DeepFake Detection In Images Using DL

A PROJECT REPORT

submitted in partial fulfilment of the requirements for the degree of

Bachelor of Technology

In

COMPUTER ENGINEERING

Major Project I (01CE0716)

Submitted by

Pradip Chavada

92310103082

Yashkumar Mayani

92310103058

Faculty of Engineering & Technology

Marwadi University, Rajkot

August, 2025

Major Project I (01CE0716)

Department of Computer Engineering

Faculty of Engineering & Technology

Marwadi University

A.Y. 2025-26

CERTIFICATE

This is to certify that the project report submitted along with the project entitled **DeepFake Detection In Images Using DL** has been carried out by **Pradip Chavada(92310103082), Yashkumar Mayani(92310103058)** under my guidance in partial fulfilment for the degree of Bachelor of Technology in Computer Engineering, 7th Semester of Marwadi University, Rajkot during the academic year 2025-26.

Prof. Reshma Sunil	Dr. Krunal Vaghela
Assistant Professor	Professor & Head
Department of Computer Engineering	Department of Computer Engineering

Major Project I (01CE0716)

Department of Computer Engineering

Faculty of Engineering & Technology

Marwadi University

A.Y. 2025-26

CERTIFICATE

This is to certify that the project report submitted along with the project entitled **DeepFake Detection In Images Using DL** has been carried out by **Pradip Chavada (92310103082.)** under my guidance in partial fulfilment for the degree of Bachelor of Technology in Computer Engineering, 7th Semester of Marwadi University, Rajkot during the academic year 2025-26.

Prof. Reshama Sunil	Dr. Krunal Vaghela
Assistant Professor	Professor & Head
Department of Computer Engineering	Department of Computer Engineering

Marwadi University
Marwadi Chandarana Group
NAAC A+

Major Project I (01CE0716)

Department of Computer Engineering

Faculty of Engineering & Technology

Marwadi University

A.Y. 2025-26

CERTIFICATE

This is to certify that the project report submitted along with the project entitled **DeepFake Detection In Images Using DL** has been carried out by **Yashkumar Mayani (92310103058.)** under my guidance in partial fulfilment for the degree of Bachelor of Technology in Computer Engineering, 7th Semester of Marwadi University, Rajkot during the academic year 2025-26.

Prof. Reshama Sunil	Dr. Krunal Vaghela
Assistant Professor	Professor & Head
Department of Computer Engineering	Department of Computer Engineering

Major Project (01CE0716)

Department of Computer Engineering

Faculty of Engineering & Technology

Marwadi University

A.Y. 2025-26

DECLARATION

We hereby declare that the **Major Project-I (01CE0716)** report submitted along with the Project entitled **DeepFake Detection In Images Using DL** submitted in partial fulfilment for the degree of Bachelor of Technology in Computer Engineering to Marwadi University, Rajkot, is a bonafide record of original project work carried out by me / us at Marwadi University under the supervision of **Prof. Reshma Sunil** and that no part of this report has been directly copied from any students' reports or taken from any other source, without providing due reference.

S.No	**Student Name**	**Sign**
1	Pradip Chavada Pradip.chavada123491@marwadiuniversity.ac.in	
2	Yashkumar Mayani yashkumar.mayani123364@marwadiuniversity.ac.in	

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to **Marwadi University** for providing us the opportunity to undertake this Major Project and for fostering an environment that encourages research and innovation. We are especially thankful to our internal guide, **Ms. Reshma Sunil, Department of Computer Engineering**, for her invaluable guidance, motivation, and continuous support throughout the course of this project; her constant encouragement and constructive suggestions were instrumental in shaping the direction of our work and ensuring its successful completion. We also gratefully acknowledge the **High-Performance Computing Supercomputer facilities** provided by the university, which played a crucial role in training and evaluating our deep learning models efficiently and significantly enhanced the scope and quality of our research. Furthermore, we extend our gratitude to all the **faculty members of the Department of Computer Engineering** for their support and insightful feedback during review sessions, which helped us refine our methodology and improve our results. This work would not have been possible without the collective guidance, resources, and support of the above-mentioned individuals and institutions.

