

# A Proposed Novel Approach to Face Recognition Using CNN

by

Asif Mahmud  
17201047

A thesis submitted to the Department of Computer Science and  
Engineering  
in partial fulfillment of the requirements for the degree of  
B.Sc. in Computer Science

Department of Computer Science and Engineering  
Brac University  
September 2023

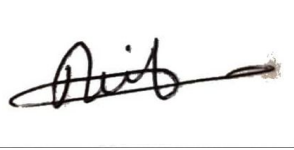
© 2023. Brac University  
All rights reserved.

# Declaration

It is hereby declared that

1. The thesis submitted is my own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. This research is acknowledged by me in all main sources of help.

**Student's Full Name & Signature:**

A handwritten signature in black ink, appearing to read 'Asif', enclosed within a thin black rectangular border.

---

Asif Mahmud  
18301261

# Approval

The thesis titled “A Proposed Novel Approach to Face Recognition Using CNN” submitted by

1. Asif Mahmud (17201047)

Of Summer, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science and Engineering.

## Examining Committee:

Supervisor:  
(Member)



---

Arif Shakil  
Lecturer  
Department of computer science and Engineering  
Brac University

Program Coordinator:  
(Member)

---

Dr.Md.Golam Rabiul Alam  
Associate Professor  
Department of Computer Science And Engineering  
Brac University

Head of Department:  
(Chair)

---

Dr.Sadia Hamid Kazi  
Chairperson and Associate Professor  
Department of Computer Science and Engineering  
Brac University

# Abstract

Facial recognition has emerged as a crucial technology with applications spanning from security and surveillance to user authentication and human-computer interaction. This work is a comprehensive study of facial recognition techniques leveraging Convolutional Neural Networks (CNN), which is a class of deep learning models known for their exceptional performance in image processing tasks. The primary objective of my research is to develop a strong facial recognition system capable of accurately identifying individuals across various real-world scenarios and challenges. The thesis begins by providing an overview of the fundamentals of CNN, their architecture, and their relevance in image-based pattern recognition. Then it delves into the pre-processing steps involved in preparing facial images for CNN-based recognition, including data collection, data augmentation, and face detection. Special attention is given to handling occlusion, illumination variations, and pose changes often encountered in real-world environments. The core of my work focuses on the design and implementation of CNN-based facial recognition models. Different CNN architectures are explored, and their performance is evaluated using benchmark datasets. In this research 21000 images from Kaggle as the dataset are used. The pre-trained models are used for the improvement of recognition accuracy, even with limited training data. Experimental results demonstrate the effectiveness of CNN-based facial recognition models in achieving high accuracy and robustness across varying conditions. This research segmented the process into three parts: Testing, training, and validation. Firstly, the proposed CNN model was trained with this dataset. Moreover, some pre-trained models are also run. They are: Inceptionv3, EfficientNet B0, EfficientNet B6, Xception, and Resnet50. This contributes to the field of facial recognition by offering a comprehensive exploration of CNN-based techniques and addressing real-world challenges.

**Keywords:** CNN, Pre-processing, Training, Testing, Dataset, Exploration, Inceptionv3, EfficientNet B0, EfficientNet B6, Xception, and Resnet50.

## **Dedication**

Without the assistance of my outstanding supervisor who has been a constant source of guidance and counsel, I would not have been able to complete this thesis. This paper is dedicated to him.

## **Acknowledgement**

Firstly, I want to thank Allah, the Almighty, for giving me the tools, chances, and direction I needed to finish this research. As a second tribute, I would like to express my gratitude to Mr. Arif Shakil, my thesis supervisor. I shall always be grateful for his direction and motivation to do my work successfully.

# Table of Contents

<b>List of Figures</b>	<b>VIII</b>
<b>List of Tables</b>	<b>IX</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Background . . . . .	1
1.2 Problem Statement . . . . .	2
1.3 Motivation . . . . .	3
1.4 Thesis Orientation . . . . .	4
<b>2 Related Work</b>	<b>5</b>
<b>3 Research Objective</b>	<b>7</b>
<b>4 Methodology</b>	<b>9</b>
4.1 Working Plan . . . . .	9
4.2 Dataset . . . . .	10
4.3 Data Sample: . . . . .	10
4.4 Data Preprocessing . . . . .	11
4.5 Training Set: . . . . .	12
4.6 Testing Set: . . . . .	12
<b>5 CNN Model Implementation</b>	<b>13</b>
5.1 Introduction to CNN . . . . .	13
5.1.1 Optimizer Adam: . . . . .	15
5.1.2 Softmax: . . . . .	15
5.2 Model Architecture Using CNN: . . . . .	16
5.2.1 Input Layer: . . . . .	16
5.2.2 Convolutional Layers: . . . . .	16
5.2.3 Activation Function: . . . . .	17
5.2.4 Pooling Layers: . . . . .	17
5.2.5 Fully Connected Layers: . . . . .	17
5.2.6 Output Layer: . . . . .	17
5.2.7 Loss Function: . . . . .	17
5.3 Proposed CNN Model . . . . .	18
5.3.1 Implement to Proposed Model: . . . . .	18
5.4 Proposed model Architecture: . . . . .	19
5.4.1 22-Layered CNN Model: . . . . .	19
5.5 Convolutional Layer: . . . . .	19

5.6	Batch Normalization: . . . . .	19
5.7	Pooling Layer: . . . . .	19
5.8	Fully Connected Layer: . . . . .	20
5.9	Proposed Model Summary . . . . .	21
5.10	Pre-Trained Model of CNN . . . . .	22
5.10.1	InceptionV3: . . . . .	22
5.10.2	EfficientNet B0 . . . . .	23
5.10.3	Xception: . . . . .	23
5.10.4	ResNet50: . . . . .	24
5.10.5	EfficientNet B6: . . . . .	25
<b>6</b>	<b>Performance Analysis</b>	<b>26</b>
6.1	Performance Parameter: . . . . .	26
6.2	Performance Parameter: . . . . .	27
6.3	Performance of Pre-trained Models . . . . .	29
6.3.1	Inception V3: . . . . .	29
6.3.2	Xception: . . . . .	30
6.3.3	EfficientNet B0: . . . . .	31
6.3.4	EfficientNet B6: . . . . .	32
6.3.5	ResNet50: . . . . .	33
6.4	Compare and Analysis . . . . .	34
<b>7</b>	<b>Conclusion</b>	<b>35</b>
7.1	Future Work . . . . .	36
	<b>Bibliography</b>	<b>37</b>



# List of Figures

4.1	Work Flow Diagram . . . . .	9
4.2	Sample Data . . . . .	10
5.1	CNN's Operational Process . . . . .	14
5.2	Softmax Equation . . . . .	15
5.3	Illustration of Softmax Activation . . . . .	16
5.4	Max pooling . . . . .	17
5.5	Architecture of CNN . . . . .	18
5.6	Pooling Layer . . . . .	20
5.7	Inception V3 Architecture . . . . .	22
5.8	EfficientNet B0 Architecture . . . . .	23
5.9	Xception Architecture . . . . .	24
5.10	ResNet50 Architecture . . . . .	24
5.11	EfficientNet B6 Architecture . . . . .	25
6.1	Training and Validation loss graph of proposed model . . . . .	28
6.2	Training and Validation accuracy graph of proposed model . . . . .	28
6.3	Training and Validation loss graph of Inception V3 model . . . . .	29
6.4	Training and Validation accuracy graph of Inception V3 model . . . . .	29
6.5	Training and Validation loss graph of Xception model . . . . .	30
6.6	Training and Validation accuracy graph of Xception model . . . . .	30
6.7	Training and Validation loss graph of Efficientnet B0 model . . . . .	31
6.8	Training and Validation accuracy graph of Efficientnet B0 model . . . . .	31
6.9	Training and Validation loss graph of EfficientNet B6 model . . . . .	32
6.10	Training and Validation accuracy graph of EfficientNet B6 model . . . . .	32
6.11	Training and Validation loss graph of ResNet50 model . . . . .	33
6.12	Training and Validation accuracy graph of ResNet50 model . . . . .	33
6.13	Accuracy comparison of all models . . . . .	34

# List of Tables

5.1	Table of proposed CNN model . . . . .	21
6.1	Proposed model Parameters . . . . .	27
6.2	Proposed model Loss and Accuracy . . . . .	27
6.3	Comparison between all models . . . . .	34

## Nomenclature:

This part describes the symbols and their abbreviation which are going to use in our whole thesis paper over and over. This part will help to recognize them easily.

AI	Artificial Intelligence
CNN	Convolutional Neural Network
LFW	Labeled Faces in the Wild
LBP	Local Binary Pattern
GDPR	General Data Protection Regulation
TFRT	Traditional Facial Recognition Techniques
DLFR	Deep Learning for Facial Recognition
TLPM	Transfer Learning and Pretrained Models
DFR	Datasets for Facial Recognition
PSC	Privacy and Security Challenges

# Chapter 1

## Introduction

### 1.1 Background

The research of computer vision has seen a revival in an increasingly linked digital world with significant implications for security, customization, and human-computer interaction. Facial recognition is one of the most fascinating and revolutionary uses of computer vision technology. This is the use of the capabilities of Convolutional Neural Networks, a breakthrough class of deep learning models to investigate the dynamic and changing facial recognition landscape. Once confined to science fiction, facial recognition is now a crucial aspect of our daily lives. This technology offers unmatched ease and security, from unlocking cellphones to improving security systems and streamlining access control. The road to developing trustworthy and moral facial recognition systems. However, is paved with difficulties and moral dilemmas. The argument develops against a backdrop of technological and societal development, where issues with prejudice, privacy, and security are major concerns. Concerns regarding the possible abuse of face recognition systems and the degradation of individual privacy have risen to the fore in public conversation as they become more widely used. This study makes its way through this complicated environment, attempting to strike a balance between the unquestionable value of facial recognition and the need to protect people's rights and the general welfare of society through CNN. CNN is a class of neural networks derived from the human visual system. These deep-learning models are excellent at identifying complex patterns in pictures, which makes them perfect for jobs like facial recognition. The study expands on this groundwork by examining the architecture. It strongly emphasizes ethical considerations in addition to technological progress. It addresses the possible drawbacks of facial recognition systems that may disproportionately affect particular demographic groups while grappling with issues of fairness and bias. To guarantee that the technology serves society as a whole, without compromising individual rights or maintaining prejudice, ethical principles and responsible deployment methodologies are presented. Therefore, this is more than just a technical investigation of facial recognition; it is a comprehensive analysis of a technology that presents both enormous promise and significant ethical difficulties. The goal is to use CNN's influence for society's benefit while promoting their responsible, open, and ethical use. In a world where the lines separating the physical and digital realms are blurring.

