Use of Transfer Learning on Facial Emotion Recognition

Muhammad Ali

3059828

Submitted in partial fulfillment for the degree of

Master of Science in Big Data Management & Analytics

Griffith College Dublin
June 2023

Under the supervision of Aqeel Kazmi

Disclaimer

I hereby certify that this material, which I now submit for assessment on the program of study leading to the Degree of Master of Science in Big Data Management & Analytics at Griffith College Dublin, is entirely my work and has not been submitted for assessment for an academic purpose at this or any other academic institution other than in partial fulfilment of the requirements of that stated above.

Signed: __Muhammad Ali_____ Date: __June 2023____

Acknowledgments

I would like to express my deepest gratitude to my thesis advisor Prof. Aqeel Kazmi, for his guidance, support, and patience throughout the entire process. His invaluable feedback and insightful suggestions helped me navigate the complexities of this research project. I would also like to thank the Computer Science and Data Analytics Department faculty members, for providing me with a rigorous academic environment that helped me grow both academically and personally.

I would like to acknowledge my family and friends, for their unwavering support and encouragement during this journey. Their love and belief in me gave me the strength and motivation to persevere. I would also like to extend my appreciation to the official data providers used in this research project. Their willingness to share their data and insights made this project possible.

Finally, I would like to express my sincere gratitude to all those who have played a role, however small, in the completion of this thesis.

Thank you all.

Table of Contents

List of Equations

List of Figures

List of Tables

Abstract

Facial emotion recognition is the ability of technology to predict emotions based on the facial expressions of people. There is a wide range of applications of FER including education, healthcare, interrogations, autonomous vehicles checking the emotional states of drivers, phones, and social media applications using FER to develop face filters, etc. Deep convolution neural networks have shown prominent results in image processing, image classification, object classification, and facial detection tasks.

While CNNs have also been used for FER, training models from scratch requires a lot of research, time, and computational resources, and the same tasks can be performed relatively quickly using pre-trained models that have already been trained on similar tasks and data. This study aims to use the CNN model and state-of-the-art pre-trained deep CNN architectures for Facial emotion recognition and compare their performance. Two datasets were collected for this study including Karolinska Directed Emotional Faces and Japanese Female Facial Expressions. Open AI's DALL-E 2 was used to generate additional images for training the models. All datasets were mixed to create a master dataset out of which 100 images for each emotion class were separated for validation as well as testing dataset. Giving a total of 700 images for each set and 4,485 images for the training dataset. One custom-built CNN and two state-of-the-art pre-trained models were selected including Xception and Inception V3 were fine-tuned on six different optimizers including SGD, ADA Delta, ADA Grade, NADAM, ADAM, and ADA Max. The training was conducted for 120 EPOCHS, with image sizes of 128 x 128 pixels and a batch size of 64.

Xception showed the highest performance of 91.86% with Inception V3 at 90.43% on testing data proving the high performance of pre-trained models over the custom-built model where custom CNN achieved 86.29% accuracy on testing data. The highest-performing model was used to develop an app that can predict emotions from an uploaded facial image. Future work can be performed on fine-tuning models on different batch sizes, image sizes, and epochs.

Chapter 1. Introduction

This chapter briefly introduces this research project's topic and is divided into four subsections. The first subsection briefly introduces the definition of Facial Emotion Recognition, its applications, and challenges, and touches upon transfer learning. The second subsection briefly explains the objectives of this study. The third subsection provides a general overview of the approaches used to carry out this project and the last subsection provides a guideline of the structure of this document and what to expect in the coming chapters.

1.1 Facial Emotion Recognition and Transfer Learning

Facial Emotion recognition is the ability of machine learning models to detect human emotions from facial expressions. An article published in Tech Dispatch defines it as:

> "Facial Emotion Recognition (FER) is the technology that analyses facial expressions from both static images and videos to reveal information on one's emotional state." (Tech Dispatch) [1]

Facial emotion recognition (FER) is a field of computer vision focused on improving Computer-Human Interaction [2]. FER has a wide range of applications in industries like robotics and education for computer vision [2], self-driving cars where the cars can detect the sleepiness of drivers [3], medical, interactive gaming, and public security [4] as well as marketing & advertisements [5]. One of the major applications includes Computer-Human interaction which has seen rapid developments in recent years [6]. The way computers interact with humans calls for the need for a greater understanding of humans and facial detection systems which are currently in use for face authentication systems, facial clustering, and video conferencing [7] as well as social media applications, healthcare, employment, and crime detection [1].

There are several challenges involved in the field of FER. These challenges include the lack of high-quality datasets [6], difficulty in pre-processing noisy data in publicly available datasets [8], achieving higher accuracies [9], small image sizes in the datasets

to promote online storage and sharing [10] as well as the ethical issues regarding privacy of participants in the datasets. Accuracy is the biggest challenge and goal in the field of FER due to the obvious and delicate nature of its use in the applications mentioned above. This document will expand more on these challenges and more in detail in the coming chapters and how they are handled in this study.

Transfer Learning is a machine learning technique in which knowledge learned from training on previous problems in a similar domain can be used to train models for new problems in a similar domain [10]. This makes it easy to train models that were already trained on similar problems and datasets instead of making a new model from scratch and training it. Moreover, this helps in the case of fewer computational resources as there are models that have already been trained on large-scale datasets that require large-scale computational resources but during transfer learning those models just need to be loaded and fine-tuned to the problem at hand.

This is just a brief introduction to the topic and in the coming chapters, this paper will expand more on transfer learning approaches used for facial emotion recognition during this project. The next subsection briefly explains the goals and objectives of this study.

1.2 Goals

This research focuses on the review of the vast literature in the field of FER to identify best practices in the field and use that knowledge on improving the overall accuracy of the models used in transfer learning for facial emotion recognition. Furthermore, this research aims to test the effectiveness of transfer learning approaches against traditional machine learning approaches. This research also aims to provide practical recommendations and guidelines for using transfer learning in facial emotion recognition.

Based on the objectives above, this project aims to answer the following research questions:

- What best practices are available in the field of using transfer learning for FER?

- How do transfer learning approaches perform in the field of FER?
- How do transfer learning approaches perform against traditional machine learning approaches in the field?
- What are the practical implications of using transfer learning in facial emotion recognition for real-world applications?

Answers to the above questions will be beneficial for the industries that use facial emotion recognition technologies including marketing & advertising, computer vision, autonomous cars, healthcare providers, biometric technology services as well as phone applications that use facial filters. Moreover, it will be beneficial for the researchers in the field of computer vision, robotics, and Computer Human interaction to carry out future studies as the results of this study can guide them on what to do and what not to do.

The coming subsection provides a brief overview of the approach used to carry out this study.

1.3 Overview of Approach

Following steps highlight the overview of the approach used to conduct this study:

- Data Collection from official dataset providers.
- Data Creation using Open AI DALL-E 2 for balancing datasets.
- Data Cleaning to discard noisy images that don't have faces.
- Face Detection using Multitask Convolution Neural Networks MTCNN.
- Crop Faces using Open CV.
- Scale images to centralize through padding.
- Grayscale conversion.
- Scale normalization to equalize brightness to contrast ratio.
- Data augmentation using Tensor Flow.
- Preparing batches.
- Training models using TensorFlow and Keras.
- Fine-tuning models

- Testing models
- Evaluations

This is a high-level view of the approach used for the study and these steps will be explained in detail in the coming chapters. The next subsection briefly explains the document structure and what to expect in the coming chapters.

1.4 Document Structure

The rest of this document is as follows. Chapter Two provides a literature review of the area of "Using transfer learning approaches for facial emotion recognition" and the sources consulted in accomplishing this project, in addition to related work. Chapter Three describes the methodology and high–level design of the project structure. Chapter Four discusses the system design, requirements, and specifications including any hardware and software used. Chapter Five provides the implementation details of this project. In Chapter Six, details of the working prototype of this project are provided, including testing and evaluation techniques used. Results are also discussed including any revisions to the overall design and implementation that were deemed necessary. Finally, Chapter Seven presents a conclusion and guidelines for future work.

Chapter 2. Background

This chapter includes a literature review of the research done in the field of facial emotion recognition with the help of machine learning. This chapter includes two subsections. The first subsection expands on the literature in the field while the second chapter reviews the literature that is highly related to this project.

2.1 Literature Review

Emotions play an important role in everyday communication between humans and have a direct impact on our decision-making abilities and quality of life [11] and can easily be seen and expressed through our faces. Hence, facial expressions also play an important role in modern-day science on the topics of emotions, communication, psychology, and how they impact us [3]. There is a growing interest in the field of technology to detect human emotions through facial expressions [4]. The so-called field of Facial Emotion Recognition focuses on the use of technology and machine learning models to analyse expressions of the face to predict human emotions [1].

Data mining is a process based on mathematics and statistics used to discover hidden patterns and knowledge from large datasets based on a variety of techniques [12]. These techniques include data integration, data selection, data cleaning, data transformation, data mining, evaluation, and knowledge discovery [13]. It involves the use of descriptive and predictive models where descriptive models are used to describe past behaviours based on data and predictive models are used to predict future behaviours based on data [12]. These models can perform association by linking similar items together, classifying the types of data, regression, and clustering to group items together based on their distance [13].

Machine learning is a field of data mining in which computer algorithms learn knowledge gained from data sources fed to them and perform tasks based on that knowledge [14]. Machine learning has become popular in a variety of domains in recent years due to innovations in technology and the availability of data [13]. From customer

relationship management, risk management, technology, business, and human-centric aspects of the businesses to the applications of predicting disease in health care, sports performance and more, machine learning has been applied with great results to achieve tasks that were not possible before [15].

Machine learning can be used to perform a variety of predictive tasks including clustering, association, classification, or regression and each has its models [13]. The choice of models depends on the type of task that needs to be accomplished, for example, decision trees and random forests are used for classification purposes where a model learns the attributes of a class on training data and using those attributes it can predict the class that is not known [14]. A neural network is one such model that has gained prominence in the field of image analysis to extract useful features from the edges of the images and classify objects based on the information extracted from the images [3].

Convolution Neural Network is a type of neural network that has been widely used in machine learning in the field of FER because they are specifically designed to extract useful features from the pixels of images [3]. CNNs have been used in the field of image recognition, video recognition, facial expression recognition, natural language processing, and artificial intelligence [3] [10] [2].

A CNN includes three types of layers namely the convolution layer, the pooling layer, and the fully connected layer [2]. The convolution layer is the most important layer of a CNN because it learns the spatial relationship between pixels of an image by performing dot product of the image and kernel matrices and preserves that feature map [4]. Pooling layer is used to reduce the dimensionality of the extracted features without compromising the information by combining nonoverlapping areas of the feature map and reducing them into one single value [10] [2]. The fully connected layer has all the neurons of previous layers connected to all the neurons of the output layer and is used as a classifier to give an output from the previous connected layers [2].

A combination of convolution and pooling layers with some regularisation layers like dropout and batch normalization are attached to form a block and several such blocks

are stacked on top of each other to form a fully-fledged convolution neural network [2]. These CNNs are then trained on data and customized based on experimentation, research, and development to enhance their performance on the task at hand.

Transfer learning is another type of deep learning where models that are already trained on large datasets are used where such models already learned useful features from the training datasets and this learning can be employed in a very similar task without creating the models from scratch [10]. This has many advantages including leveraging pre-learned features that models learned in previous training, faster training and convergence since the models have pre-trained weights loaded so they start from already optimized features, the ability of such models to generalize and regularise better since they have already been trained on very large and diverse datasets which also helps in reducing the over-fitting problem. [7].

Several highly related studies have been conducted on facial emotion recognition using CNN and transfer learning techniques that will be discussed in the next section.