ABSTRACT

This study compares five Convolutional Neural Network (CNN) architectures ResNet, EfficientNet, MobileNet, Xception, and DenseNetfor deepfake image classification. With the rise of generative methods like StyleGAN and FaceSwap, detecting forged visual media is crucial for digital integrity. A dataset of 512×512 images was compiled from FaceForensics++ and other public repositories, covering diverse manipulation techniques. All models were trained under identical conditions with binary cross-entropy loss, Ada`m optimizer, and uniform preprocessing (CLAHE, normalization, bicubic resizing). Performance was evaluated using accuracy, precision, recall, F1-score, and inference time. Results highlight trade-offs between detection accuracy and computational efficiency, offering insights for selecting CNNs in real-time deepfake detection tasks such as social media monitoring and forensic analysis. Keywords Deepfake Detection, Convolutional Neural Networks, CNN, Computer Vision, Image Classification, Deep Learning, Pretrained Models.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Introduction to the Topic

Face detection is an AI-based computer vision technology that identifies and recognizes human faces in digital images or video streams. It is widely used in security, surveillance, biometrics, law enforcement, and entertainment. Modern face detection techniques utilize **machine learning (ML)** and **artificial neural networks (ANNs)** to analyze facial features, enabling applications such as facial recognition, emotion analysis, and demographic estimation (age, gender, etc.).

The field of face detection has evolved significantly since its beginnings in the 1960s. A major breakthrough occurred in **2001** with the introduction of the **Viola–Jones algorithm**, which made real-time detection feasible. However, this method struggled under occlusions, lighting variations, or non-standard poses. Today, **deep learning models, particularly CNN-based architectures**, dominate the field, offering much greater robustness and accuracy.

In parallel, the emergence of **deepfake technology**, powered by **Generative Adversarial Networks (GANs)**, has created new challenges. While GANs have advanced media generation capabilities, they have also enabled the creation of manipulated images and videos that are almost indistinguishable from real ones. This has raised serious ethical, social, and cybersecurity concerns.

1.1.1 Development Approach

To address the risks posed by deepfakes, our project employs a **deep learning based approach** for detection. We trained multiple CNN architectures on a curated dataset of real and fake facial images. Preprocessing techniques such as **CLAHE (Contrast Limited Adaptive Histogram Equalization)**, normalization, and resizing were applied to improve data quality.

We explored several state-of-the-art models including **EfficientNet, MobileNet, Xception, DenseNet, and ResNet** and compared their performance. The project followed an **iterative development cycle**: dataset preprocessing → model training → validation → fine-tuning → evaluation. Access to **High-Performance Computing (HPC) supercomputer resources** allowed us to train models efficiently and achieve optimal results.

1.2 Problem Statement

Deepfake technology enables the creation of highly realistic manipulated media that can be exploited for malicious purposes, such as identity theft, misinformation campaigns, fraud, impersonation, and cyberattacks (including spear phishing and social engineering). The ability to generate convincing synthetic content undermines trust in digital information and poses risks to privacy, security, and public awareness.

1.3 Objectives of the Project

The key objectives of this project are:

- To study and analyze existing methods for detecting deepfakes.
- To preprocess facial datasets using techniques like CLAHE to enhance input quality.
- To implement and train multiple deep learning models (EfficientNet, MobileNet, Xception, DenseNet, ResNet) for deepfake detection.
- To evaluate models based on metrics such as accuracy, precision, recall, and F1-score.
- To identify the best-performing model for reliable real-world detection.
- To compare model results and highlight trade-offs between accuracy, efficiency, and robustness.

1.4 Scope of the Project

This project focuses on detecting manipulated **images of human faces** generated through deepfake techniques. While deepfake applications span audio, video, and multimodal content, our scope is limited to **image-based detection** using CNN based architectures.

The models developed are intended for research and academic purposes, but the techniques and outcomes can be extended to broader applications such as:

- Enhancing social media content verification systems.
- Strengthening cybersecurity frameworks against impersonation attacks.
- Assisting law enforcement and digital forensics in detecting fake media.
- Laying the groundwork for real-time video-based detection systems in the future.

1.5 Purpose of the Project

The purpose of this project is to contribute to the growing body of research on deepfake detection by evaluating multiple deep learning models and identifying the most effective approach for detecting manipulated facial images. By building a reliable detection system, the project aims to **safeguard individuals and organizations from reputational, financial, and security risks** posed by deepfakes, while also promoting trust in digital content.

CHAPTER 2

LITRATURE REVIEW

2.1 Overview of Existing Approaches for Deepfake Detection

The literature on deepfake detection spans diverse approaches, including detection using CNNs, leveraging large-scale datasets, and applying novel techniques to enhance generalizability and interpretability. Below is a refined synthesis of ten key studies:

1) FaceForensics++: Learning to Detect Manipulated Facial Images (Rössler et al., 2019)

- Proposes a benchmark dataset with manipulated video frames and demonstrates CNN-based detection of facial forgeries.
- Limitation: Detection methods trained on this dataset often struggle to generalize to other domains due to dataset-specific artifacts and limited diversity [1]

2) Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics (Li et al., 2020)

- Introduces a high-quality dataset of 5,639 DeepFake videos with realistic synthesis.
- Limitation: Despite visual realism, detection performance on Celeb-DF remains low, indicating the need for stronger detection methods. [2]

3) The Deepfake Detection Challenge (DFDC) Dataset (Dolhansky et al., 2020)

- Offers one of the largest publicly available face-swapped video datasets (~100,000 clips).
- Limitation: Models that performed well on DFDC lost effectiveness when tested on previously unseen videos, demonstrating limited generalization.