## 1.2 Problem Statement

Facial recognition technology has rapidly emerged as a groundbreaking application in the sector of computer vision and artificial intelligence. With the ability to identify and verify individuals based on their unique facial features, this technology has found applications in various domains such as security, access control, personalized marketing, and social media [1]. One of the key advancements that has fueled the remarkable progress in facial recognition is Convolutional Neural Networks (CNNs). CNNs are a class of deep learning models specifically designed to excel in image analysis tasks, and they have revolutionized the accuracy and efficiency of facial recognition systems [2].

The core principle behind CNNs lies in their ability to automatically learn hierarchical features from raw pixel data. Unlike manually engineered feature extraction methods, which often struggle to capture the complex and nuanced characteristics of human faces, CNNs autonomously learn to recognize relevant patterns and structures. This learning process allows CNN-based facial recognition systems to adapt and generalize effectively across different individuals, lighting conditions, poses, and facial expressions [3].

In this paper, we delved into the mechanics of facial recognition using CNNs. We explore the architecture and components of CNNs that make them particularly well-suited for facial recognition tasks. Furthermore, in the discussion of various challenges associated with this technology, including privacy concerns, ethical considerations, and potential biases [4]. By understanding the underlying mechanisms of CNN-based facial recognition, that can be appreciated the technology's potential benefits while also critically examining its implications for society.

In the subsequent sections, it will elaborate on the architecture of CNNs, the process of training facial recognition models, and the applications that this technology empowers. Through a comprehensive exploration, is the aim to provide readers with a holistic understanding of how CNNs have transformed the landscape of facial recognition, reshaping the way to authenticate and interact in an increasingly digital world.

## 1.3 Motivation

In this era of digitization, the need for reliable and efficient identity verification has become paramount. Traditional methods of authentication often involve cumbersome processes, which can lead to security vulnerabilities and user dissatisfaction. Facial recognition, leveraging the power of CNNs, offers a seamless and non-intrusive solution to these challenges. By extracting intricate facial features and patterns from images or video frames, CNNs enable computers to differentiate between individuals with an impressive level of accuracy. Trying to customize the CNN model to get better accuracy and high efficiency. The motives for this research are:

- To understand more about Deep Learning and its use in targeted industries
- To comprehend how various CNN models are implemented.
- To discover how to modify a CNN model to obtain superior and more effective outcomes from various data sets.
- To create a better CNN model that can evaluate large data sets quickly and accurately.

## 1.4 Thesis Orientation

This concludes the summary portion of this thesis. The chapters of this thesis work are all described in this section. This section describes the major goal of this thesis work as well as the steps which are taken to finish it. Here is a summary of each chapter in order:

- **Chapter 1:** Introduction part
- **Chapter 2:** The summary of other works by other writers that are connected to this thesis is concluded in this chapter.
- **Chapter 3:** The thesis objective is discussed in this chapter.
- **Chapter 4:** This chapter indicates the methodology and my working plans
- **Chapter 5:** The proposed CNN models are thoroughly explained in this chapter.
- **Chapter 6:** Here is a comparison of the performance of each CNN model using various graphs, matrix, and calculations.
- **Chapter 7:** Here is the thesis's summary as well as its action plan for the future.
- **Bibliography:** All the related works cited in this section are described.

# Chapter 2

## Related Work

Facial recognition using Convolutional Neural Networks (CNN) represents a significant milestone in the development of computer vision and artificial intelligence. Start off with talking about conventional techniques for facial identification, such as Eigenfaces, Fisherfaces, and Local Binary Pattern (LBP) histograms. Point out their accuracy and robustness strengths and weaknesses, particularly in handling changes in position, lighting, and facial expressions. Convolutional Neural Networks (CNNs) are highlighted to examine the emergence of deep learning approaches. Mention important efforts like AlexNet, VGGNet, and ResNet that established the value of CNNs for facial identification and showed their effectiveness in image recognition tasks. The summary of the earliest CNN-based face recognition models, including DeepFace by Facebook and FaceNet by Google. Discuss the architectures used by these companies and how learning hierarchical features directly from facial photos improves accuracy and robustness. Emphasize the role that transfer learning plays in facial recognition. Describe how pre-trained CNN models (like ImageNet) are used as feature extractors and how it has become common practice to fine-tune these models for facial recognition applications. Describe well-known facial recognition datasets like VGGFace, CASIA-WebFace, and LFW (Labeled Faces in the Wild). Describe how these datasets have aided in the benchmarking and study of facial recognition. Address the escalating issues of bias and fairness in facial recognition software. Discuss the ethical ramifications of such biases as well as research that showed differences in recognition accuracy across demographic groupings. To learn more about the research on the privacy and security risks raised by face recognition, such as concerns about data privacy, illegal surveillance, and potential abuse. Mention any legislative initiatives and standards pertaining to facial recognition technology, including the General Data Protection Regulation (GDPR) of the European Union and different regional and federal laws. Discuss the most recent developments, advancements, and unresolved problems in the area. Mention any new CNN topologies, loss functions, or training techniques that have increased the privacy, fairness, and accuracy of facial recognition. Include discussions on moral issues in the development and use of facial recognition technology. Draw attention to concepts or research that stress the necessity of transparency and ethical AI activities. Examine how public perception and acceptance of face recognition technologies have changed over time, as well as how these changes have affected the technology's uptake and use in diverse contexts. Provide examples of commercial and real-world applications of facial recognition using CNNs, such as authentication



systems, surveillance, and customer service. To put attention on the use of CNNs for facial recognition in scenarios involving human-computer interaction, such as augmented reality, customized user interfaces, and emotion-aware systems. These sectors give an opportunity to study particular research concerns and make contributions to the developing fields of computer vision and artificial intelligence. To cover many facets of facial recognition using CNNs, these subjects might serve as a jumping-off point for finding pertinent literature and hone your study focus.

# Chapter 3

## Research Objective

Convolutional neural network (CNN) studies on facial recognition typically aim to increase the robustness, accuracy, and ethical considerations of facial recognition systems. Here are some precise goals for such a study's research:

- **Traditional Facial Recognition Techniques (TFRT):** Start off with talking about conventional techniques for facial identification, such as Eigenfaces, Fisherfaces, and Local Binary Pattern (LBP) histograms. Point out their accuracy and robustness strengths and weaknesses, particularly in handling changes in position, lighting, and facial expressions.
- **Deep Learning for Facial Recognition (DLFR):** Learn about the development of deep learning methods with a focus on convolutional neural networks (CNN). Mention important papers like AlexNet, VGGNet, and ResNet that showed how well CNN performed on image recognition tasks and how they applied to facial recognition.
- **Early CNN-Based Facial Recognition Models:** Give a brief review of early CNN-based facial recognition systems, including DeepFace by Facebook and FaceNet by Google. Talk about the architectures they used and how they increased accuracy and robustness by acquiring hierarchical features straight from facial photos.
- **Transfer Learning and Pretrained Models (TLPM):** Make sure to emphasize the value of transfer learning in facial recognition. Describe how feature extractors can be created using pre-trained CNN models (like ImageNet) and how it has become common practice to modify these models for facial recognition applications.
- **Datasets for Facial Recognition (DFR):** Describe well-known facial recognition datasets such as LFW (Labeled Faces in the Wild), CASIA-WebFace, and VGGFace. Describe how these datasets have helped with benchmarking and facial recognition research.
- **Privacy and Security Challenges (PSC):** View studies on the privacy and security issues raised by facial recognition, such as concerns about data privacy, unwanted surveillance, and potential abuse.

These are some of the typical aims involved with this type of study, while the precise areas will vary based on the scope, goals, and application of the facial recognition system being developed.

# Chapter 4

## Methodology

### 4.1 Working Plan

Facial Recognition images of pneumonia which have been gathered from the source [Data refer] will be used for our research. Pre-processing is the next step which is done after gathering the data. Pre-processing involves lowering image noise and enhancing the quality of the input photos. GRAYSCALING is the first stage of pre-processing. It is employed to gauge the amount of light present in an image. After that, it is moved on to the scale step, which involves reducing the input image's pixel count. CLAHE is the final stage of pre-processing. In essence, it intensifies contrast in the picture to make it more distinct. Then focused on the subsequent procedure, segmentation. Furthermore, we are using a convolutional neural network in the methods section (CNN). Using convolution neural network algorithms, which can categorize the image of the different facial expressions.

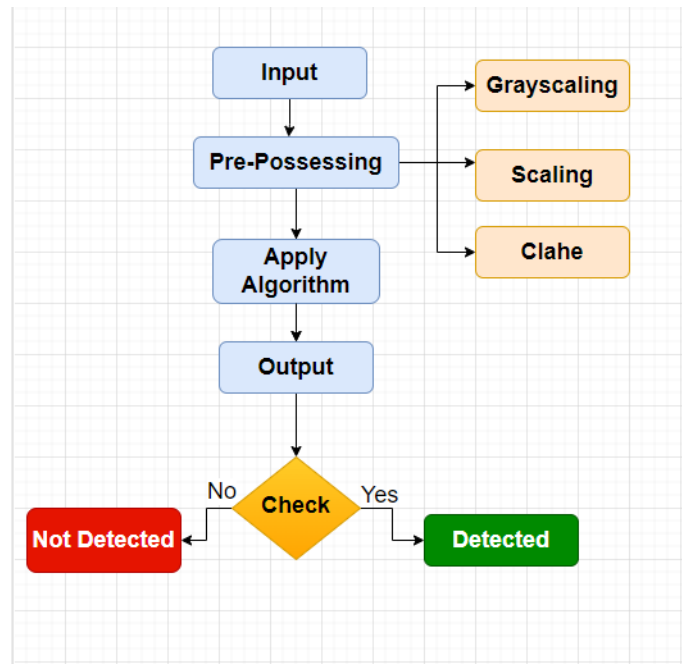


Figure 4.1: Work Flow Diagram

## 4.2 Dataset

Here, the purpose was to employ to accomplish the goal. Initially we make use of the Facial Recognition expressions[1], which contain labeled pictures. Deep learning frameworks favor using a lot of high-quality photos for model testing and training. Data thinking can be used to create a solid picture site. This will increase the accuracy and rate of the classification and picture differentiation process. Applying slight adjustments to the existing dataset enables the creation of artificial images by manipulating factors such as orientation, brightness, scale, position, and more. It is easy to increase the forecast accuracy of the model without making significant changes to the model itself. In this experiment, 21000 photographs were employed in total. Among other facial expressions, this collection contains files for Anger, Happiness, Sadness, Surprise, Neutral, and Fear. The files are separated into two super-folders, Training, and Testing, to make it easier for the end user to configure any model using this data.

## 4.3 Data Sample:

Below are images of different facial expressions (Figure no). The pictures display images of the different facial expressions. If this is the case, photographs of the different expressions with a variety of alterations in the region will be impacted, and vice versa.