2.2 Related Work

This subsection reviews and examines related recent works and studies on FER using Convolutional Neural Networks (CNNs) and related techniques. CNNs are a type of neural network commonly used for image-processing tasks, including facial emotion recognition. This review will provide an overview of CNNs and highlight related research in the field by critically evaluating and comparing the works with each other.

Available datasets, sizes, and quality:

The data plays an important role in training convolution neural networks to produce high accuracy and is one of the great challenges in facial emotion recognition due to the lack of good datasets that have quality and quantity [6].

Several studies have been conducted on FER 2013 dataset [9] [2] [16] [7] because of its vastness with 30,000 images on seven emotions namely Anger, Disgust, Sadness, Happiness, Surprise, Neutral, and Fear [10]. Other studies [10] [11] have used the

Japanese Female Facial Expressions JAFFE dataset and have achieved very high accuracies of more than +95% because of the smaller size of the dataset as the dataset contains only 217 images of a few Japanese female models and the images are taken from the front view of the face that makes it easy to classify emotions and is hard to do in real-world problems [5].

AffectNet is another dataset used in a study [17] where the researchers achieved relatively high accuracy scores of 51% for the dataset as the dataset contains more than a million images of people with different emotions and the study only used 300,000 images which is a huge amount and requires a lot of computational resources for processing and training of models. CK+ is another dataset based on short video clips showing human emotions and is used by several studies [7] [5] [10] where researchers have achieved around 85% accuracies but the dataset is not publicly available to everyone and needs approval from the Cohn Kanade research institute.

Karolina Directed Emotional Faces KDEF is another dataset that contains 4900 images of 70 models having seven emotions that were used in some studies [10] [3] with some prominent accuracy scores of 93.7% achieved. The dataset is publicly available and has a descent size with images taken from different angles for each subject.
Reviewing all the datasets mentioned above, FER 2013 and KDEF datasets were chosen for this study for easy availability, quantity, and diversity of the images.

Miss Classified Emotions & Irrelevant Images:
The downside of using the FER 2013 dataset is that it has so much miss-classified data of images where an image classified as one emotion is actually for another emotion Kim et al [6] conducted a study where they developed a framework to clean the dataset not only from irrelevant facial images but also to classify images properly using Facial Image Threshing machine.

The algorithm proposed by Kim et al [6] can collect a huge number of images from videos, eliminate any irrelevant images from the FER2013 dataset, correct misplaced or misclassified facial images as well as remove the background and leave the face on the image. The proposed algorithm takes videos and image datasets and converts them

into raw images, then it extracts the face using MTCNN, extracted facial images are resized, and then a data segregator based on the Xception model was used to classify images based on the detected emotions.

The proposed system was evaluated by running tests on three datasets including FER2013, CK+, and iSPL. CK+ and iSPL datasets had a 70% to 30% train-to-test split while the FER2013 dataset's default split was used. The models used in the evaluation were Simple CNN, PyFER, MobileNet, Inception V1.0, ResNet50, and Xception. All the models were trained up to 150 epochs.

The results were evaluated using precision, recall, and F1 scores. The results were evaluated by measuring these scores for each model without using the FIT machine and comparing those scores with the scores gained later using the FIT machine. All performance metrics resulted in higher scores when the FIT machine was used suggesting the success of the proposed pre-processing algorithm. The proposed facial image threshing machine can be used to pre-process facial image data and generate new facial images from videos which can be very useful to make new datasets. This study also uses this threshing machine to clean the dataset from irrelevant images.

Image Sizes:

Another challenge of the FER 2013 dataset is the small size of 48 x 48 pixels of the images which is due to the high number of images in the dataset and the need to store the dataset online for sharing purposes so the image size was reduced to get smaller size for small online storage and sharing [9]. During the fine-tuning of a variety of elements and parameters, the study conducted by Akhand et al [10] tested the accuracy scores by having all parameters the same but only changing the image sizes. The sizes that were tested in the study included 48x48, 64x64, 128x128, 224x224, and 360x360. The results showed that all models had a higher accuracy score on 128x128 image sizes.

This challenge of small image sizes in the FER 2013 dataset was solved by a study conducted by Ali et al [18] where they used a CGAN, a type of Generative Adversarial Network that can superimpose the resolution of an image by adding pixels based on predicting colours from the surrounding pixels. A similar model was designed by

Xintao et al [19] who won the first prize in the Region 3 competition for such GANs with the best perceptual index that is considered as the evaluation metrics of GANs and they made their work public and open source which is available on GitHub for free use. This study used ESRGAN proposed by Xintao et al [19] to perform pre-processing of images where images needed super-resolution to improve the quality and size of the images.

Face Detection & Crops:

One of the pre-processing steps conducted in several studies [2] [20] [16] in facial emotion recognition is to detect faces in the images and crop them. This leaves the extra image and background out of the image and only focuses on the faces to produce input for CNNs. HAAR Cascade classifier is a face detection model that can be used in the OpenCV library [16] .

Another study [6] used multitask cascade neural network MTCNN which has better performance as compared to the HAAR cascade classifier and is used in modern applications for face detection. The study used the MTCNN face detector in the Facial Image Threshing machine that is used to detect faces from the images to make datasets for facial emotion recognition projects in the future.

This study uses the same MTCNN model to detect faces and clear out images where faces are not detected to clean the dataset from images that do not have faces. OpenCV is used to crop the faces from images and generate new images that only have faces and everything else including other body parts and background were removed from the image.

Colour to Light Ratios:

Zhang et al [4] conducted a study on face emotion recognition using image edge computing in which they superimpose the preserved image edges extracted from convolution neural networks and pass them to maximum pooling layers. The FER2013 and Labelled face in the wild (LFW) datasets were mixed in this study to generate a dataset with mixed and complex backgrounds.

The study uses HAAR face detection to detect faces from the images. Once faces are extracted, the image scale is normalized to 128 x 128 pixels. Moreover, the images go through a greyscale equalization process to streamline the contrast-to-light ratio so the faces that are darker because of light can become a little brighter which can make it easy for image edge extraction.

The CNN model has 6 hidden layers with two convolution layers, one pooling layer, one convolution layer, and one pooling layer followed by one fully connected layer which feeds forward to the SoftMax output layer. The proposed model has a recognition rate of 88.56% which is compared with an R-CNN model with a core of 79.34%. It is not mentioned what performance metrics were used in the study to evaluate results which are referred to as recognition rate.

Use of Filters:

Zadeh et al [11] conducted a study in which they proposed the application of Gabor filters on original images to make the edges of the face, nose, eyes, and lips darker while the other facial surface lighter. This way Gabor filters make the edges more visible where the texture in the image changes.

The proposed mythology applies Gabor filters twice on an image to make the edges highly visible. Once the processing is done, the images are passed through deep neural networks to extract facial features which are then classified for emotions.
The CNN architecture had three convolution layers with max pooling and afterward, the outputs were flattened, two dense fully connected layers were used which then outputs to a SoftMax layer.

The dataset used for evaluation was JAFFE which contains 213 Japanese female models depicting seven emotions. The model was trained up to 30 epochs and the study aimed to increase the training speed and accuracy of the model.

The evaluation was done by training the model without Gabor filters and comparing the results with the metrics after running the model with Gabor filters. The evaluations

show that the use of Gabor filters increased the accuracy of training from 82% to 91% in 30 epochs which suggests an increase in the learning speed.

Types of Models:

One of the recent studies [3] designed a new convolution architecture namely venturi architecture. They used this architecture in comparison with two already existing architectures to test which performs better in terms of accuracy scores.

The study used Karolinska Directed Emotional Faces (KDEF) dataset images which were scaled down to 256 x 256 pixels. The training was done up to 25 epochs.
The three architectures were named "Rectangular" architecture, "Modified Triangular" architecture, and "Venturi" architecture which were named after the shape of layers within each architecture. All these architectures were trained on the same dataset with the same training and test split.
The training accuracy of rectangular, modified triangular, and venturi architectures was 98.64%, 98.29%, and 98.87% respectively. The validity accuracy of the same models in the same order was 79.61%, 82.70%, and 86.78%. The results showed that the proposed venturi model had a higher testing accuracy compared to the other two approaches.

The venturi architecture uses 6 hidden layers and an output layer with 7 nodes for each emotion class. The hidden layers architecture consists of 256, 192, and 128 nodes for the first three consecutive layers and then the remaining three layers mirror the first three starting from 128, 192, and 256 forming a shape of a venturi.

Another recent study [2] used convolution neural networks to detect the emotions of students using facial features. The study used the FER2013 dataset for training the convolution neural networks model. The training and validation were done at an 80% to 20% ratio and the model was trained up to 106 epochs. The results showed a validation accuracy of 70% and upon reviewing the confusion matrix it shows that the model predicts happy and surprise classes well. The emotions of sadness and fear were mostly confused by the model a lot. The study also incorporates the model to detect emotions on live streams from the camera.

Shaees et al [5] conducted a study in which they compared handcrafted approaches with transfer learning approaches to test which ones perform better. The study used AlexNet which is one of the best pre-trained CNN models.

The study uses two separate approaches and compares them with past studies on accuracy scores. The first approach is simple transfer learning where AlexNet CNN is used. This CNN is trained on hundreds of thousands of images of different things and can classify things with very high accuracy. The initial layers of the model are taken, and the final layers are added to form a transfer learning model.

A second approach is a hybrid approach in which the features extracted from the transfer learning approach are passed on to an SVM Support Vector Machine model that classifies the emotions based on the input data.
The datasets used in the study are Natural Visible and Infrared Expression Database (NVIE) and Cohn-Kanade (CK+) and the accuracy scores for hybrid and transfer learning approaches are 99.3%, 91.5%, and 98.3%, 90.1% respectively. These results show how incorporating transfer learning approaches into handcrafted models can drastically improve accuracy scores.

One study [16] trained convolution neural networks to detect emotions from facial expressions in real-time using OpenCV. The methodology had three parts namely face detection, facial feature extraction, and emotion classification. The dataset used to train the model was FER2013. The dataset contains images for seven emotions namely anger, disgust, happiness, sadness, fear, surprise, and contempt.

Facial detection was implemented using HAAR cascade which uses the KNN algorithm to detect faces. The model was a mini version of Xception which is a CNN-based model. The training and test data was split with a ratio of 80% to 20%. The validation accuracy scores came out to be 65.97% which is a good score for the FER2013 dataset.

The study also tested the model on real-time video streams where the confusion matrix shows a promising result, but the number of test subjects was very little as 10 people

for real-time testing. The emotion of disgust was often confused with the emotion of anger because mostly both emotions elicit similar facial expressions in most people. Also, the number of training images for disgust is very low in the dataset compared to other emotions which also causes an imbalance in the dataset and the training process. Khaireddin et al [9] conducted a study on the FER2013 dataset which is considered a benchmark for studies in the field. The study aimed to adopt the VGGNet model and tested a variety of fine-tuned hyper-parameters and various optimizations to improve the accuracy scores since the dataset normally yields lower accuracy scores because of the imbalance and requirement of more data cleaning and procedures.

The study applied various data augmentations including ± 20% rescaling of images, horizontal and vertical shifting, and ± 10% rotations randomly on images with a 50% probability. The training was conducted up to 300 epochs. All available optimizers in Keras and TensorFlow were tested. The results used accuracy scores from all high-performing models and compared them with the model used in the study. The accuracy scores of the proposed model yielded the highest scores of 73.28% which is way higher than the Kaggle competition scores held on the same dataset.

Alreshidi et al [20] used a completely different approach in their study of emotion detection from facial expressions. In their study, they used the AdaBoost cascade classifier to detect human faces from images. Once faces were detected, the study used Neighbourhood Difference Features (NDF) to extract facial features. Based on those features, they trained a Random Forest Classifier that will take those NDF features and classify emotions based on those.

