer Science (Honours)
Title of Final Year Project	Deep Learning for Image Classification

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: <u>8</u> % Similarity by source Internet Sources: <u>6</u> % Publications: <u>5</u> % Student Papers: : <u>3</u> %	
Number of individual sources listed of more than 3% similarity: <u>0</u>	
Parameters of originality required and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.


 Signature of Supervisor

Name: Dr Zanariah binti Zainudin

Date: 10/09/2024

 Signature of Co-Supervisor

Name:

Date:



UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)


CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	20ACB05613
Student Name	Koi Chin Chong
Supervisor Name	Cik Zanariah Binti Zainudin

TICK (√)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.



 (Signature of Student)
 Date: 06/09/2024