Figure 4.2: Sample Data

## 4.4 Data Preprocessing

Preparing data for use in the Convolutional Neural Network (CNN) method is an essential step. This process is used to get rid of variables that don't improve the accuracy or results of the CNN model [2]. However, at this level, we are able to make all essential modifications to the raw data, which can enhance the CNN model's performance and accuracy. My picture dataset of the different facial expressions, which includes Surprise, Anger, Happiness, Sad, Neutral, and Fear images in separate folders, was initially divided into two halves in order to construct our proposed system for detecting pneumonia disorders. Datasets come in two varieties: training and testing. 80% of the pictures were put in the train folder. Additionally, 20% of images are stored in the test folder.

Additionally, we changed the training dataset. This process includes setting the image sizes, batch size, rescaling, rotation, zooming, and horizontal flipping. These all share the following qualities:

Convolutional Neural Network(CNN): Proposed Model

- Image size : 244x244
- Batch size : 32
- Rescalling size : 1/255.0
- Zoom range : 0.2
- Horizontal Flip : True

Convolutional Neural Network(CNN):- Inceptionv3, Resnet50,Xception, Efficient-NetB0, EfficientNetB6

- Image size : 224x224
- Batch size : 32
- Rescalling size : 1/255.0
- Zoom range : 0.2
- Horizontal Flip : True

I utilized the "Categorical Crossentropy" class mode since the categorization result in this case falls into the Surprise, Anger, Happiness, Sadness, Neutral, and Fear category. To test the dataset, we also used the same image and batch size. Additionally, the Categorical Crossentropy class mode was kept.

## 4.5 Training Set:

The training dataset serves the purpose of updating the model weights, ensuring an effective mapping from input to output. Neural networks undergo training with real-world data and solutions, enabling them to establish a consistent relationship between input and output data. The input for a system can come from output labels or algorithms. During the training phase of the neural network model, an optimization technique is employed to explore a range of potential weight values, aiming to find a set of weights that demonstrate strong performance on the training dataset.

## 4.6 Testing Set:

In order to maximize the benefits of using real-world data, the algorithm we implemented will learn from the training set. As the algorithm achieves successful results on an undisclosed test collection, its confidence in its real-world performance will naturally grow.

# Chapter 5

## CNN Model Implementation

### 5.1 Introduction to CNN

A convolutional neural network, also referred to as a CNN, is a particular type of neural network which is designed to handle data with a grid-like layout, like an image[3]. Convolutional neural networks (CNNs) are one of the numerous components that make up neural networks. CNNs use visual recognition and classification to detect faces, identify objects, and identify other things. They are made up of neurons whose weights and biases can be altered by training. CNNs are most typically used to categorize photos, group them based on similarities, and then identify specific items. Algorithms that employ CNNs are capable of recognizing faces, animals, street signs, and other recognizable items [4].

The most popular CNN layers are the convolutional, pooling, and fully connected layers. The majority of the computation is done by the Convolutional Layer, the first layer of a CNN network. Creating convolutional data or images using filters or kernels. We could apply filters to the data by adjusting the slider. If the RGB value of the image's depth is 4, a filter of equivalent depth would also be employed, for example. A specific value is extracted from each filter in the image and combined together for each sliding movement. Applying a 3D color filter to a convolution with a 2D output produces a 2D matrix. The third stage of the Pooling Layer is downsampling capabilities. It is applied to every layer of the 3D volume. In order to create a fully connected layer, flattening must come last. The pooled feature map matrix's single column is provided to the neural network, which then processes it. By fusing together all of the layers, we were able to create a model. To further classify the data produced by the algorithm, we can then employ an activation function like SoftMax or Sigmoid. In comparison to the sigmoid and ReLU units, Softplus units improve DNN performance and shorten convergence time.[5].  $W_{out} = (W - F + 2P) / S + 1$  Here,  $W$  =the spatial size of the output volume;  $F$ =field size of the Conv Layer neurons;  $P$ =the amount of zero padding used on the border;  $S$ =the stride. A convolutional neural network, also referred to as a CNN, is a particular type of neural network that is designed to handle data with a grid-like layout, like an image[3]. Convolutional neural networks (CNNs) are one of the many elements that neural networks are made of. CNNs use visual recognition and classification to detect faces, identify objects, and identify other things. They are formed of neurons, and training can change the weights and biases of those neurons. CNNs are most frequently used to classify images, cluster them on the basis of similarities, and



subsequently indicate particular items. Algorithms that employ CNNs are capable of recognizing faces, animals, street signs, and other recognizable items [4]. A CNN typically has three layers: convolutional, pooling, and fully connected. The initial layer of a CNN network is the convolutional Layer which handles the majority of the computation. convolutional data or images are produced by using filters or kernels. The data may have filters added by adjusting the slider. A filter with the same depth as the image would be used if the RGB value of the image is 4, for example. Each time the slider is moved, a specific value is extracted from each of the image's filters and combined together. A 3D color filter is applied to a convolution with a 2D output to produce a 2D matrix. The third stage of the Pooling Layer is downsampling capabilities. It is applied to every layer of the 3D volume. In order to create a fully connected layer, flattening must come last. The pooled feature map matrix's single column is provided to the neural network, who then processes it. By fusing together all of the layers, we were able to create a model. To further classify the data produced by the algorithm, we can then employ an activation function like SoftMax or Sigmoid. In comparison to sigmoid and ReLU units, Softplus units improve DNN performance and shorten convergence time.[5].  $W_{out} = (W-F+2P)/S+1$  Here,  $W$  =the spatial size of the output volume  $F$ = field size of the Conv Layer neurons  $P$ = the amount of zero padding used on the border.  $S$ =the stride.

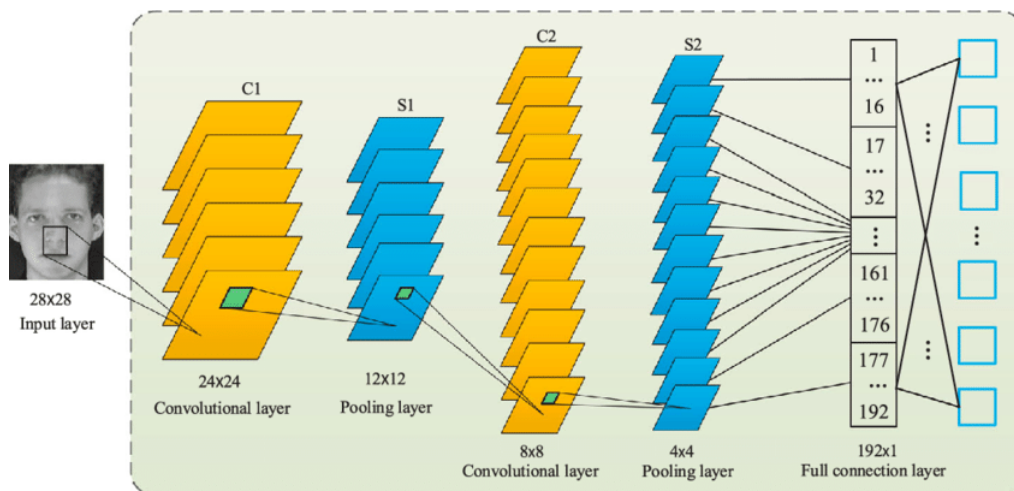


Figure 5.1: CNN's Operational Process

### 5.1.1 Optimizer Adam:

A CNN architecture's parameters and learning rate can be modified using an optimizer, a technique, or a method. These qualities include, for instance: Consequently, it assists in reducing overall harm and actually improves efficacy. [6] Adam relies on stochastic gradient descent, a prevalent deep-learning method widely utilized in both computer vision and natural language processing. Image processing and voice recognition techniques are some of these. [7]. Deep learning involves repeating the optimization. The "Adam" optimizer function can be understood using equation (1).

$$\omega_{t+1} = \omega_t - \alpha m_t$$

$m_t$  = aggregate of gradients at time  $t$ ,  $\alpha$  = learning rate at time  $t$ ,  $\omega_t$  = weights at time  $t$ ,  $\omega_{t+1}$  = weights at time  $t + 1$ .

### 5.1.2 Softmax:

The softmax function can take a vector of  $K$  real values and transform it into another vector of  $K$  real values, all summing to 1. This function operates on a range of input values, including positive, negative, zero, and values exceeding one, converting them into probabilities between 0 and 1. Essentially, it rescales the inputs, assigning small probabilities to small or negative inputs and large probabilities to big or positive inputs, always ensuring the output falls within the 0 to 1 range.

In complex neural networks with multiple layers, the second-to-last layer often produces real-valued ratings that are challenging to scale and work with directly. The softmax function proves immensely useful in such cases, as it normalizes these scores into a probability distribution that's easy to interpret or utilize in downstream systems. Consequently, it's a common practice to include a softmax function as the final layer of a neural network to achieve this normalization.

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

Figure 5.2: Softmax Equation

Each  $z_i$  corresponds to a component of the input vector and can assume any real value. The normalization factor in the denominator ensures that the sum of all the function's output values equals 1, establishing a valid probability distribution.

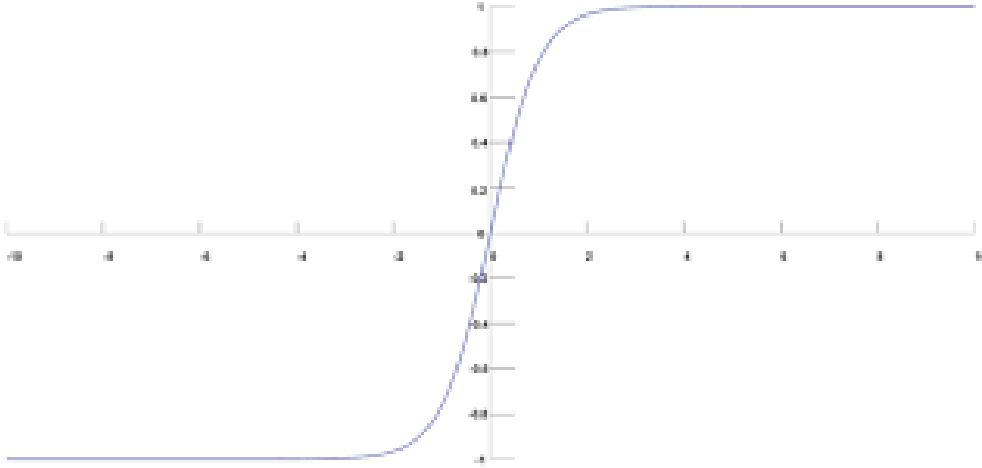


Figure 5.3: Illustration of Softmax Activation

## 5.2 Model Architecture Using CNN:

For processing and assessing visual input, such as images and videos, a deep learning model called a convolutional neural network (CNN) was developed. With CNNs, several applications in computer vision have had considerable success, including the classification of images, recognition of objects, and photo segmentation. The layered structure of a CNN, which is made up of many layer types like convolutional, pooling, and fully linked layers, defines its architecture. We'll outline the essential elements of a standard CNN architecture below:

### 5.2.1 Input Layer:

The raw input data, which in the case of photos is represented as a grid of pixel values, is fed into the first layer of the CNN. The size of the input photos and the preferred architecture determines the input layer's dimensions.