The emotion classes were anger, disgust, fear, happiness, neutral, surprise, and sad. The study also introduced a latent emotion. When the faces were detected correctly from the images, the model would classify them from the seven emotions, but the latent emotion was used to classify the image if the face was not detected properly from the images.

The datasets used in the study were static facial expressions in the wild (SFEW) and real-world affective faces (RAF). The performance metric used was average accuracy

which measures the accuracy of each emotion detected and then averages all the scores. The proposed method achieved an average accuracy score of 57.7% for SFEW and 59.0% for RAF datasets. These scores were 18% and 24% higher than the other studies conducted similarly.

Mishra et al [7] used ResNet50 which has 50 layers as the base model to conduct a study for facial emotion recognition with deep residual learning. The datasets used in the study were FER2013 and CK+ and the emotion classes were anger, fear, disgust, happiness, sadness, neutral, and surprise.

The model was trained up to 100 epochs. The study tested the learning rate from 0.1 to 0.0001 and concluded the best learning rate at 0.001. The optimal batch size was 64. The 5-fold cross-validation results showed an average accuracy of 81.94% for the FER2013 dataset which is better than any previously achieved results.

Siddiqui et al [17] propose a framework to compare models developed for facial emotion recognition in a standard way. They developed a model based on CNN that was trained on the AffectNet dataset which is one of the largest datasets available on the internet for facial emotion recognition. They also developed a web application where anyone can upload an image and the application classifies the emotion depicted in the image.

The dataset used includes more than a million images, but the study used 300,000 images which were manually annotated to represent eight emotions that include neutral, sad, happy, disgust, anger, surprise, fear, and contempt. The image sizes were 224 x 224 pixels.

The training and test split came out with 287,651 training images and 4,000 testing images. A small variant of AlexNet was used to train on the dataset for faster computation because of the large size of the dataset. The model used an Adam optimizer with recommended parameters.

The RGB values of images were normalized to be between 0 and 1. Five convolution blocks were used in the model with maximum pooling layers. Afterward, the output was flattened, and two blocks of dense fully connected layers were used before the output layer.

The evaluation metrics used were accuracy scores and the results showed 55.09% accuracy which is ranked number fourth on the AffectNet dataset with 63.50% highest and 58.00% second and third highest in the list. Based on that, the proposed model had better accuracy compared to most of the models reviewed in the study.

The study also proposed a web application framework that can take a facial image as input and classify its emotions. Moreover, the web application is used to save all uploaded images in the database to generate high-quality and high-quantity datasets which are essential in the field of facial emotion recognition.

One study [10] argues that the straightforward use of simple CNN models in the field of facial emotion recognition is not practically applicable since they use facial images with front poses only to detect faces. The study also argues that deep CNN should be used with transfer learning approaches to improve the performance of the algorithms and to overcome the face position challenges discussed earlier.

The datasets used in the study were KDEF and JAFFE. The images were pre-processed and resized to 224 x 224 pixels and the Adam optimizer was used with a learning rate of 0.005. The images were also augmented using a rotation factor set to +- 10 degrees at random. The train-to-test split was 90% to 10% for both datasets. The CNN model was tested with different sized of the images as inputs and the results show that 128 x 128 pixels size had the highest accuracy scores for both data sets.

For further fine-tuning, the model had four different versions including the entire model made from scratch, the model that used dense layers plus VGG16 as the base model, the model with dense layers and a fifth block of VGG16, and finally a model with dense layers only.

The results showed that the model made from scratch had the lowest accuracy scores while the model with VGG16 used as a base with a few dense layers performed highest with a 100% accuracy score for JAFFE and 93.47% accuracy scores for the KDEF dataset.
The study also conducted a test of pre-trained models that included VGG-16, VGG19, ResNet-18, ResNet-34, ResNet-50, ResNet-152, Inception-v3, and DenseNet-161. The 10-fold cross-validation scores were used to evaluate the results.

Then the study used the proposed method to incorporate transfer learning from each of the above models. The results showed the use of DenseNet-161 used for transfer learning on 10-fold cross-validation yielded the highest accuracy scores of 96.51% for KDEF and 100% for the JAFFE dataset. This also suggests why the use of deep learning and transfer learning together can achieve high accuracies which are reliable for practical use.

Based on the review of the literature, KDEF [10] [3] and JAFFE [10] [11] are high-quality publicly available datasets that can be used for FER as they are small but the image sizes are better. Xception [6] and Inception V3 [10] have seen some prominent results in FER which can also be used for transfer learning purposes. The optimal image sizes can be 128 x 128 [10] and most of the studies did training or more than 100 EPOCHs. Using this information, we can define our methodology in the next chapter.

Chapter 3. Methodology

This chapter outlines the steps and procedures undertaken in this project to investigate the problem of facial expression recognition using deep learning models. This chapter provides an overview of the dataset used, the pre-processing steps applied, the architecture of the deep learning models, and the evaluation metrics employed for performance analysis.

3.1 Dataset Selection

This section explains the high-level decisions made in the selection of the datasets and the collection process.

Quality datasets are an important component for training high-performance models as reflected in the literature review. The publicly available dataset for FER is very limited in quality and quantity. Most of the datasets are very large ranging from 30,000 images to more than 300,000 images and for ease of sharing datasets with the public, the image sizes of these datasets are compressed to make the size of datasets small.

This is problematic because the small images have limited pixels and image quality has a direct impact on the performance of the datasets. When small images are rescaled to a larger size, the pixels of the image get distorted leading to loss of valuable information. So, having datasets with high-resolution quality and quantity of images was the ideal selection criteria for the datasets. Using this criterion, two datasets named Karolinska Directed Emotional Faces (KDEF) and Japanese Female Facial Expressions (JAFFE) were selected for the study. Both datasets are publicly available and were collected from their official websites which required an agreement to not disclose datasets out of the research scope.

This section only includes the high-level decisions made to collect the datasets; the datasets will be further explained in the implementation chapter.

3.2 Dataset Creation

The selected datasets KDEF and JAFFE together had 5123 images in total, but the KDEF dataset had a lot of images that were not going to be included after pre-processing as they were not related to the study. This would drastically decrease the size of the dataset leading to an inadequate quantity for training models. In such circumstances, data can be generated using a variety of techniques. If the datasets are limited in quantity or the classes in the datasets are not balanced, the data can be created to increase the quantity of the dataset.

Several AI-based generative models can generate images based on textual prompts but most of these services have certain niches and themes of the type of images they generate. These models can generate images in themes of animation, fantasy, imaginative, divine, and realistic. DALL-E 2 is a generative model developed by Open AI that can generate realistic images from textual prompts in bulk using their API. This model was used to generate an extra 2700 images based on textual prompts that were realistic and could be generated in bulk using Python. The details of how this was accomplished are explained in the implementation chapter.

3.3 Data Pre-Processing

Once the datasets are obtained, they are pre-processed to prepare and structure them in a way that machine learning models can train on. This section explains the high-level decisions made for pre-processing steps taken for the datasets.

3.3.1 Face Detection, Crops, and Cleaning

The images in datasets may contain more than what is needed for training the models including other body parts like the neck and shoulder. Some images may not be necessary for training that include images that do not have faces, animated character images, or images taken from side poses that can be used in other face and body detection studies but when it comes to facial expressions, this study only needed images that show a full face of an individual with their facial expressions excluding side poses.

For this purpose, the first pre-processing step needed to be detecting faces from the images and cropping them.

This step would remove the parts of the image that are not necessary for training the models such as neck and shoulder and it would also help in removing images that do not have realistic faces such as blank images or animated characters. Haar cascade face detector is a feature in the OpenCV library in Python that can detect faces. MTCNN is another face detector based on deep neural networks that work wonders in a different light and pose settings. MTCNN is sensitive compared to the Haar cascade and was used for the data cleaning step while the Haar cascade is faster and was used in the application development in the later stage. These are explained in more detail in the implementation chapter.

3.3.2 Rescaling Images

After the faces are cropped, the dimensions of the newly cropped face image may be different for all the images. Training deep CNN models require images to be of uniform dimension and size which deemed this step appropriate to make cropped facial images of uniform dimensions. The aspect ratio of the cropped image faces was checked to see if the height of the image is greater than the width, a width-wise padding was added on both sides of the image to make the width equal to the height. The same was done in case the width was greater than the height of the image, the height-wise padding was added on both vertical sides of the image. Adding the padding made the images come in the centre with a white padding around and made them of uniform dimensions which is a requirement for training deep CNN models.

3.3.3 Grayscale Conversion

Images from the JAFFE dataset was grayscale while the images from the KDEF dataset and Open AI images were in colour. This can confuse the training models and the images needed to be in a uniform colour scheme. This step was taken to convert all images into grayscale to ensure consistency in the training images. OpenCV library was used to convert images to greyscale as this library is used for all image processing steps.

3.3.4 Normalising Brightness and Contrast

After the images are converted to grayscale, the brightness and contrast of the images may vary with some images having very high brightness and some having very high contrast. This can also be confusing for the neural networks and this step was needed to normalize the brightness-to-contrast ratio of all the images to ensure further consistency in the training images. OpenCV has a histogram normalization function that makes it easy to normalize the histograms of the images into uniform distribution maps.

3.3.5 Post-Process Cleaning

Once all the pre-processing is done on the datasets, a post-processing step is taken to ensure that all the pre-processing is done properly. Some images might be over-processed during pre-processing and might not be needed for training the models. This step is required to ensure that all the pre-processing has been performed according to the requirements and that the data obtained after pre-processing has been cleaned properly and does not include any undesired or over-processed images. This step was performed manually for a subjective evaluation of the pre-processed data.

Once this subjective evaluation is over, the dataset is ready to be prepared for training. For this, it is split into training, validation, and testing sets and a further data augmentation step can be taken the details of which will follow in the next section.

3.3.6 Training, Validation, and Testing Splits

The master dataset is split into three sets, each containing seven class folders. This splitting of the dataset is essential to check the generalizing ability of the trained model. While training, the model learns the features of images in the training data and can easily predict the class if the same image is presented to the trained model, but a good, trained model should be able to predict images that the model has not seen before, and this ability defines generalization ability of the model.

If the model cannot generalize well on the new unseen data, it can be overfitting and will not be useful in real-world applications. To test this generalizing ability, the master dataset is split into three different sets. The training set is used to train the model where the model learns all the features from the training examples. A validation set is used to measure how well the model is performing on unseen data during the tuning of hyperparameters for re-training if necessary. Once the model has been trained using optimal hyper parameter tuning. It can be tested against the testing dataset to see how well it performs on new unseen instances. This helps evaluating models more realistically.

For this project, the model was split into training, validation, and testing datasets where 100 images for each class were randomly chosen from the master dataset and moved to these sets which give us 700 images for validation and 700 images for testing the datasets. The remaining images were kept for the training dataset.

3.3.7 Data Augmentation

Data augmentation techniques are performed in cases when the training data is limited but training requires more data. Some of these techniques include flipping the images horizontally, vertically, rotations by small degrees, etc. This provides some variations of the same images that are visually different from each other in small proportions.

Since most of the images were taken for testing and validation sets, the training set was left with a very small number of images that might not be adequate for training the model. For this reason, data augmentation was performed only on the training set to tackle the imbalanced data and increase the size of the training data set.

Horizontal flips were deemed appropriate because the human face is symmetrical in design and the left-side expressions may vary a bit from right-side expressions by doing a horizontal flip the model can learn a few examples of such existence of differences in expressions on either side of facial symmetry.