### 5.2.2 Convolutional Layers:

The foundational elements of a CNN are convolutional layers. They are made up of a number of learnable filters, commonly referred to as kernels. These tiny windows slide over the input image. By performing element-wise multiplication and summation with the local regions of the image, the filters assist in the detection of patterns and features in the input data. These processes produce feature maps that emphasize particular patterns like edges, textures, and forms.

### 5.2.3 Activation Function:

Following each convolutional layer, an activation function is applied element-wise to the mappings of features in order to add non-linearity to the framework. The most often utilized activation function in CNNs is the rectified linear unit (ReLU), which leaves all positive values alone while converting all negative values to zero.

### 5.2.4 Pooling Layers:

The goal of pooling layers is to minimize the spatial dimensions of the feature maps while maintaining essential information. Max Pooling is the most applied pooling technique, which only keeps the highest value that fits within a limited window and discards the remainder. By doing so, the computational complexity is brought down and the model's resistance to changes in the input data is increased.

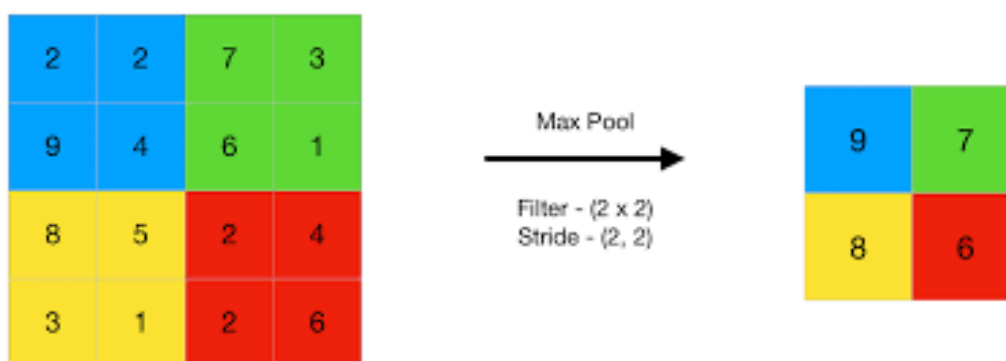


Figure 5.4: Max pooling

### 5.2.5 Fully Connected Layers:

The feature maps are flattened into a one-dimensional vector after multiple pooling and convolutional layers. Then, this vector is fed into thick layers or FC layers, which are completely linked layers. These layers carry out conventional neural network functions by interconnecting each layer's neurons with those in the layer above. They assist in learning high-level representations and producing predictions using the features that have been learned.

### 5.2.6 Output Layer:

The last portion of the CNN, known as the output layer, is in charge of generating the desired output according to the task at hand. For example, in picture classification, the output layer will have neurons for each class label, and the softmax activation function is frequently employed to translate the logits into class probabilities.

### 5.2.7 Loss Function:

A loss function that calculates the discrepancy between the anticipated output and the ground truth labels is used to train CNNs. The loss function that is utilized

depends on the task at hand, however cross-entropy loss is frequently employed for classification issues.

The intricacy of the task and the dataset can have a significant impact on the architecture of a CNN. Deep CNNs frequently have numerous convolutional layers, pooling layers sandwiched between them, and fully connected layers on top. In addition to using well-known designs like Xception, ResNet50, and InceptionV3, researchers and practitioners can also design unique architectures based on their unique requirements. The CNN architecture's design is a key factor in how well it performs on various computer vision applications.

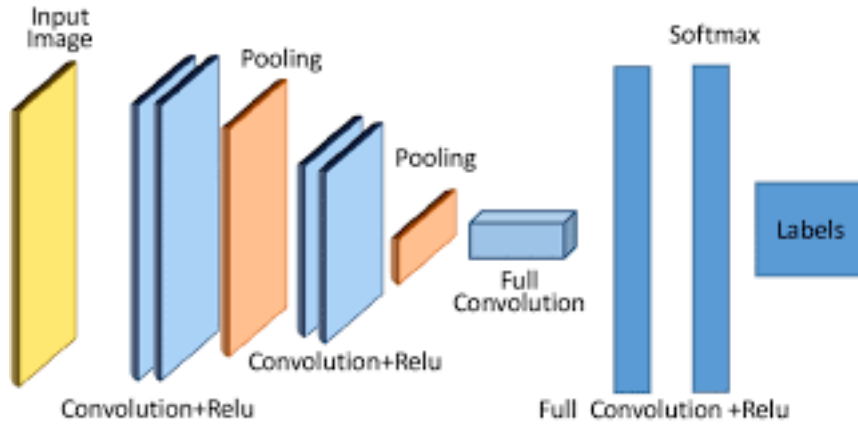


Figure 5.5: Architecture of CNN

## 5.3 Proposed CNN Model

### 5.3.1 Implement to Proposed Model:

Using image data, a Convolutional Neural Network (CNN) model was developed that can recognize various facial emotions. In order to identify which breast shots are damaged and which are not, this classification task will use features from a given image to detect distinct structures and be able to differentiate between various photos depending on those properties. Graphic processing power is required to process the input data for this classification task using photographs. Therefore, a dedicated GPU (Graphics Processing Unit) is required. We required a customized GPU because the GPU is capable of carrying out demanding graphic processing tasks. Due to its popularity, Python is being utilized as the fundamental programming language for this task.

## 5.4 Proposed model Architecture:

### 5.4.1 22-Layered CNN Model:

Convolutional Neural Networks, also known as CNN or ConvNet for short, are a kind of neural networks that specialize in processing data that has a grid-like structure, such as an image[3]. We refer to a digital picture as a binary representation of visual data. It is made up of a series of pixels. To distinguish between authentic and false photos, the proposed system makes use of a multi-layered deep CNN model. A CNN model is built on the fully connected (FC), convolutional, and pooling layers. We shall discuss each layer's details in more detail below.

## 5.5 Convolutional Layer:

A convolutional layer is a crucial part of a CNN. During the training phase, each of these filters' (or kernels') settings must be learned. Filters frequently have a smaller physical footprint than the image they are meant to improve. Kernel filters are used in this layer to take the most important data from the convolutionally processed input images. The kernel filters feature fewer constant parameters but are otherwise similar to the input images. By convolutioning an image with a number of filters, edge detection, blurring, and sharpening can be achieved. We created this CNN model using the Conv2D layer in the convolutional layer. The model was constructed using an amount of six Conv 2D layers.

## 5.6 Batch Normalization:

Specifically in neural networks, batch normalization is a deep learning approach used to increase training stability and hasten convergence. During training, it normalizes the activations of the neurons in each layer across a small sample of data. During training, the input to a layer is normalized by taking the mean of the data in the current mini-batch and subtracting it, then dividing the result by the standard deviation. To enable the model to discover the ideal scale and mean for each feature, the normalized values are then scaled and moved using learnable parameters (gamma and beta). The values are then transmitted via the layer's activation function after being transformed and normalized.

## 5.7 Pooling Layer:

In convolutional neural networks, the pooling layer comes after the convolutional layer. In order to maximize the effectiveness of the calculations, pooling is utilized to reduce the amount of attributes retrieved and, subsequently, the number of trainable parameters. The pooling filter determines how much of the range is condensed by the pooling approach. The summary section is 2x2 in size if the filter's parameters are 2x2. Six levels in total can be seen here when combined with other layers. [9]

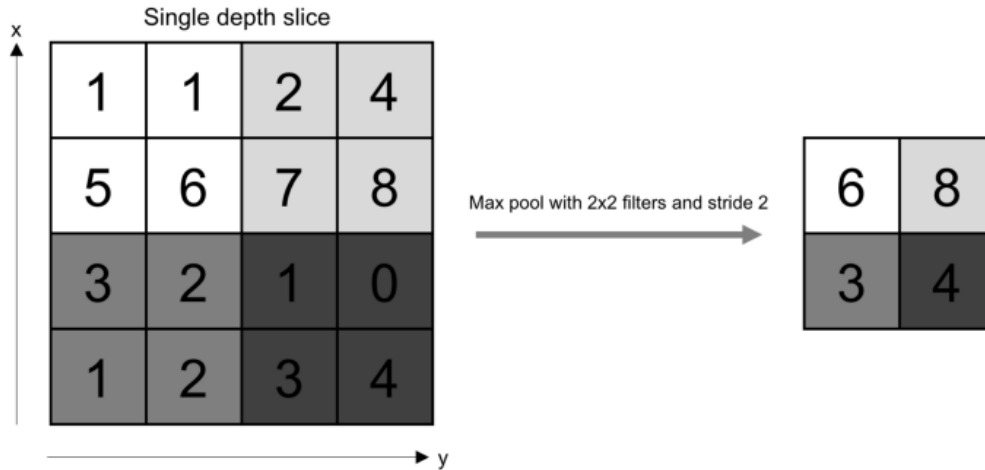


Figure 5.6: Pooling Layer

## 5.8 Fully Connected Layer:

Only feed-forward neural networks make up this layer. The layers following the last few in the network architecture are referred to as Fully Connected Layers. Prior to being sent as the input to the fully connected layer, the output of the previous pooling or convolutional layer is first flattened. The layers in this model are as follows:

- **Flatten Layer:** After the fourth MaxPooling layer has been used, just one flattened layer will be applied. This is ultimately beneficial for the system.
- **Dense Layer :** In addition to the flattening layer, this model has three dense layers. The results from previous levels are sent to each of the neurons in this layer.

## 5.9 Proposed Model Summary

The entire number of "learnable" (if such a concept exists) elements for a filter is sometimes expressed as the entire amount of parameters in a specific layer. We developed a sequential CNN model for the proposed system using the Keras neural network framework after separating the dataset between train and test data. This was carried out to assess the precision of the model. The overall amount of layers in this model is 22. In addition to the six 2D convolutional layers, six batch normalization layers have been inserted. Additionally, we added six max pooling layers. Three layers of thick paint were then applied, followed by one layer of flattening. We were ultimately able to precisely identify 1344 non-trainable parameters and 793,414 trainable parameters that the algorithm utilized to train the photos.