3.4 Model Architectures

Deep Convolution Neural Networks have shown excellent performance in object recognition and image classification tasks because of their ability to extract hierarchical representations from raw data including low-level features like edges and textures as well as high-level semantic information such as facial landmarks and expressions [10]. This study included three DCNN models to be trained. Two pre-trained models were selected based on the top performance reviewed in the literature namely Xception and Inception V3. The third model was built from scratch. All models were compared to answer the research questions of comparison between transfer learning and deep learning approaches. This section explains the architecture of the models and the customizations performed for training on FER.

3.4.1 Inception V3

Inception V3 is a deep CNN architecture developed by Google that features a multi-branch structure with different filter sizes that allows it to capture information at different scales [21]. The model employs a combination of 1x1, 3x3, and 5x5 convolutions to extract features at multiple levels of abstraction and utilizes the concept of bottleneck layers to reduce the computational complexity of the model. Inception V3 has achieved notable success on various image classification tasks, including facial expression recognition [10].

The high performance of the model shows the potential to extract meaningful features from the facial images used in this study and predict emotion classes accurately. Moreover, the model uses bottleneck layers that make the model more efficient in terms of computational resources and parameters that make it an ideal choice for use in cases of less computational resources. This model also has pre-trained weights available on large-scale datasets such as ImageNet which allows leveraging the learned features from vast amounts of data, providing a head start for facial expression recognition tasks and reducing the need for extensive training on limited facial expression datasets.

3.4.2 Xception

Xception is another deep CNN architecture developed by Google which is an extension of the Inception architecture but employs depth-wise separable convolutions, which separate the spatial and channel-wise dimensions allowing for more efficient and parameter-efficient learning [22]. Xception has shown strong performance on various computer vision tasks, including image classification, and has achieved state-of-the-art results on the ImageNet dataset which were slightly better than the Inception V3 model.

The model has also achieved state-of-the-art results on facial emotion recognition tasks [6] which indicates the potential for extracting useful features from the facial images used in this study and predicting emotion classes accurately. Also, the depth-wise separable convolutions used by the model make it more efficient in terms of computational resources and parameters which makes it an ideal model to be used in times of limited computational resources. This model also has pre-trained weights available on large-scale datasets such as ImageNet which allows leveraging the learned features from vast amounts of data, providing a head start for facial expression recognition tasks and reducing the need for extensive training on limited facial expression datasets.

3.4.3 Sequential CNN

Deep CNN architectures are well known for automatically extracting meaningful features from the images during training and have shown prominent results in image classification tasks including object detection, face detection, and facial expression recognition [4]. The architecture consists of convolution layers that consist of multiple filters or kernels that convolve over the input image to create feature maps which are then used to capture visual maps and local structure and patterns in the images [2].

CNNs have been widely used for image classification tasks and studies have been conducted for such tasks where CNNs are designed from scratch and trained on the task at hand [17]. While this is a good approach, using pre-trained models that have already

shown great results can reduce the training and computational cost drastically. This model was developed from scratch to compare the results of pre-trained models with the newly developed models and evaluate the difference in the performance of deep learning and transfer learning approaches.

The selected models are downloaded and customized to suit the task at hand by adding any additional layers or blocks of layers and customizing the output layer for the number of classes to be predicted. Once this is done, the models are ready for training.

3.5 Model Training

This section explains the training process for the deep learning models.

3.5.1 Batch Sizes

The dataset is divided into batches of a specific size which helps process the dataset more efficiently as processing a large dataset can be time-consuming and memory intensive. Dividing the dataset into smaller batches helps the model update its parameters frequently which leads to faster convergence during training. Moreover, deep learning models calculate gradients of the loss function concerning parameters during backpropagation which provides estimated information on how the parameters should be adjusted for minimizing loss and by computing gradients at a batch level provides a good estimate that helps the model adjust the parameters rather than computing the gradient for the whole dataset.

Batches also allow randomness in the training data with each batch having different sets of samples from the dataset which leads to different estimates of gradients and updates accordingly which helps the model's generalization ability as it gets exposed to different patterns. This randomness also helps prevent the model from memorising certain examples, but too few batch sizes can also lead to noise in the data where the model does not learn any of the examples.

The batch sizes are carefully selected based on the computational resources available and the nature of the dataset and can be used as hyperparameters for finetuning models where different batch sizes can be tested for best performance. For this study, the dataset was divided into batches with a batch size of 64 images per batch which has been proven optimal for the studies conducted on FER.

Once the batches are ready, the models are initialized and compiled by defining the loss function and optimizers which are explained next.

3.5.2 Loss Function – Categorical Cross Entropy

For compiling the initialized models, a suitable loss function is selected based on the task at hand. Categorical cross entropy and binary cross entropy are used as loss functions for common classification purposes. The choice of loss function depends on the nature of the problem and the desired output. Binary cross entropy is used in cases of binary classification where only two classes are used as output whereas categorical cross entropy is used in cases where the number of classes for output is more than two.

Categorical cross entropy measures the dissimilarity between the predicted probability distribution and the true probability distribution of the target classes. The goal of this class function is to assign an input example to one of several predefined classes calculating the loss by comparing the predicted class probabilities outputted by the model with the true class labels.

First, the true class labels are one-hot encoded and then the model predicts the probability distribution over the classes for each input example by passing probability distributions over a SoftMax activation function ensuring the predicted probabilities sum up to 1. The following equation is used to calculate the loss.

$$\boldsymbol{L = -\sum_{i=1}^{C} y_i \log[Pi]}$$

Equation 1

Where:

- L is the categorical cross-entropy loss.
- C is the number of classes which is 7 in this case.
- Yi is the true label for class i.
- Pi is the predicted label.

3.5.3 Optimization Algorithm - ADAM

The parameters of a model during the training process are updated based on the optimization algorithms used in deep learning CNNs which minimize the loss function and improve the model's performance in each iteration hence determining how the model learns and adjusts its weights and biases on the gradients computed during backpropagation. There are several optimizing algorithms available but for this study Adaptive Moment Estimation, the ADAM algorithm was used as the default optimizer for all the models.

ADAM optimizer combines the characteristics of ADAGrade and RMSProp optimizers to provide efficient and adaptive updates of the parameters during training by computing adaptive learning rates for each parameter based on beta 1 which is the mean and beta 2 which is the uncentered variance of the gradients which help in adapting the learning rate for each parameter individually by considering the historical values of the gradients.

The learning rate of 0.0001 is used which is a parameter of the optimizer. Other parameters such as beta1 and beta2 were set to default values of 0.9 and 0.999 respectively.

3.5.4 Training Loop & EPOCHS

The training process goes through several iterations where the model makes predictions of the input batch and adjusts the parameters accordingly. Since the dataset is divided into several batches, the process continues to predict classes for each batch and adjust

parameters, and once all the batches are finished that makes 1 EPOCH which means the model has seen the entire dataset once.

In deep learning, the training loop needs to go multiple times so the model can learn from the dataset several times and update the parameters accordingly. This makes it important to choose the number of epochs carefully and it also depends on the complexity of the problem and the size of the dataset. If the number of epochs is too small, the model can be underfitting as it might not have seen the training data enough times to learn from it properly.

For this study, the number of EPOCHS was set to be 120 considering the size of the data and low computational resources. The low-level facial features require several iterations for the model to start differentiating between expressions that are detailed which also contributed to the selection of this number. If the training goes for so long, the model may start overfitting which can be stopped early.

3.5.5 Saving Models – Check Pointing

While the model is learning during the iterations, its performance is tested against a separate validation set against some evaluation metrics like accuracy or loss. These values can be used to monitor the performance of the model during training and the evaluation metrics can be set for monitoring for any improvements. If the evaluation metric improves from earlier epochs the model's parameters and weights can be saved. This helps preserve the best-performing parameters and weights during training as a model may not perform or start to overfit in the upcoming epochs. This process is called checkpointing where a new checkpoint is created, and the model version is saved in a file. For this study, validation accuracy was monitored for checkpointing as this is the goal of this study.

3.6 Fine Tuning

Once the training has been completed, the model is finetuned to improve the performance metrics by tweaking the hyperparameters and experimenting with

different parameter settings. This section explains these hyperparameters and the fine-tuning.

Batch size is an important hyperparameter that can be tweaked to monitor any improvements in the performance of the model. Batches provide variations of data to the model so the model can adjust its parameters accordingly. Larger batch sizes can provide less noise in the data but are also more memory intensive whereas smaller batch sizes save memory but can add in noise. Different batch sizes can be experimented with for optimal model performance.

Optimization algorithms are also important hyper-parameter that minimize the loss function and adjust the weights and biases of the parameters accordingly. There are several optimization algorithms available such as SGD, ADAM, NADAM, RMSProp, ADAMax, ADAGrade, and ADADelta. These Optimization algorithms can be used to monitor the performance of the model.

The learning rate is a parameter of the optimization algorithms that define how much information the optimization algorithm takes in to adjust the parameters. Learning rates are different and depend on the optimization algorithm used. These can be tweaked to monitor the performance of the model.

Image Sizes can be used as a hyper-parameter as the deep CNNs can perform differently on different image sizes. CNNs extract meaningful information from the pixels and their relationship with the surrounding pixels in the convolution layer. The smaller image sizes require less memory for processing during training while larger image sizes are more memory intensive. Larger images can be reduced in size while retaining the information but increasing the size of the smaller images can distort pixels which can lead to loss of information and important features and the model may not perform well. Keeping that in mind, different image sizes can be used to monitor the performance of the model.

For this study, models were fine-tuned based on all available optimization algorithms only because of limited computational and memory resources available as finetuning

on all hyper-parameters is a long and time-consuming process. Once the model has been fine-tuned, it can be evaluated on different unseen data to monitor the generalization ability of the model. The evaluation metrics are explained next.

3.7 Evaluation Metrics

After fine-tuning the models, they are tested on separate unseen testing data and evaluated on performance based on the following performance metrics.

3.7.1 Accuracy

Accuracy is a performance metric used to evaluate a model's overall performance by checking how many instances the model predicted correctly out of the overall instances. Accuracy is measured as follows.

$$\boldsymbol{Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}}$$

Equation 2

This provides an overall view of the performance of a model.

3.7.2 Precision

Precision is a performance metric that can be used on individual classes for further evaluation where it provides the proportion of correctly predicted positives out of all the instances predicted positively by the model. It is calculated as follows.

$$\boldsymbol{Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}}$$

Equation 3

This is useful in cases where the cost of predicting false positives is high and this can be used for an overall view of the quality of positive predictions made by the model.

3.7.3 Recall

The recall is a performance metric that can be used to check how many instances the model predicted correctly out of all the actual positive instances and is usually referred

to as the sensitivity of the model as it tells how sensitive the model is towards predicting true positives. It is calculated as follows.

$$\boldsymbol{Recall} = \frac{\boldsymbol{True\ Positives}}{\boldsymbol{True\ positives} + \boldsymbol{False\ Negatives}}$$

Equation 4

It is useful where the cost of false negatives is high and to minimize the number of times a model predicts instances of negatives wrongly.

3.7.4 F1 Scores

F1 score is a harmonic mean of both precision and recall scores that provides a single score focusing on false positives and false negatives providing a balanced evaluation of the model. It is calculated as follows.

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

Equation 5

These performance metrics were used to evaluate the models where accuracy was used as to see the overall performance of the model and the rest were used to check on individual classes to evaluate how the models performed for each emotion class.

The next Chapter provides details about the system design and specifications.

Chapter 4. System Design and Specifications

This section provides information about the hardware and software used for this project as well as the programming languages, deep learning libraries, frameworks, and system configurations of CPU, GPU, and memory specification.