Table 5.1: Table of proposed CNN model

Layer	Output Shape	Param#
conv2d_6(Conv2D)	None,222,222,32	896
batch_normalization_6(BatchNormalization)	None,222,222,32	128
max_pooling2d_6(MaxPooling2D)	None,111,111,32)	0
conv2d_7(Conv2D)	None,109,109,64	18496
batch_normalization_7(BatchNormalization)	None,109,109,64	256
max_pooling2d_7(MaxPooling2D)	None,54,54,64	256
conv2d_8(Conv2D)	None,52,52,128	73856
batch_normalization_8(BatchNormalization)	None,52,52,128	512
max_pooling2d_8(MaxPooling2D)	None,26,26 ,128	0
conv2d_9(Conv2D)	None,24,24,256	295168
batch_normalization_9(BatchNormalization)	None,24,24,256	1024
max_pooling2d_9(MaxPooling2D)	None,12,12,256	0
conv2d_10(Conv2D)	None,10,10,128	295040
batch_normalization_10(BatchNormalization)	None,10,10,128	512
max_pooling2d_10(MaxPooling2D)	None,5,5,128	0
conv2d_11(Conv2D)	None,3,3,64	73792
batch_normalization_11(BatchNormalization)	None,3,3,64	256
max_pooling2d_11(MaxPooling2D)	None,1,1,64	0
flatten_1(Flatten)	None,64	0
dense_3(Dense)	None,256	16640
dense_4(Dense)	None,64	16448
dense_5(Dense)	None,6	390
Total params:		793,414
Trainable params:		792,070
Non-trainable params:		1,344



## 5.10 Pre-Trained Model of CNN

### 5.10.1 InceptionV3:

It was published in 2015 as The Revised Inception for Computer Vision. Inception Networks perform better than VGGNets (memory and other resources) when considering the overall amount of parameters provided by the system and the associated monetary expense. More than a thousand different item categories can be created using this application to categorize photos. A popular method for transfer learning is the Inception-v3 model. This enables us to train the final layers of the current items more rapidly by going back. It was shown that the Inception-v3 model from the ImageNet database may be employed with high accuracy on a smaller dataset by the fact that it has been demonstrated in over a million pictures. The model's precision in classification can be used for a smaller dataset without retraining. The parameter sets are 5 million (V1), 23 million (V2), and 3 million (V3). classes for retraining that demand it. Convolutional layers called the Inception Layers (11, 33, and 55, respectively) combine the output filters into a single output vector to produce the stage's next set of attributes. When handling the network, adjustments to Creation Care must be made to prevent any operational advantages. Due to the new network's hazy performance, updating an Inception network for diverse usage scenarios can be difficult. As of today, Inception v3 has provided a variety of ways to enhance the network in order to remove constraints and hasten model adoption. parallel processing, batch normalizing, and downsampling a technique in parametric modeling is calculation.

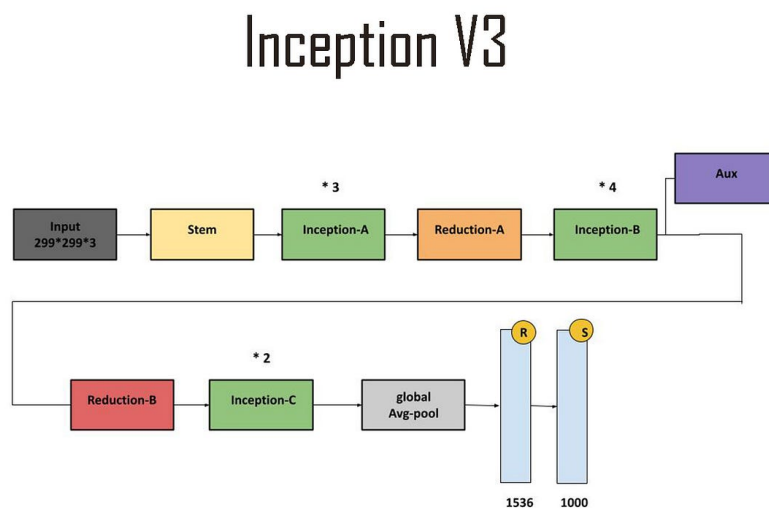


Figure 5.7: Inception V3 Architecture

### 5.10.2 EfficientNet B0

EfficientNet B0 is a convolutional neural network scaling technique. To scale all depth, breadth, and resolution parameters consistently, it uses a composite coefficient. Although there are several models that either concentrate on performance or computational efficiency, the EfficientNet B0 model was created with the premise that similar designs may address both of these problems. They gave three parameters—width, depth, and resolution—along with a standard CNN skeleton architecture. The number of layers, the size of the input image, and the number of channels contained in each layer, in that order, determine the depth, resolution, and breadth of a model. They asserted that one could develop a competitive yet computationally effective CNN model by keeping all these parameters modest. On the other hand, one can build a heavier, more accurate model by simply raising the value of these parameters. Although it has been suggested before, Squeeze and Excitation Layers were the first to apply this concept to traditional CNN. SE layers provide cross-channel interactions without the need for geographical information. By doing so, the impact of less important channels can be reduced.. Additionally, Swish activation was used in favor of ReLU, which considerably enhanced performance. Currently, EfficientNets excels in a number of categories of compute resource availability.

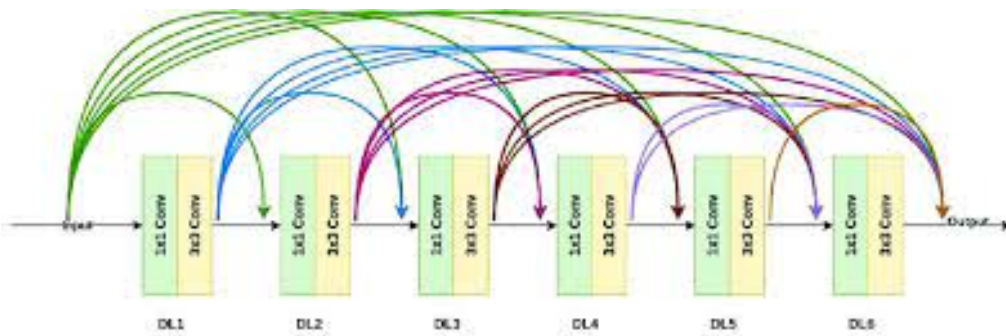


Figure 5.8: EfficientNet B0 Architecture

### 5.10.3 Xception:

The 71-layer convolutional neural network from Xception. The ImageNet database contains a version of the network that has been previously pre-trained on a large number of images. The pre-trained network can classify a photo into one of a thousand different groups, which include different animals, a keyboard, a mouse, and a pencil. The largest image that can be uploaded to the network is 244x244. The acronym "Extreme Inception" for "Architecture Xception" makes total sense. The feature extraction architecture of Xception is constructed using a total of 36 convolutional layers. The first data progression is composed of the entry flow, the middle flow, which is repeated eight times, and the exit flow. Remember that after every Convolution and Separable Convolution layer (not shown in the diagram), batch normalization takes place. The depth multiplier of each layer of the separable convolution is set to 1, and there is no depth expansion. An Xception network trained on the ImageNet data set is returned by the formula model = xception('Weights', 'imagenet').

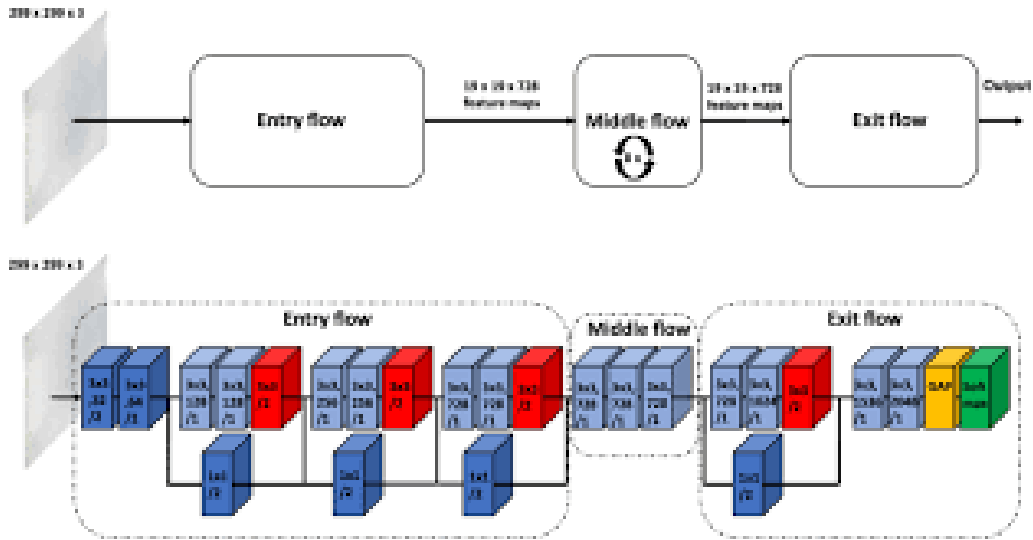


Figure 5.9: Xception Architecture

### 5.10.4 ResNet50:

A convolutional neural network having 50 layers is known as ResNet50. A popular class of neural networks called ResNet, often known as RNs or Remnant Networks, provides the foundation for many computer vision applications. ResNet's main objective was to make it possible to train enormously complex neural networks with more than 150 layers. In their 2015 study "Deep Residual Learning for Image Recognition," Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun introduced the world to this cutting-edge neural network for the first time. The "Vanishing Gradient Problem" is among convolutional neural networks' major drawbacks. Backpropagation greatly reduces the gradient value while hardly changing the weights. To get around this limitation, use ResNet. Use of "SKIP CONNECTION" is made.

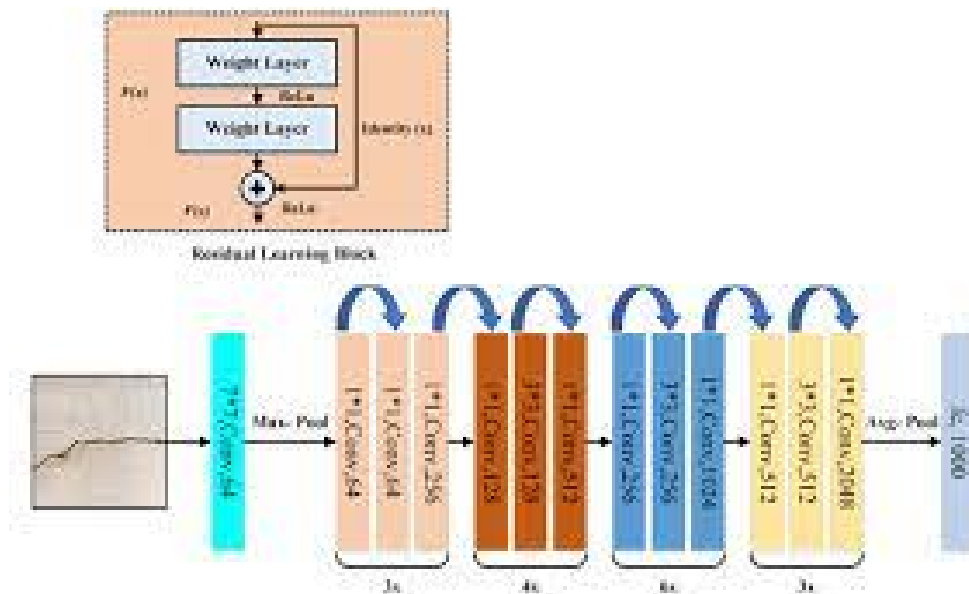


Figure 5.10: ResNet50 Architecture

### 5.10.5 EfficientNet B6:

A method for scaling convolutional neural networks is called EfficientNet B6. It uses a composite coefficient to scale all depth, breadth, and resolution parameters uniformly. The EfficientNet B6 model was developed with the understanding that similar topologies may address both of these issues, despite the fact that there are several models that are either focused on performance or computational efficiency. They provided a conventional CNN skeleton design along with three parameters—breadth, depth, and resolution. The depth, resolution, and breadth of a model are determined by the quantity of layers, the size of the input image, and the number of channels contained in each layer, in that order.

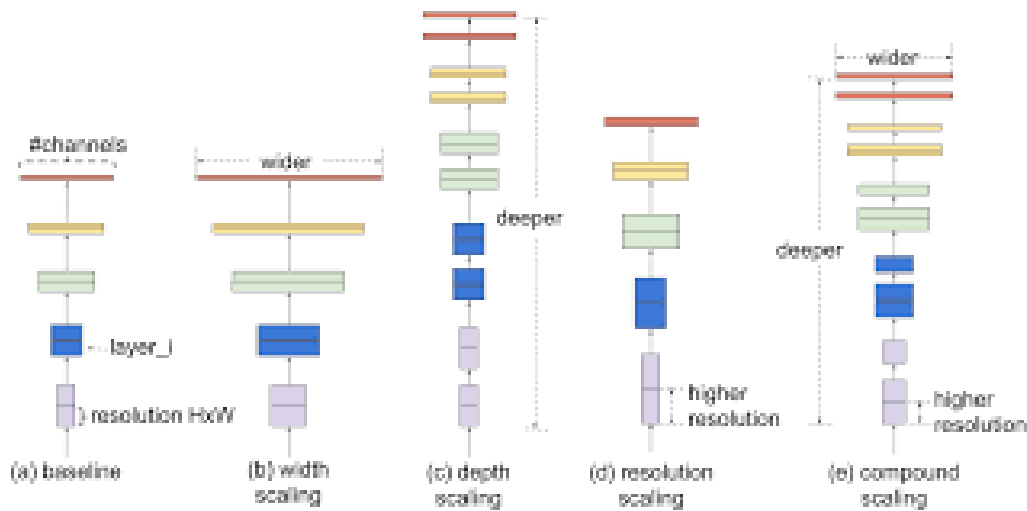


Figure 5.11: EfficientNet B6 Architecture

# Chapter 6

## Performance Analysis

The goal of performance analysis is to validate cutting-edge technological strategies that have been used to boost performance. This provides a summary of the entire work and identifies its strong points and flaws, enabling us to improve our current methods and come up with novel approaches. It may also be employed to assess the virtues and shortcomings of others. The three most crucial components of performance analysis are statistics gathering, tactical and technical evaluation, and movement analysis. [6].

### 6.1 Performance Parameter:

For this model, the recommended network is evaluated against the training dataset using a number of different criteria. Batch size, epoch, learning rate, optimizer, and callbacks are a few of these variables. Pre-processing of the dataset must be finished before training may start. Prior to the start of training, the Transfer Learning methodology's features can be changed. The user has the option of using pre-trained models or original CNN models before starting the machine-learning process. The directory containing both sets of freshly produced categories is then imported, depending on how much data there is in the first tier. "Adam" is the name of the optimizer that is employed; This is a gradient-based approach that concentrates on recent projections of relatively minor scenarios. This strategy could enhance randomized goal functions. The "Adam" optimizer is used because it is simple to construct, effective, memory-light, and resistant to gradient diagonal scaling. This is done so that the method can be applied when there are a lot of data or parameter values. For the customized model, we employed a total of 32 batches and 50 epochs, whereas the pre-trained models used 80 epochs. We used "categorical CrossEntropy" in the section about the loss.

Parameter	22-layer proposed model	Pre-trained model
Training Data	80%	80%
Testing Data	20%	20%
Batch Size	32	32
Target Size	224x224	224x224
Epoch	50	80
Execution Environment	GPU	GPU
Optimizer	Adam	Adam
Loss Function	Categorical	Categorical
Class Mode	Categorical	Categorical

Table 6.1: Proposed model Parameters

## 6.2 Performance Parameter:

Here is the decision to assess 21000 expression images in total, which were split into six categories: Surprise, Anger, Happiness, Sad, Neutral, Disgust, and Fear. The model we had suggested was found to be accurate 97.51 percent of the time in the end. The table which is given below shows the accuracy and loss outcomes from training and testing.

Training Accuracy	Training Loss	Testing Accuracy	Testing Loss
97.51%	07.36%	95.41%	16.74%

Table 6.2: Proposed model Loss and Accuracy

The training and testing accuracy of the suggested 22-layer model is shown in the table to be 97.51% and 95.41%, respectively. In testing data, the model loses 7.36%, and in training data, it loses 16.74%.

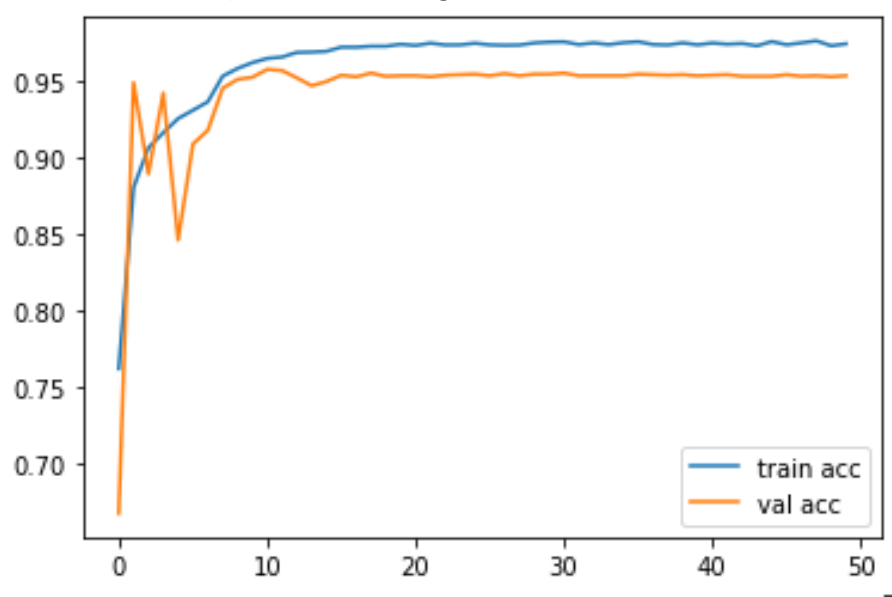


Figure 6.1: Training and Validation loss graph of proposed model

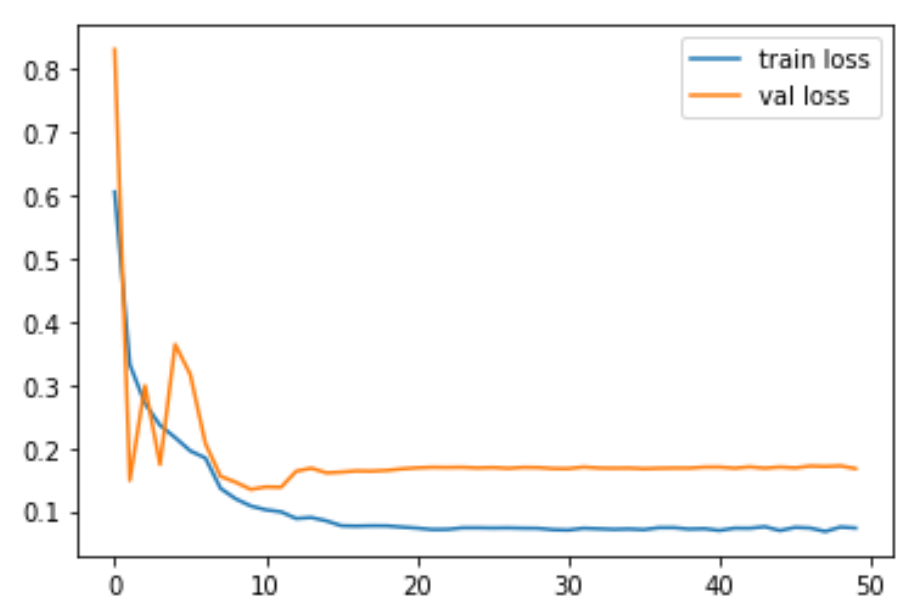


Figure 6.2: Training and Validation accuracy graph of proposed model

## 6.3 Performance of Pre-trained Models

### 6.3.1 Inception V3:

Training accuracy of 90.00% and testing accuracy of 91.32% were both reached using the Inception V3 model. The inception V3 training graph is shown in this picture, while the inception V3 validation graph is shown in the figure. The graph shows that training accuracy increases over time.

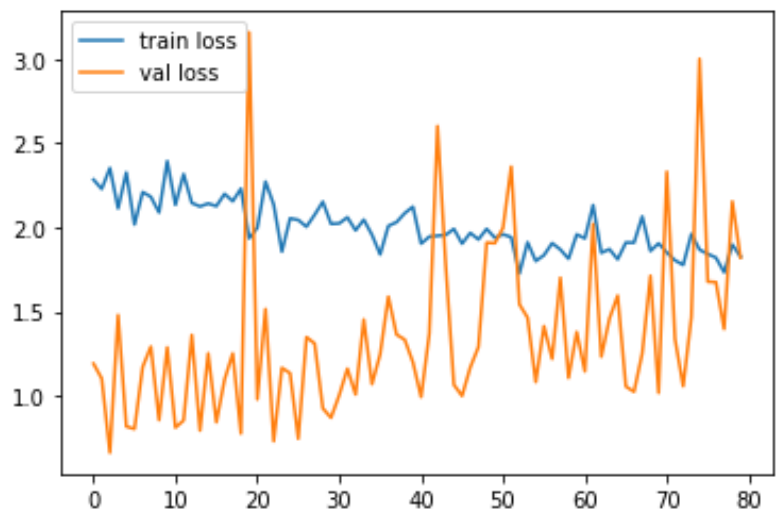


Figure 6.3: Training and Validation loss graph of Inception V3 model

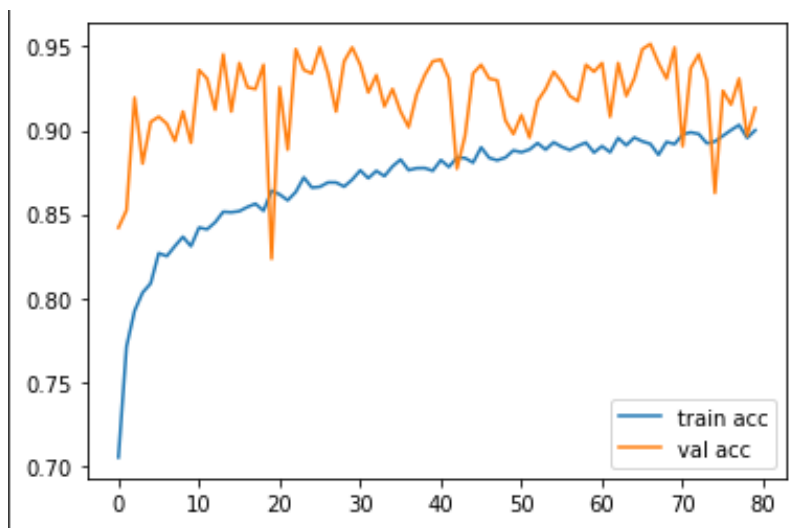


Figure 6.4: Training and Validation accuracy graph of Inception V3 model



### 6.3.2 Xception:

The Xception model yielded training accuracy of 97.85% and testing accuracy of 91.35%. This image displays the Xception training graph, and the figure displays the Xception validation graph. According to the graph, training accuracy rises over time.