4.1 Hardware Specifications

Hardware specifications used to accomplish this project are as follows:

- Processor: Intel(R) Core(TM) i3-7020U CPU @ 2.30GHz 2.30 GHz
- RAM: 12.0 GB
- System Type: 64-bit operating system, x64-based processor
- Disk: Kingston M2 PCI Card 256 GB

4.2 Software Specifications

Software specifications used to accomplish this project are as follows:

- Operating System: Windows 10 Home 64 bit
- Google Chrome: Version 112.0.5615.138 (Official Build) (64-bit)
- Programming Language: Python 3.8
- IDE: Google Colab, Visual Studio Code
- Application Development Framework: Flask 2.0.3
- Image Processing Library: OpenCV 4.7.0
- Face Detection: MTCNN
- Machine Learning Library: Keras 2.12.0
- Data processing library: Numpy 1.23.0, Pandas 1.4.3
- Visualisation Library: MatPlotLib 3.5.3
- Image Generation Library: OpenAI 0.27.5
- Model Evaluation Library: SciKit Learn

4.3 Selection Criteria for Libraries & Frameworks

The design decisions that led to the selection of libraries and frameworks used in this project are explained as follows:

4.3.1 Python

Python is a powerful programming language that is easy to use, straightforward forward, and versatile and can be used for web development, data analysis, and machine learning which makes it ideal for this project. It also has huge community support allowing many libraries, frameworks, and resources for development, and its programs can be run on different platforms and operating systems which also made it an ideal choice for this project.

4.3.2 Google Colab

Google Colab provides a cloud-based environment where code can be run without needing to install libraries locally and provides access to computational resources like GPUs and TPUs which were essential for this project for processing and training models on image data. It also provides ease of sharing and connectivity with other Google services like Google Drive which was used to store all the important data and files providing more security and backup.

4.3.3 OpenCV

OpenCV is a Python library specially designed for computer vision tasks including image and video processing providing functionalities that were very useful for this project including image manipulation, transformation, filtering, and analysis which made this an ideal choice for this project.

4.3.4 MTCNN

MTCNN is a deep learning algorithm designed specifically for face detection from images and videos and is known for its high accuracy, real-time performance, and the ability to detect faces and facial landmarks from a variety of light conditions, angles, and occlusions making it an ideal choice for this project.

4.3.5 Keras

Keras is a deep learning library with an easy-to-use interface for building and training neural networks offering a flexible and modular design that allows developers to build neural networks easily by stacking layers. Moreover, it provides support for a wide range of applications including image classification, natural language processing and generative models and it can utilize backend frameworks like TensorFlow. Since this project requires building and loading up deep CNN models, Keras is an ideal choice for this project.

4.3.6 MatPlotLib

MatPlotLib is a library for creating static, animated, and interactive visualizations in Python with options to customize plots including colours, line styles, labels, and headings. Moreover, it supports a wide range of plots, graphs, and charts including scatter plots, bar plots, histograms, and 3D plots which are produced in high quality and suitable for academic papers, reports, and publications that makes it an ideal choice for this project.

4.3.7 OpenAI

OpenAI is known for developing state-of-the-art generative models like DALL-E 2 that can generate images based on textual prompts and it provides APIs that can be used to generate images programmatically in bulk. This project required generating images for increasing the dataset and the best part about DALL-E 2 images is that the images are not even from real people allowing the generation of data without the data privacy considerations of real people.

4.3.8 SciKit Learn

SciKit Learn is a Python library that provides a wide range of machine learning algorithms including classification, regression, and clustering, it is easy to use and provides functions for evaluation metrics like Confusion Metrics, Accuracy, Precision, Recall, and F1 Scores. This project utilized this library for these evaluation metrics as it is easy to get these using the library instead of calculating the scores from scratch.

4.3.9 Flask

Flask is a web development framework in Python known for its simplicity that provides routing mechanisms to map URLs to specific views or functions enabling the creation of dynamic web pages with the ability to use extensions and Python libraries in the backend that makes it easy to add additional functionalities. This project used Flask to develop a web application that can take a facial image and show the emotions in the image.

4.4 System Design & Architecture

This section provides details of the system design & architecture and how all the components work together.

First, a user uploads an image. The image moves through pre-processing steps where the MTCNN detects and crops the face. Then padding is added to the cropped image. If the height is greater than the width, padding is added width wise and vice versa. This makes the image dimensions equal. Then the image is converted into grayscale using OpenCV and histogram normalization is applied to normalize the brightness-to-contrast ratio. At this stage pre-processing finishes.

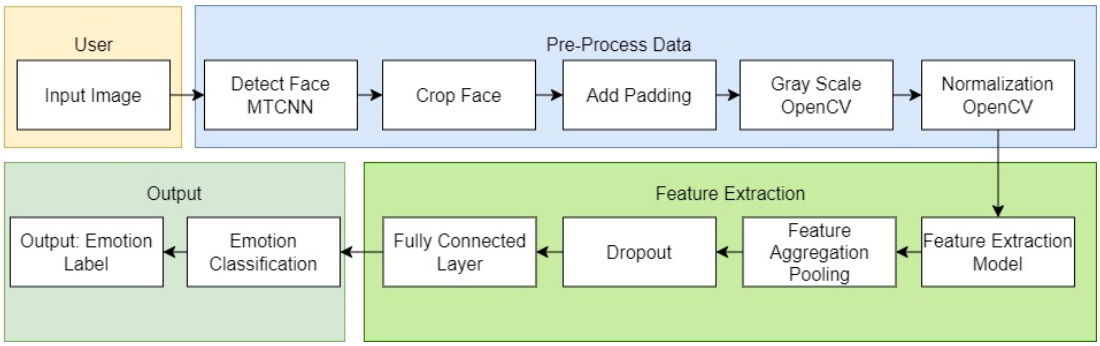

Figure 1: System Design

The image is then passed down to the model which extracts its features. Then features are aggregated using the Global Average Pooling layer. Then a dropout layer drops some features from the images, and it is passed on to the fully connected layer which

classifies the emotion. Based on the classification, the label of the emotion is presented as an output. The whole process is shown in the diagram below.

4.5 Model Design & Architecture

The pre-trained models were loaded with ImageNet weights and customized to add a Global Average Pooling layer that aggregates the information from the previous layer based on the overall average of the information gained. Another dense layer with 1024 filters was added and a drop-out layer was added at the end before it was attached to a fully connected layer. The same block that consists of global average pooling, dense layer, dropout layer, and a fully connected layer was used for both Xception and Inception V3 customizations. These diagrams are given in Appendix I.

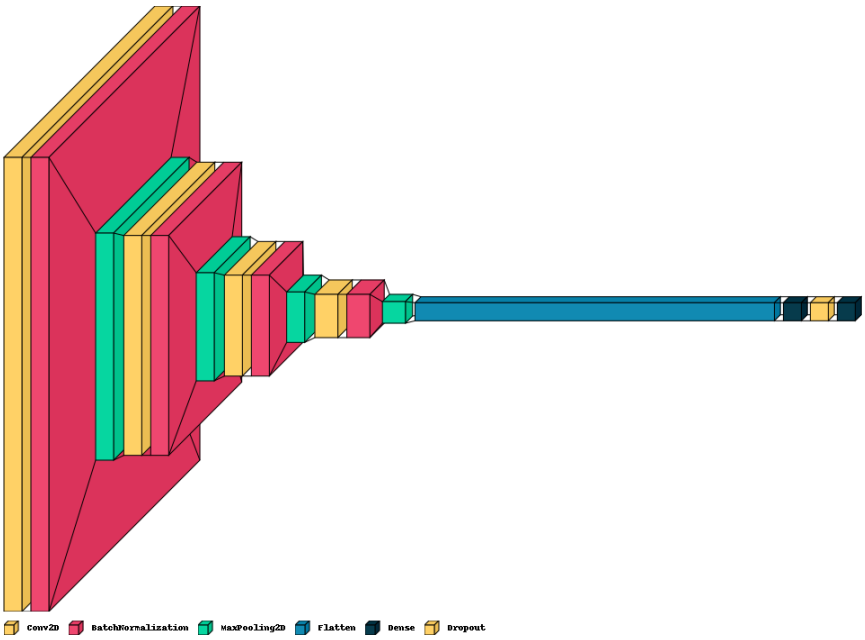

Figure 2: CNN Architecture

For the CNN developed from scratch, four convolution blocks were used. Each block contained a convolution layer, a batch normalization layer, and a 2D Maximum Pooling layer. The first convolution block had 32 filters, the second block had 64, the third had 128, and the fourth contained 256 filters. 'ReLu' was used as the activation function in all the blocks. These four blocks were then connected with a flattening layer to flatten the feature maps. Flattened feature maps were then passed onto a fully connected layer and a dropout layer. Finally, the last fully connected layer would classify the image using the SoftMax activation function.

Chapter 5. Implementation

This chapter explains how the project was implemented from start to finish including, data collection, pre-processing, model training, fine-tuning, and evaluation of the models.

5.1 Data Collection

This section explains the datasets used in this project, why these datasets were chosen, and additional steps taken for data collection.

5.1.1 Karolinska Directed Emotional Faces KDEF

This data set contains 4900 images of 70 individuals taken from 5 different angles showing 7 different emotions including anger, disgust, sadness, happiness, neutral, surprise, and fear. Out of 70, 35 individuals were male and 35 were female. The image sizes were 562 x 762 pixels. The dataset was collected from its official site by signing an NDA for not disclosing the dataset to anyone other than the individuals taking part in the research.[1]

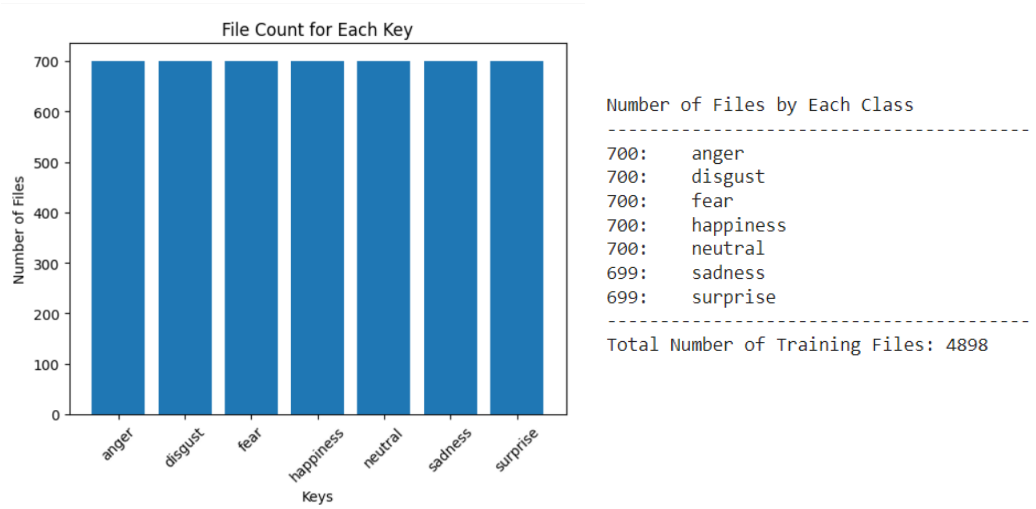

Figure 3: KDEF Dataset Bar Plot

Here are some sample images of a participant in the KDEF dataset.