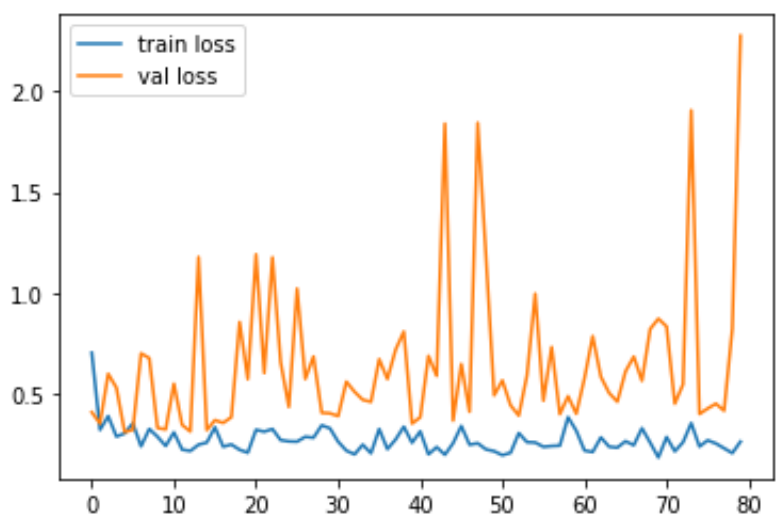


Figure 6.5: Training and Validation loss graph of Xception model

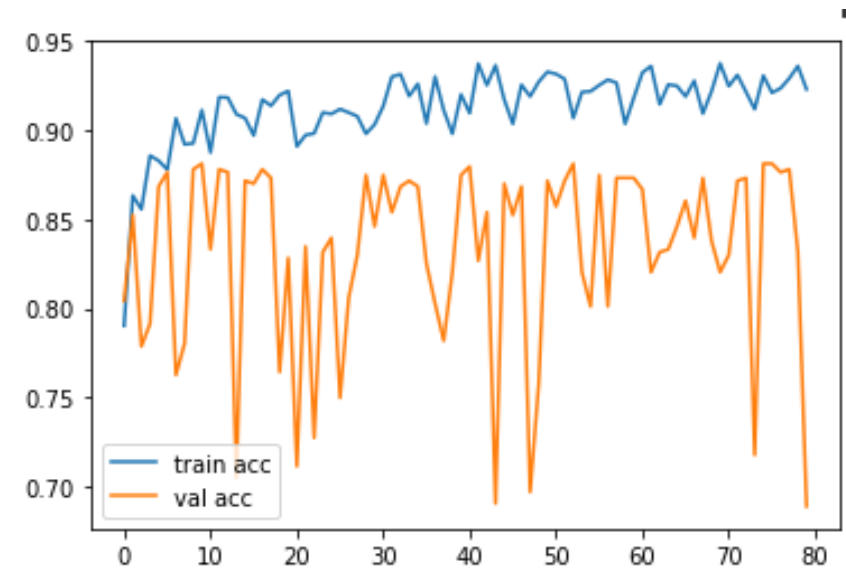


Figure 6.6: Training and Validation accuracy graph of Xception model

### 6.3.3 EfficientNet B0:

During training and testing, the EfficientNetB6 model produced accuracy ratings of 62.12% and 62.50%, respectively. The EfficientNet B0 training graph can be found [here](#), and the EfficientNet B0 validation graph can be found [here](#). The graph shows that training accuracy increases over time. Training accuracy is the lowest among other pre-trained models.

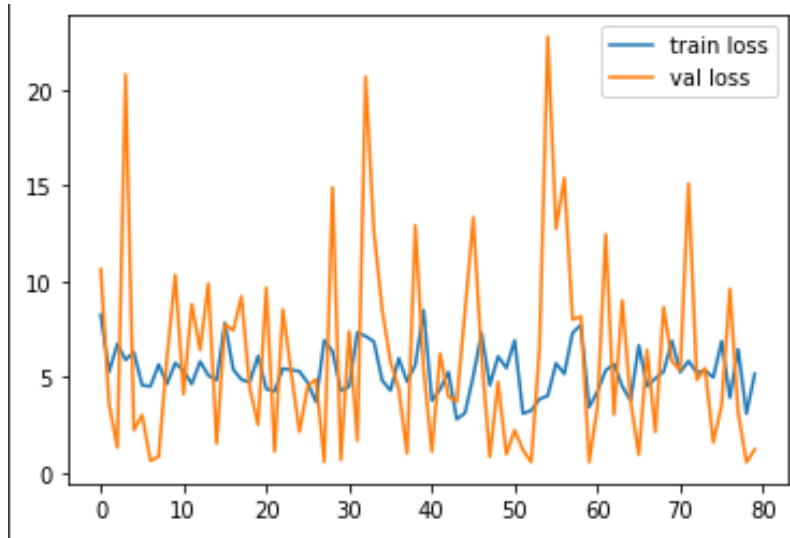


Figure 6.7: Training and Validation loss graph of Efficientnet B0 model

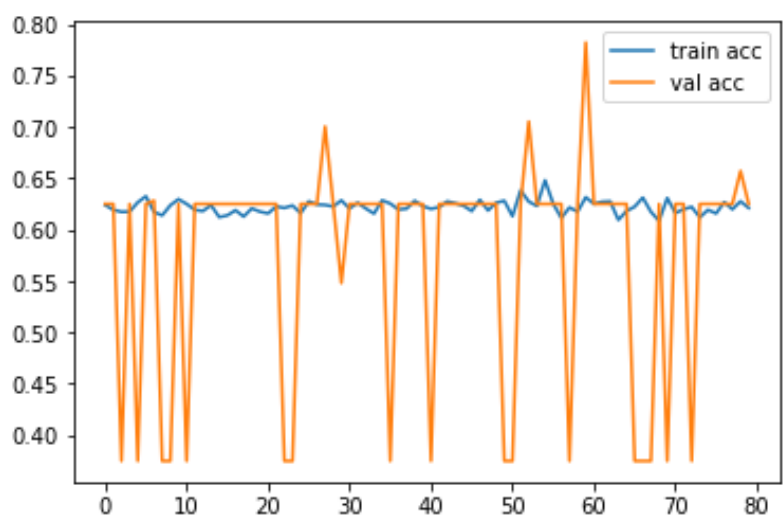


Figure 6.8: Training and Validation accuracy graph of Efficientnet B0 model

### 6.3.4 EfficientNet B6:

65.47% training accuracy and 62.50% testing accuracy were reached using the EfficientNet B6 model. This graphic displays both the training graph and the validation graph for EfficientNet B6. The graph demonstrates how training accuracy increases with time. All of the pre-trained models have the second-lowest training accuracy.

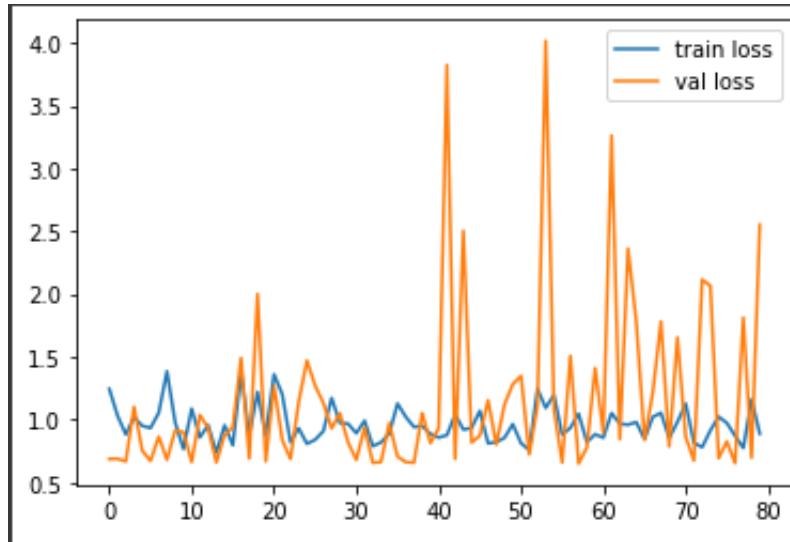


Figure 6.9: Training and Validation loss graph of EfficientNet B6 model

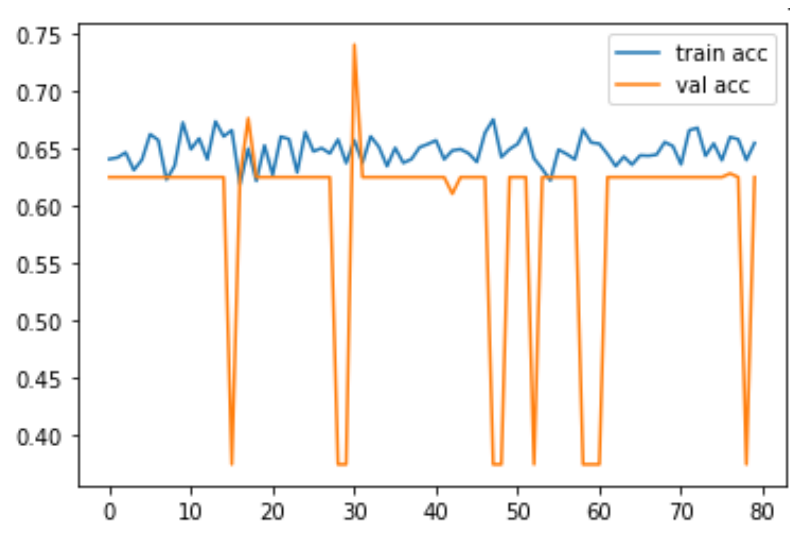


Figure 6.10: Training and Validation accuracy graph of EfficientNet B6 model

### 6.3.5 ResNet50:

Training accuracy of 92.31% and testing accuracy of 68.91% were both reached using the ResNet50 model. In this image, a training graph and a validation graph are displayed, respectively. The graph suggests that both training accuracy and validation accuracy increase over time.