[1] KDEF Dataset Source: https://kdef.se/download-2/

Figure 4: KDEF Sample Images

5.1.2 Japanese Female Facial Expressions JAFFE

This dataset was collected from the official dataset website and contains 213 images 256 x 256 pixels in size. The number of participants was 10 Japanese female models exhibiting seven basic emotions of anger, disgust, sadness, happiness, neutral, surprise, and fear. An NDA had to be signed for not sharing the dataset outside of the research scope and was received on terms of using it only for non-commercial scientific research.[2]

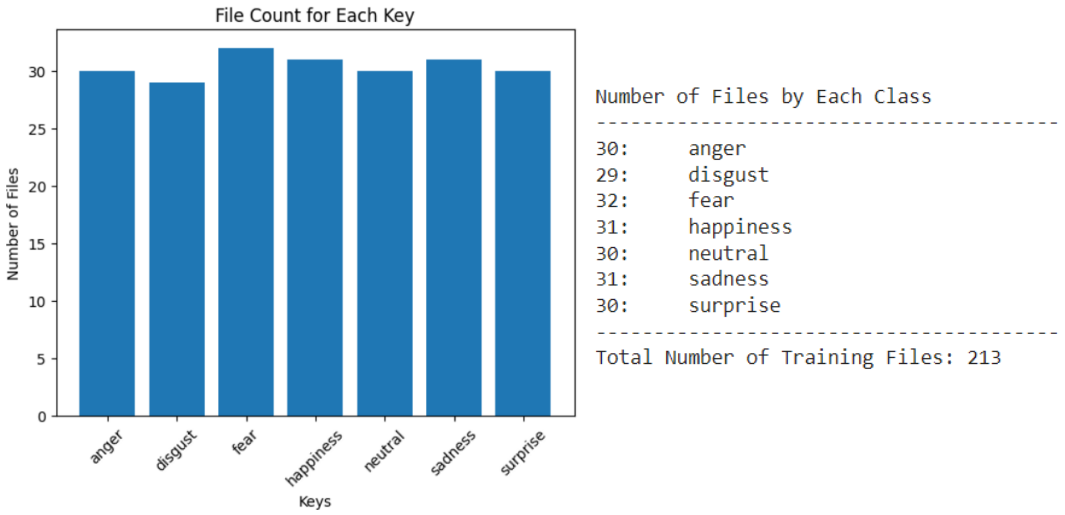

Figure 5: JAFFE Dataset Bar Plot

Here are some sample images from the JAFFE dataset.

Figure 6: JAFFE Sample Images

[2] JAFFE Dataset Source: https://zenodo.org/record/3451524

5.1.3 Using Generative AI Models

DALL-E 2 is a generative AI model by Open AI that can generate images using textual prompts. This model was used to generate 2,730 images of facial expressions exhibiting seven basic emotions of anger, disgust, sadness, happiness, neutral, surprise, and fear. The size of the images was 512 x 512 pixels. Textual prompts used to generate images were experimented with different variations until the prompt with the best results was chosen and the same prompt was used to generate images for all emotions by replacing the name of the emotion in the prompt. The figure below shows the details of images generated using DALL-E 2 for each emotion class.

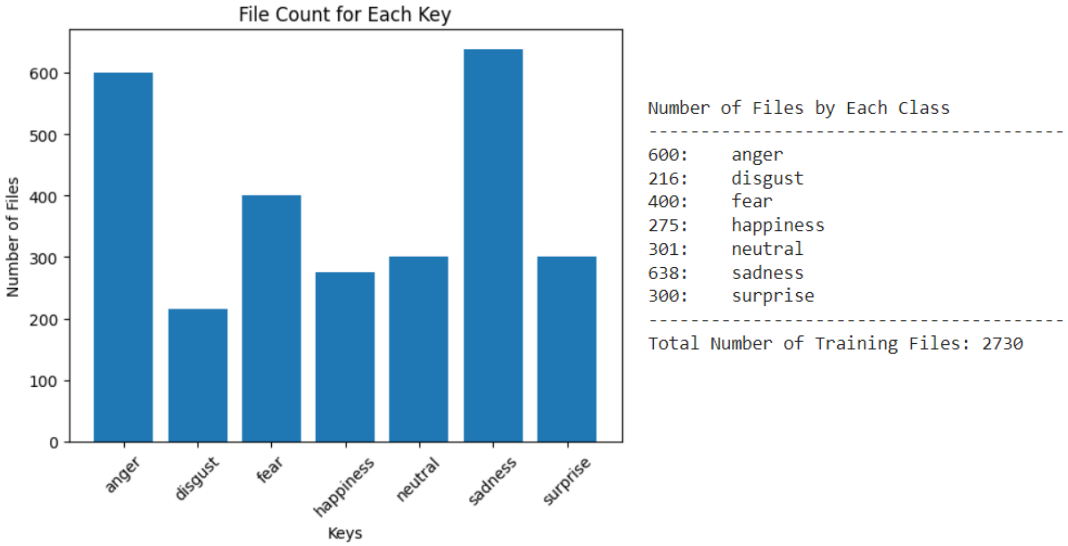

Figure 7: Open AI Dataset Bar Plot

Here are some sample images generated using DALL-E 2.

Figure 8: Open AI DALL-E 2 Sample Images

5.1.4 Creating Master Dataset

Once all the images were generated by DALL-E 2 and the datasets collected from official sources, a new dataset folder was created and for each dataset, images were extracted from the zip files and programmatically moved to their respective emotion

class folder to form a new dataset that contains images from all the datasets mentioned above. This new dataset contained seven emotion classes as folders with each having its respective images moved in it and was used for the next steps.

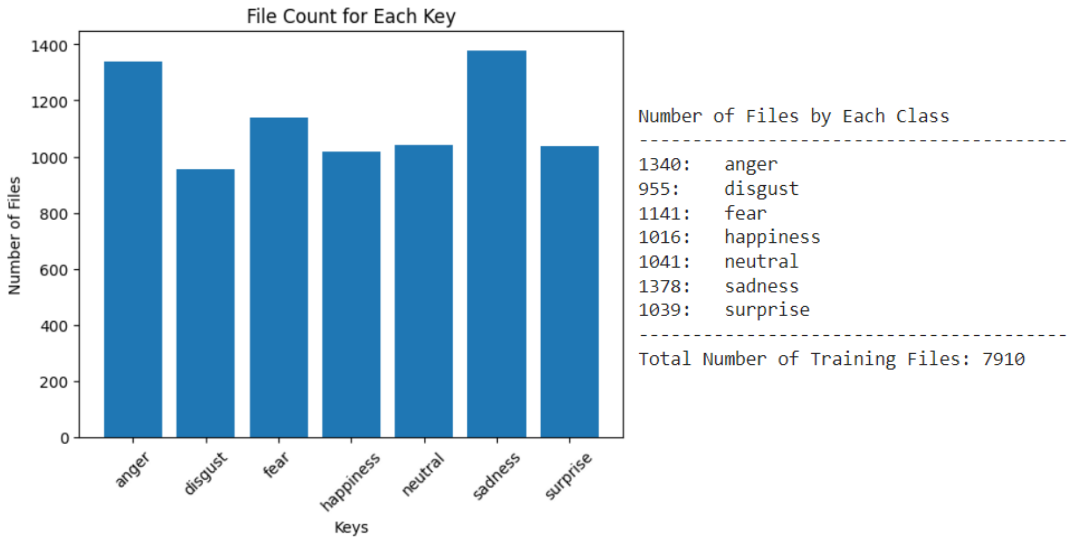

Figure 9: Master Dataset Bar Plot

5.2 Data Pre-Processing

Following pre-processing steps were applied to the datasets before the data could be used for training the models.

5.2.1 Face Detection, Crops, and Cleaning

Multi-task Convolution Neural Network [23] is a deep CNN-based model that can detect faces in unconstrained environments with a variety of poses, illumination, and occlusion conditions. This model was used to detect faces from images. Once the faces are detected, MTCNN gives the coordinates of the box frame that contains the face which is used for cropping faces.

Furthermore, this step was used to perform cleaning of the dataset from all the images where a face was not fully detected. Some images in the KDEF dataset have side poses that do not reflect full facial expressions. As the motives of this study are to detect emotions from facial expressions, such images were of no use. Image data generated from Open AI DALL-E 2 also had images that do not exhibit full faces or cartoon

images that were not useful for the training. All such images were filtered out during this step and were stored separately for further manual inspection.

Here is a sample of such images that were filtered out.

Figure 10: Sample Images Removed in Face Detection Phase

5.2.2 Rescale Images

The dimensions of each image after the face crop varied. To overcome this challenge, padding was added to the images. If the height of the cropped image was greater than its width, padding was added width wise. If the width of the cropped image was greater than the height, padding was added height-wise. Padding was added against the size of 128 x 128 pixels image size to ensure consistent image sizes. After adding the padding, the cropped face images were placed in the centre of the frame to ensure further consistency.

5.2.3 Grayscale Conversion

The images passed to the models for training need to be of the same colour. Since different datasets had a different colour scheme for images, they were all converted to grayscale using Python's Open CV library. This ensured consistency in the training images.

5.2.4 Grayscale Normalisation

The contrast and brightness of the image can also have an impact on the learning of the training models as it can become difficult to detect and extract edges of the face and facial features if the brightness is too high and a high contrast can darken some facial features that are important for training [4]. Histogram normalization from Python's Open CV library transforms the image graph into a uniform distribution graph. This ensured a consistent brightness-to-contrast ratio for all the images.

Most of the pre-processing was complete after all the above steps and the images obtained had a uniform size, padding, colour scheme, and normalized brightness-to-contrast ratio. Here is a sample of pre-processed images.

Figure 11: Sample Images after Pre-Processing

5.2.5 Post-Process Cleaning

Each class folder was checked one by one for any overly processed images or images that might not be required for training. The manual inspection led to finding out images that were animated facial expressions. 47 such images were deleted manually. Some samples of these images are as follows.

Figure 12: Sample Animated Images Removed in Post Processing

Moreover, 92 images in total were overly processed by the MTCNN face detection step where the face was detected but was not there in the image. Such images were deleted as well. Some sample images are also posted below.

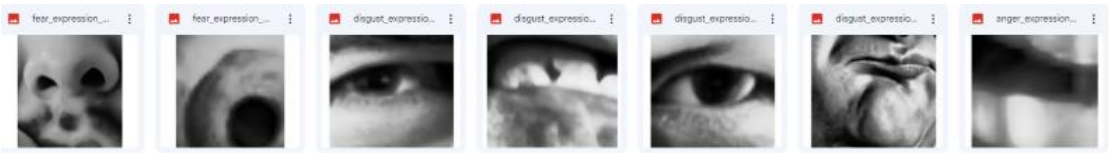

Figure 13: Sample Over Processed Images Removed in Post Processing

Once this subjective evaluation was over, the dataset was ready to be prepared for training. For this, it was split into training, validation, and testing sets, and a further data augmentation step was taken the details of which will follow in the next section.

5.2.6 Training, Validation, and Testing Splits

For this project, the model was split into training, validation, and testing datasets where 100 images for each class were randomly chosen from the master dataset and moved to these sets which gives us 700 images for validation and 700 images for testing the datasets. The remaining images were kept for the training dataset.

5.2.7 Data Augmentation

The lowest number of images among all classes was 225 for the emotion class of surprise. This number was chosen, and 225 randomly selected images were flipped horizontally as a copy in the same class to increase the number of images. This way the class folders with low images were increased almost by double but the number of images in classes with high numbers was increased by some portion. In the end, 100 flipped copies from two classes were removed since those are just copies and the originals already exist it will not affect the training. This way, the training examples were increased with slight variations in the images. Horizontal flips were deemed appropriate because the human face is symmetrical in design and the left-side expressions may vary a bit from right-side expressions by doing a horizontal flip the model can learn a few examples of such existence of differences in expressions on either side of facial symmetry. The chart below shows the details of the images in the training set after the data augmentation was performed.

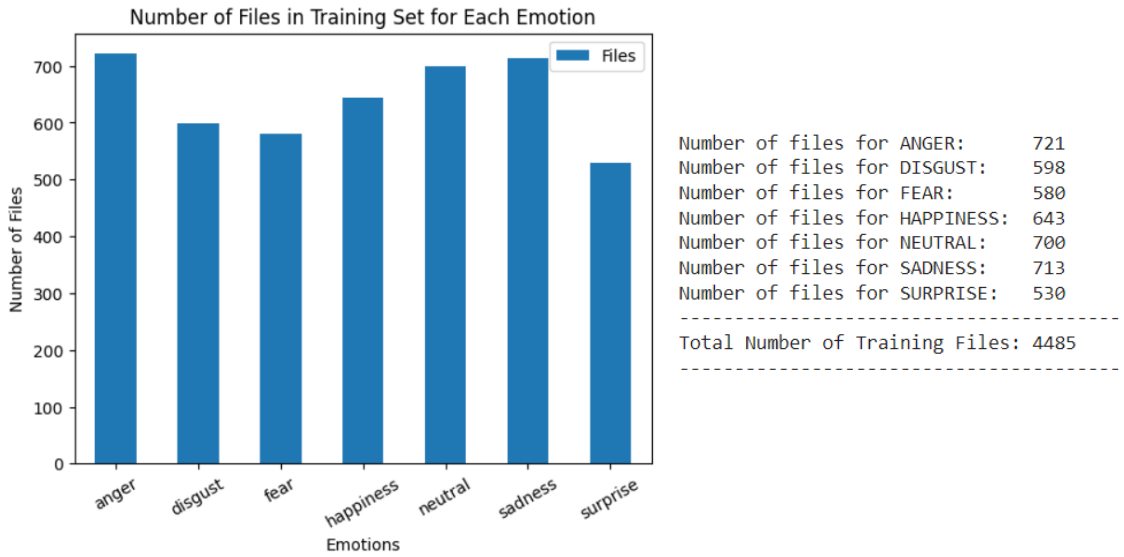

Figure 14: Final Master Dataset for Training Bar Plot

After data augmentation, the master dataset was prepared using Image Data Generator from the Keras library and divided into batch sizes of 64 with image sizes of 128 x 128 pixels with a random shuffle.

5.3 Training Models

Xception and Inception V3 models were loaded with default ImageNet weights and the customized block was attached at the end of the models that consisted of a Global Average Pooling layer with a fully connected Dense layer with ReLu activation function, a drop-out layer and the output layer using SoftMax activation function. The Custom CNN model was also constructed based on the architecture explained in the model Design and Architecture section.

Each model was run for 120 EPOCHS using categorical cross entropy as the loss function and ADAM as the default optimization algorithm with a learning rate of 0.0001.

During training, the Xception model reached a training accuracy of 99.75% with a loss of 0.0114 and a validation accuracy of 90.42% with a validation loss of 0.4617. Inception V3 reached a training accuracy of 99.58% with a training loss of 0.0153 and a validation accuracy of 90.85% with a validation loss of 0.5344. The Custom CNN model reached a training accuracy of 97.95% with a loss of 0.0579 and a validation accuracy of 86.85% with a loss of 0.6341.
The results are shown in the table below.

Table 1: Performance of Models During Training

Model	Train Loss	Train Accuracy	Val Loss	Val Accuracy
Xception	**0.0114**	**99.75%**	**0.4617**	**90.42%**
Inception V3	**0.0153**	**99.58%**	**0.5344**	**90.85%**
Custom CNN	**0.0579**	**97.95%**	**0.6341**	**86.85%**

5.4 Fine Tuning Models

Fine-tuning was conducted on five different optimization algorithms including SGD, ADAMax, ADAGrade, ADADelta, and NADAM. Xception had the best validation accuracy of 93.57% on ADADelta with a validation loss of 0.4244. Inception V3 had the best validation accuracy of 93.86% with a loss of 0.4742. The Custom CNN model had the best validation accuracy of 89.71% with a validation loss of 1.6448. The performance of the model on different Optimizers during finetuning is mentioned in the table below.

Table 2: Performance of Model on Different Optimisers

Model	Optimizers	Train Loss	Train Acc	Val Loss	Val Acc
Xception	SGD	0.0089	99.87%	0.4391	91.57%
	ADAGrade	0.0054	99.84%	0.4974	90.43%
	ADADelta	1.3722e-04	100.00%	0.4244	93.57%
	ADAMax	6.0474e-05	100.00%	0.5926	93.14%
	NADAM	1.7264e-05	100.00%	0.7067	93.14%
Inception	SGD	0.0031	99.93%	0.4731	92.00%
	ADAGrade	0.0054	99.82%	0.4577	91.42%
	ADADelta	0.0158	99.69%	0.4243	92.86%
	ADAMax	8.9052e-04	99.96%	0.4742	93.86%
	NADAM	0.0012	99.96%	0.5523	93.27%
Custom CNN	SGD	0.0335	98.84%	0.6215	85.86%
	ADAGrade	0.0128	99.62%	0.7025	88.86%
	ADADelta	0.0677	98.51%	1.2519	88.14%
	ADAMax	0.0049	99.80%	1.6448	89.71%
	NADAM	0.0130	99.49%	0.9580	88.29%

Afterward, the models were evaluated on separate testing data which is explained in the next section.

Chapter 6. Testing and Evaluation

This chapter discusses the testing of fine-tuned models and evaluations based on accuracy, precision, recall, and F1 scores.

6.1 Testing Models

Each fine-tuned model had six versions saved based on their fine-tuning on different optimizers. All these versions were loaded and tested on separate validation and testing datasets. The testing dataset contained 700 images in total with 100 images for each emotion class. None of these images were previously seen by the models during training.

Xception:

The Xception model fine-tuned on the ADA Max optimization algorithm performed best on both validation and test datasets with accuracies of 92.71% and 91.57% respectively. The accuracies of both were considered for evaluating performance as it tells how well the model can generalize on unseen datasets. The least performing optimizer was ADA Grade. Details are shown in the table below.

Table 3: Testing Performance of Xception with all Optimiser Versions

Optimizer	Val Loss	Val Acc	Test Loss	Test Acc
ADAM	**0.5082**	**90.57%**	**0.5468**	**89.86%**
ADA Delta	**0.4660**	**92.43%**	**0.5645**	**90.71%**
ADA Grade	**0.5326**	**89.29%**	**0.5542**	**88.43%**
NADAM	**0.7999**	**92.29%**	**0.8077**	**90.71%**
ADA Max	**0.6461**	**92.71%**	**0.7111**	**91.57%**
SGD	**0.4638**	**90.43%**	**0.5835**	**90.00%**

Inception V3

Inception V3 model fine-tuned on NADAM optimization algorithm performed best on both validation and test datasets with accuracies of 93.43% and 90.43% respectively. The least-performing optimizer was ADAM. Details are shown in the table below.

Table 4: Testing Performance of Inception V3 with all Optimiser Versions

Optimizer	Val Loss	Val Acc	Test Loss	Test Acc
ADAM	0.5913	90.00%	0.6891	88.00%
ADA Delta	0.4249	92.86%	0.7174	88.71%
ADA Grade	0.5134	90.14%	0.5938	89.14%
NADAM	0.5398	93.43%	0.8924	90.43%
ADA Max	0.5496	92.14%	0.8466	89.14%
SGD	0.5185	91.29%	0.5890	89.14%

Custom CNN

Custom CNN model fine-tuned on ADA Grade optimization algorithm performed best on both validation and test datasets with accuracies of 89.71% and 86.29% respectively. The least-performing optimizer was ADAM. Details are shown in the table below.

Table 5: Testing Performance of Custom CNN with all Optimiser Versions

Optimizer	Val Loss	Val Acc	Test Loss	Test Acc
ADAM	0.7314	85.00%	0.7862	82.71%
ADA Delta	1.3331	87.29%	1.9014	84.43%
ADA Grade	0.6877	89.71%	1.0381	86.29%
NADAM	1.0410	87.57%	1.3044	84.71%
ADA Max	1.7756	88.43%	2.4966	86.29%
SGD	0.7016	84.71%	0.7160	84.43%

After running all the tests, the best-performing versions of each model were compared with each to see how they performed. Xception performed better on testing data with an accuracy of 91.86% while custom-made CNN had the lowest performance. Inception

performed best on the validation dataset with the highest accuracy of 93.43% but the performance dropped to 90.43% on test data which was slightly lower than Xception. The table below shows the details of these versions.

Table 6: Top Performance of Each Model after Testing

Model	Val Loss	Val Acc	Test Loss	Test Acc
Xception	0.7459	92.86%	0.7838	91.86%
Inception V3	0.5398	93.43%	0.8924	90.43%
Custom CNN	0.6877	89.71%	1.0381	86.29%

6.2 Evaluating Models

Xception outperformed Inception V3 and the custom CNN model during testing. Accuracy and Loss were the only performance metrics monitored till the last section. This section explores further the classes and evaluates individual emotion labels to see how the model performs for each emotion class.

A quick look at the confusion metrics shows that sadness and surprise are two emotions that the model confused most of the time with 9 images of sadness predicted as surprise and 9 images of surprise predicted as sadness. Disgust is another emotion that is confused with neutral where 7 images of disgust were predicted as neutral and 6 images of neutral were predicted as disgust. The model performed very well in predicting anger, fear, and happiness.

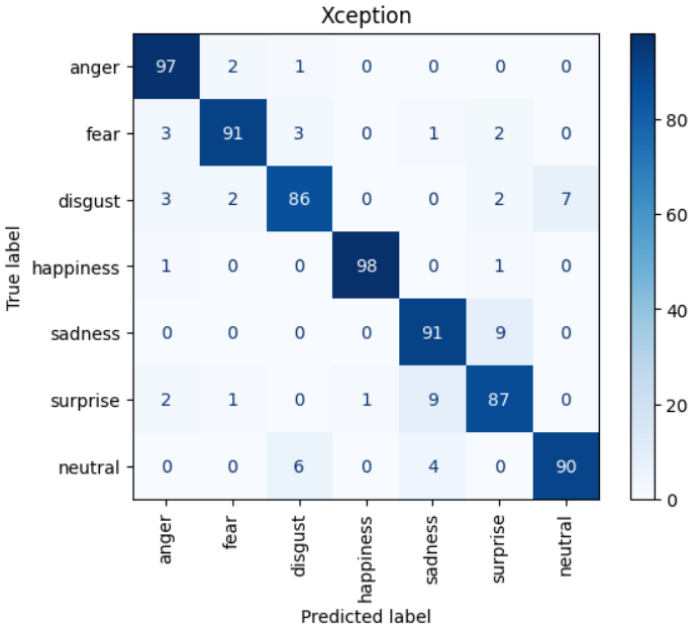

Figure 15: Confusion Matrix Xception

This can be further clarified by looking at the precision, recall, and F1 scores of each emotion class given in the next figure.

----------Xception----------

	precision	recall	f1-score	support
anger	0.92	0.97	0.94	100
fear	0.95	0.91	0.93	100
disgust	0.90	0.86	0.88	100
happiness	0.99	0.98	0.98	100
sadness	0.87	0.91	0.89	100
surprise	0.86	0.87	0.87	100
neutral	0.93	0.90	0.91	100
accuracy			0.91	700
macro avg	0.91	0.91	0.91	700
weighted avg	0.91	0.91	0.91	700

Figure 16: Classification Report Xception

The precision score focuses on the quality of the positive predictions made by the model by checking how many instances the model correctly predicted as positives out of all the positive predictions. Precision is ranging from 0.86 to 0.99 indicating that the model has performed well in identifying emotions with their respective classes. The precision scores for sadness and surprise are the lowest at 87% and 86% further clarifying that the model was confusing these emotion classes.

The recall scores are ranging from 0.86 to 0.98 indicating that the model has identified the emotion classes well. The recall is the ratio of how many instances the model predicted true positives out of all the instances of that class. F1 Scores combine both precision and recall and show that all emotions were predicted well by the model apart from sadness and surprise having the lowest scores of 89% and 87% respectively.