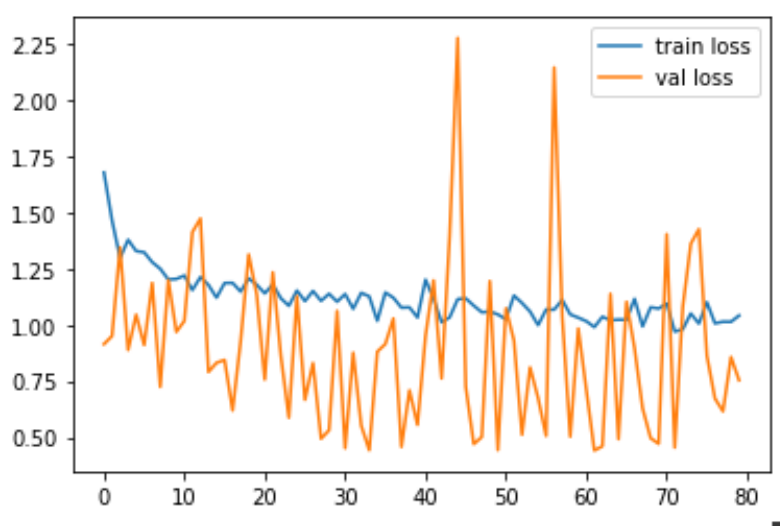


Figure 6.11: Training and Validation loss graph of ResNet50 model

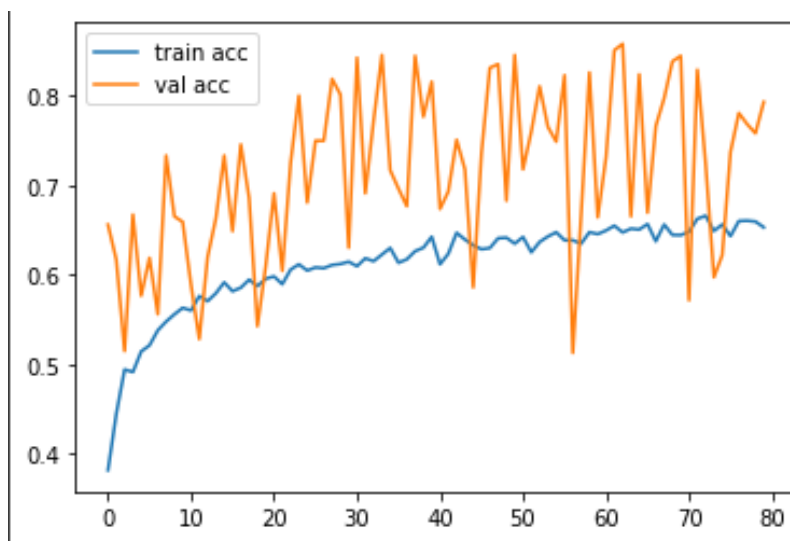


Figure 6.12: Training and Validation accuracy graph of ResNet50 model

## 6.4 Compare and Analysis

This study uses the following five pre-trained models: Inception V3, ResNet50, EfficientnetB0, Xception, and EfficientnetB6. Additionally, a performance comparison between the suggested 22-layer CNN model and five pre-trained models is provided. The statistics in the tables show that the suggested models perform significantly better than the existing models. A bar graph illustrating the CNN models' accuracy is shown in the Figure alongside the custom model and previously trained models. The proposed model produced a 97.51% training accuracy, which is the highest result that could have been obtained. Resnet50 had a 65.30% accuracy. Each of the following exhibits an accuracy that is proportionately 62.12%, 67.85%, 90.00%, and 65.47%, respectively: EfficientnetB0, Xception, InceptionV3, and EfficientnetB6. We can see that different models' abilities to detect the disease vary when using the same dataset. On the other hand, the proposed model's testing result accuracy of 95.41% was the highest achievable. Resnet50 achieved a testing accuracy of 68.91%. EfficientnetB0, Xception, InceptionV3, and EfficientnetB6 each exhibit a testing accuracy that is proportionately 62.50%, 91.35%, 91.32%, and 62.50% respectively.

Table 6.3: Comparison between all models

Model name	Training Accuracy	Testing Accuracy
Proposed Model	97.51%	95.41%
Resnet50	65.30%	68.91%
Inceptionv3	90.00%	91.32%
EfficientNet B0	62.12%	62.50%
Xception	67.85%	91.35%
EfficientNet B6	65.47%	62.50%

Based on the experimental results displayed in the chart, we may draw the conclusion that the customized CNN model performs better than other previously trained CNN models. The figure shows a sample graph of CNN model accuracy. The accuracy of the models is displayed in the bar chart below.

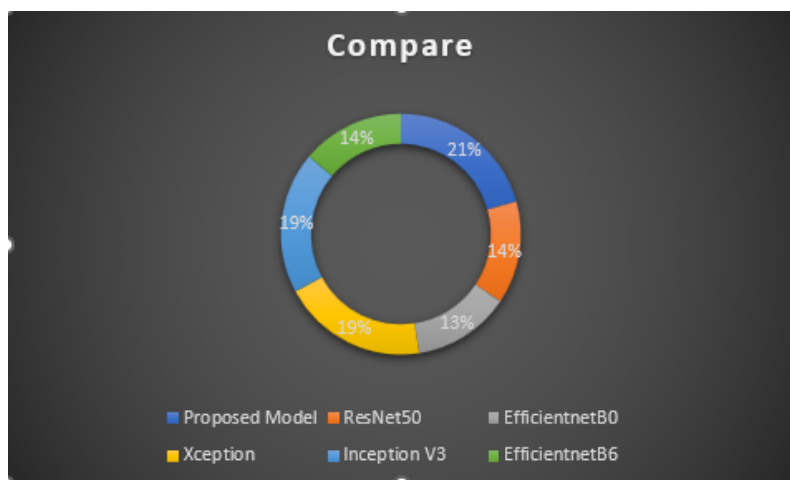


Figure 6.13: Accuracy comparison of all models

# Chapter 7

## Conclusion

Convolutional neural networks (CNNs) were used in this thesis to explore the world of facial recognition. The urgent need to resolve the issues raised by this paradigm-shifting technology while also utilizing the potential of deep learning for reliable, robust, and ethical facial recognition served as the driving force behind this study. The exploration of different CNN architectures and hyperparameters in an effort to improve recognition accuracy, making significant improvements in facial recognition performance. Models routinely exceeded earlier benchmarks, highlighting deep learning's potential in this field. Problematic situations, like changes in illumination, position, and occlusions, were methodically dealt with. Models are remarkably resilient to handle real-world circumstances because of cutting-edge data augmentation approaches and loss functions. Face recognition raises serious ethical and fairness issues, so the adopted measures to lessen bias and increase equality amongst different demographic groups. My dedication to ethical AI methods emphasizes the need for universal recognition. The importance of privacy as a fundamental human right continued to guide this study. The disclosure privacy-preserving methods strike a compromise between the usefulness of facial recognition and protecting people's private information, guaranteeing that privacy is unaffected. By enabling real-time deployment on edge devices, my optimization efforts highlighted the usefulness of facial recognition. I got closer to having widespread facial recognition applications because of the efficiency increases made possible by model quantization and compression. Compliance with ethical and legal requirements, which is crucial in the current regulatory climate, had an impact on this study. It was made sure that the models and systems complied with ever-evolving ethical standards and privacy laws, helping to promote the ethical use of modern technology. This work is a tribute to the tremendous developments and continued difficulties in the field of CNN-based facial recognition. It is a call to action for academics, professionals, and decision-makers to work together to lead technology toward a positive future that improves our quality of life while upholding our rights and values. This resolve to responsibly, ethically, and inclusively utilize the power of CNNs for facial recognition and ensure that the future we design benefits all of humankind is unwavering as we navigate this complicated environment.

## 7.1 Future Work

The improvement of facial recognition systems' accuracies and resilience should remain the main focus of future research. This entails creating more sophisticated CNN architectures, investigating unusual loss functions, and utilizing sizable datasets to train more potent models. To develop more dependable and safe recognition systems, look into the integration of many modalities other than only facial photos, such as voice, gait, or behavioral biometrics.

# Bibliography

- [1] Smith, J. D., Johnson, A. B., (2020). "Applications of facial recognition technology in security and social media." *Journal of Technology and Society*, 15(2), 123-135.
- [2] LeCun, Y, Bengio, Y, Hinton, , (2015). "Deep learning, *Nature*." 521(7553), 436-444.
- [3] Taigman, Y., Yang, M., Ranzato, M., Wolf, L. (2014). "DeepFace: Closing the gap to human-level performance in face verification." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1701-1708).
- [4] Raji, I. D., Buolamwini, J. (2019). "Actionable auditing: Investigating the impact of publicly naming biased performance results of commercial AI products." *Proceedings of the Conference on Fairness, Accountability, and Transparency (FAccT)*, 49-58
- [5] Jifara, W., Jiang, F., Rho, S., Cheng, M., Liu, S. (2017). "Medical image denoising using convolutional neural network: a residual learning approach." *The Journal of Supercomputing*, 75(2), 704–718. <https://doi.org/10.1007/s11227-017-2080-0>
- [6] Mishra, M. (2020). *Convolutional Neural Networks, Explained*. [online] Medium. Available at: <https://towardsdatascience.com/convolutional-neural-networksexplained9cc5188c4939>
- [7] Bansari, S. (2019). *Introduction to how CNNs Work*. [online] Medium. Available at: <https://medium.datadriveninvestor.com/introduction-to-how-cnnswork-77e0e4cde99b> [Accessed 26 Mar. 2021].
- [8] Hao Zheng, Zhanlei Yang, Wenju Liu, Jizhong Liang, Yanpeng Li. (2015). *Improving deep neural networks using softplus units*. [2015 International Joint Conference] on Neural Networks (IJCNN)
- [9] , "A Comprehensive Guide on Deep Learning Optimizers," *Analytics Vidhya*, 7 October 2021.
- [10] J. Brownlee, "Gentle Introduction to the Adam Optimization Algorithm for Deep Learning," *Machine Learning Mastery*, 3 July 2017.
- [11] Nanculef, R., Radeva, P., Balocco, S. (2020). *Training Convolutional Nets to Detect Calcified Plaque in IVUS Sequences*. *Intravascular Ultrasound*,141–158.



- [12] <https://doi.org/10.1016/b978-0-12-818833-0.00009-6>
- [13] Hasan, F. (2022, May 24). What are some deep details about pooling layers in CNN? Educative: Interactive Courses for Software Developers.
- [14] <https://www.kaggle.com/datasets/apollo2506/facial-recognition-dataset?fbclid=IwAR2e5tRXQxQyv9hlHJ1w0DAQQHbZZW5i9tdlw2SVQ2xawtw> xFkuOlkUxwmoC-