Inception V3 also had low performance on disgust and surprise where 10 images of surprise were predicted as sadness and 8 images of disgust were

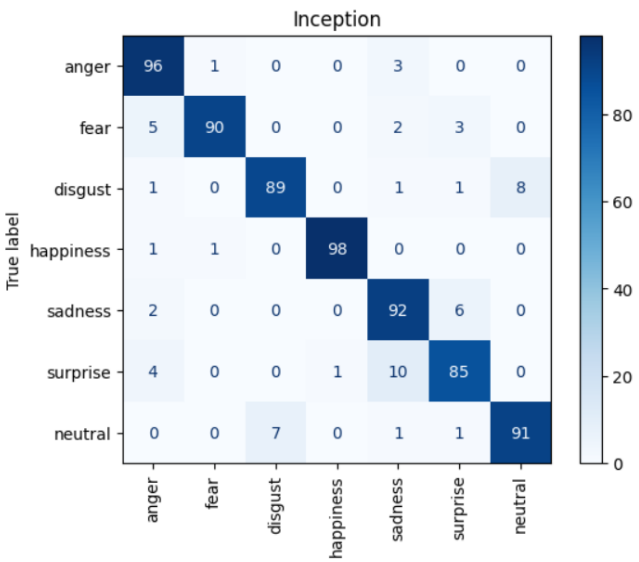

Figure 17: Confusion Matrix Inception V3

predicted as neutral indicating that the model is confusing disgust with neutral and sadness with surprise.

The same labels can be seen performing low in the classification report of the Inception V3 model below where precision and F1 scores of sadness and surprise are low which means this model is getting confused on the same labels as well. If two high-performing models are getting confused on the same labels, it's a good idea to explore the respective labels to see if there are any similarities between the features of the images. Moreover, disgust and neutral can also be seen performing a bit lower in Inception V3 compared to Xception. Here's the classification report.

----------Inception----------

	precision	recall	f1-score	support
anger	0.88	0.96	0.92	100
fear	0.98	0.90	0.94	100
disgust	0.93	0.89	0.91	100
happiness	0.99	0.98	0.98	100
sadness	0.84	0.92	0.88	100
surprise	0.89	0.85	0.87	100
neutral	0.92	0.91	0.91	100
accuracy			0.92	700
macro avg	0.92	0.92	0.92	700
weighted avg	0.92	0.92	0.92	700

Figure 18: Classification Report Inception V3

The Custom CNN model had the lowest performance on the testing data with an overall accuracy of 86.29%. The confusion matrix shows the model was able to successfully predict anger and happiness but confused 20 neutral labels with disgust and 8 disgust labels with neutral labels. The model performed lowest in predicting neutral images while its performance on sadness and surprise emotions was almost the same as Xception and Inception V3.

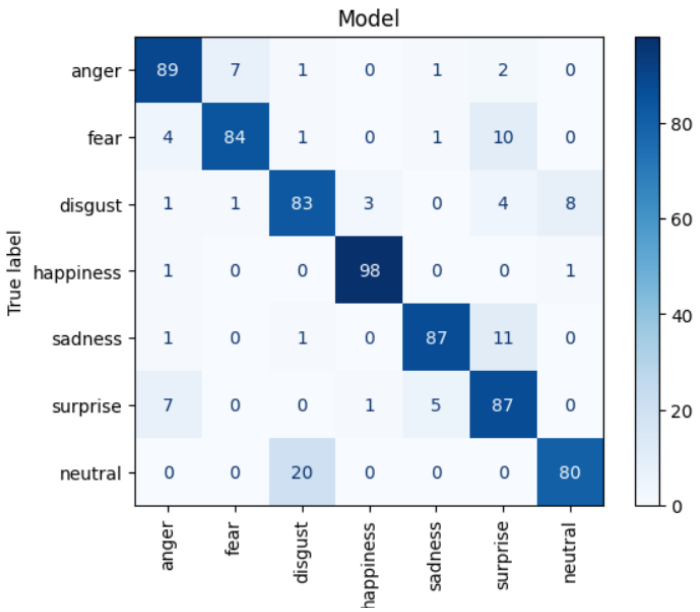

Figure 19: Confusion Matrix Custom CNN

The classification report further clarifies this where disgust and surprise have the lowest precision and F1 Scores indicating low performance on these variables. Here is the classification report.

----------Model----------

	precision	recall	f1-score	support
anger	0.86	0.89	0.88	100
fear	0.91	0.84	0.87	100
disgust	0.78	0.83	0.81	100
happiness	0.96	0.98	0.97	100
sadness	0.93	0.87	0.90	100
surprise	0.76	0.87	0.81	100
neutral	0.90	0.80	0.85	100
accuracy			0.87	700
macro avg	0.87	0.87	0.87	700
weighted avg	0.87	0.87	0.87	700

Figure 20: Classification Report Custom CNN

The overall outlook on the performance of the models suggests that the emotions of disgust, surprise, and sadness are confused by the models a lot and perhaps better-quality training images can improve these performance issues, but the overall performance of the models is still good with accuracy reaching 87% above for custom CNN and 92%+ for Xception and Inception V3.

Chapter 7. Conclusion & Future Work

This section provides the concluding remarks and the future work that can be performed for this study.

7.1 Conclusion

Xception was the top performing model compared to Inception V3 and the custom CNN with a training accuracy of 99.75% and a loss of 0.0114, validation accuracy of 92.86% and loss of 0.7459 and testing accuracy of 91.86% and loss of 0.7838. Inception V3 was the second-best model performing with a training accuracy of 99.58% with a loss of 0.0153, validation accuracy of 93.43% and loss of 0.5398, and testing accuracy of 90.43% with a loss of 0.8924. These models performed better than the custom CNN justifying that transfer learning is better than deep learning and making models from scratch.

While custom-built CNN performed lower than pre-trained models the performance was not poor as it achieved a training accuracy of 97.95% with a loss of 0.0579, a validation accuracy of 89.71% with a loss of 0.6877 and a testing accuracy of 86.29% with a loss of 1.0381. The model needs more regularisation as the performance fluctuated a lot in terms of accuracy and loss during training. This further justifies the ease of using the transfer learning approach over building models from scratch as it can take a lot of time and research for debugging and constructing high-performing models that can achieve better results that can be achieved by loading pre-trained models and weights easily.

Data quality is another important point that can enhance the performance of the models. The models underperformed when predicting the emotion classes of disgust, sadness, and surprise. Perhaps better-quality images that differentiate the facial features of these emotions can improve the performance of the models as most of the images in these emotion classes have quite similar facial features.

7.2 Future Work

As suggested before, the emotion classes of sadness, surprise, and disgust were confused a lot by the models so in future work better quality images can be added to the dataset for such emotions that have to differentiate facial features for models to recognize. Moreover, the dataset was small because of the limited computational resources and time and more images in the overall dataset can be added to see if the models learn better.

Fine-tuning of the models was performed only based on optimization algorithms. Further, fine-tuning can be performed based on batch sizes, image sizes, epochs, and different learning rates to see any improvements in the performance of the models. Moreover, the application can only predict emotions from an input image of a single individual with a uniform background. More features can be added to the application where the model can predict the emotions of multiple individuals in an image as well as from videos in real-time.

References

[1] K. VEMOU and A. HORVATH, "Facial Emotion Recognition," *Tech Dispatch,* no. 1, 2021.

[2] I. Lasri, A. R. Solh and M. Belkacemi, "Facial Emotion Recognition of Students using Convolutional Neural Network," *IEEE,* 2019.

[3] A. Verma, P. Singh and J. S. R. Alex, "Modified Convolutional Neural Network Architecture Analysis for Facial Emotion Recognition," *IEEE,* pp. 169-173, 2019.

[4] H. ZHANG, A. JOLFAEI and M. ALAZAB, "A Face Emotion Recognition Method Using Convolutional Neural Network and Image Edge Computing," *IEEE Access,* vol. 7, 2019.

[5] S. Shaees, H. Naeem, M. Arslan, M. R. Naeem, S. H. Ali and H. Aldabbas, "Facial Emotion Recognition Using Transfer Learning," *International Conference on Computing and Information Technology,* vol. 1, no. 1441, pp. 192-196, 2020.

[6] J. H. Kim, A. Poulose and D. S. Han, "The Extensive Usage of the Facial Image Threshing Machine for Facial Emotion Recognition Performance," *Sensors,* vol. 21, no. 2026, 2021.

[7] S. Mishra, B. Joshi, R. Paudyal, D. Chaulagain and S. Shakya, "Deep Residual Learning For Facial Emotion Recognition," *Research Gate,* 2021.

[8] C. DALVI, M. RATHOD, S. PATIL, S. GITE and K. KOTECHA, "A Survey of AI-Based Facial Emotion Recognition: Features, ML & DL Techniques, Age-Wise Datasets and Future Directions," *IEEE Access,* vol. 9, 2021.

[9] Y. Khaireddin and Z. Chen, "Facial emotion recognition: State of the art performance on FER2013," *arXiv preprint,* p. arXiv:2105.03588, 2021.

[10] H. Akhand, S. Roy, N. Siddique, A. S. Kamal and T. Shimamura, "Facial Emotion Recognition Using Transfer Learning in the Deep CNN," *Electronics,* vol. 10, no. 1036, 2021.

[11] M. . M. T. Zadeh, M. Mohammad Taghi and B. Majidi, "Fast Facial emotion recognition Using Convolutional Neural Networks and Gabor Filters," *5th Conference on Knowledge Based Engineering and Innovation (KBEI) IEEE,* pp. 577-581, 2019.

[12] K. Sumiran, "An Overview of Data Mining Techniques and Their Application in Industrial Engineering," *Asian Journal of Applied Science and Technology (AJAST),* vol. 2, no. 2, pp. 947-953, 2018.

[13] A. Mostafa and H. Mahmoud, "Review of Data Mining Concept and its Techniques," *International Journal of Academic Research in Business and Social Sciences,* vol. 12, no. 6, pp. 611-699, 2022.

[14] X. Teng and Y. Gong, "Research on Application of Machine Learning in Data Mining," *IOP Conference Series: Materials Science and Engineering,* vol. 392, no. 6, p. 062202, 2018.

[15] V. Plotnikova, M. Dumas and F. P. Milani, “Data Mining Methodologies in the Banking Domain: A Systematic Literature Review,” *International Conference on Business Informatics Research,* pp. 104-118, 2019.

[16] L. Zahara, P. Musa, E. P. Wibowo, I. Karim and S. B. Musa, “The Facial Emotion Recognition (FER-2013) Dataset for Prediction System of Micro-Expressions Face Using the Convolutional Neural Network (CNN) Algorithm based Raspberry Pi,” *Research Gate,* 2020.

[17] 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,” *Computing and Machine Learning IEEE,* 2022.

[18] A. Siam, N. Soliman, A. Algarni, F. Abd El-Samie and A. Sedik, “Deploying Machine Learning Techniques for Human Emotion Detection,” *Computational Intelligence and Neuroscience,* p. 16, 2022.

[19] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, C. C. Loy, Y. Qiao and X. Tang, “ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks,” *arXiv,* 2018.

[20] A. Alreshidi and M. Ullah, “Facial Emotion Recognition Using Hybrid Features,” *Informatics,* vol. 7, no. 6, 2020.

[21] C. Szegedy, V. Vanhoucke, S. Ioffe and J. Shlens, “Rethinking the Inception Architecture for Computer Vision,” *arXiv,* 2015.

[22] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” *arXiv,* 2017.

[23] K. Zhang, Z. Zhang, Z. Li and Y. Qiao, “Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks,” *IEEE Signal Processing Letters,* vol. 23, no. 10, 2016.

Appendix I