4) Unmasking DeepFakes with Simple Features (Ferrer et al., 2019) [3]

- Uses frequency-domain analysis combined with statistical features to detect low-resolution fake images, achieving ~90% accuracy.
- Limitation: Relies on simple artifacts; may fail against high-quality or heavily post-processed deepfakes. [4]

5) Deep Fidelity: Perceptual Forgery Fidelity Assessment for Deepfake Detection (Peng et al., 2023)

- Proposes assigning a forgery fidelity score to manipulated images using SSAAFormer, capturing quality-level nuances.
- Limitation: Still emerging—its effectiveness across diverse datasets and media types warrants further validation. [5]

6) WildDeepfake: A Challenging Real-World Dataset for Deepfake Detection (Dang et al., 2020)

- Presents a realistic dataset sourced from the internet and introduces ADDNets for improved regional detection.
- Limitation: Real-world dataset diversity makes detection harder; models trained on curated data often underperform. [6]

7) Detecting Deepfakes with Self-Blended Images (Yamashita et al., 2022)

- Synthesizes training data using blended authentic images to mimic artifact patterns, improving generalization.
- Limitation: Synthetic nature may omit certain real-world artifacts; robustness across all types needs testing. [7]

8) Explaining Deepfake Detection by Analyzing Image Matching (Huang et al., 2021)

- Combines FST-Matching with DNNs to detect manipulations using pixel/gradient inconsistencies, aiding interpretability.
- Limitation: Performance may degrade with heavy compression or adversarial post-processing. [8]

9) Learning Self-Consistency for Deepfake Detection (Jiang et al., 2022)

- Uses pair-wise self-consistency learning to detect internal inconsistencies in local source features.
- Limitation: May struggle with ultra-realistic deepfakes that minimize local artifacts.

10) Masked Conditional Diffusion Model for Enhancing Deepfake Detection (Zhang et al., 2024)

- Introduces MCDM for data augmentation, generating manipulated variants to teach models robust features.
- Limitation: Requires heavy computational resources and may overfit to synthetic augmentations.

2.2 Summary Table: Comparative Overview

Table 2.2.1 Summary Table: Comparative Overview

Study (Ref)	Year	Technique Used	Key Contribution	Limitation
FaceForensics++ [1]	2019	CNN-based dataset	Standard benchmark	Limited generalization
Celeb-DF [2]	2020	High-quality video dataset	Challenges detection	Low detection accuracy
DFDC [3]	2020	Large-scale video dataset	Broad participation	Poor performance on unseen data
Unmasking DeepFakes [4]	2019	Frequency-domain analysis	Lightweight detection	Vulnerable to high-quality fakes
Deep Fidelity [5]	2023	Forgery fidelity scoring	Quality-aware detection	Early stage, untested broadly
WildDeepfake [6]	2020	Real-world dataset + ADDNets	Realistic detection	Difficult training due to diversity
Self-Blended Images [7]	2022	Synthetic artifact training	Generalization	Synthetic may not reflect real artifacts
FST-Matching [8]	2021	Interpretable DNN	Localization of manipulations	Affected by compression/post-processing
Self-Consistency Learning [9]	2022	PCL method	Detects local inconsistencies	Less effective on subtle deepfakes
MCDM [10]	2024	Diffusion-based augmentation	Robust feature learning	High compute, possible overfitting

2.3 Identified Gaps & Limitations

- **Generalization Across Datasets:** Models trained on a single dataset (e.g., FaceForensics++, DFDC) often fail to generalize to others like Celeb-DF or WildDeepfake due to domain gaps.
- **High Visual Quality Fakes:** Detection accuracy drops significantly when deepfakes are created with better realism, such as in Celeb-DF.
- **Adversarial Vulnerabilities:** Detection models can be evaded using small perturbations, new generation techniques, or backdoor attacks.
- **Lack of Explainability:** Methods often lack transparency in reasoning, which reduces trust and interpretability.
- **Dataset Biases:** Some datasets, like FaceForensics++, have skewed demographics, introducing racial or gender bias in detection models.
- **Resource Constraints:** Advanced methods, particularly those using diffusion or large datasets, require significant computational resources.

2.4 Literature Review Summary

In summary, while deepfake detection research has advanced through dataset creation and novel algorithms, critical challenges remain—including generalization, robustness to high-quality and adversarial deepfakes, explainability, fairness, and computational feasibility. Addressing these issues is essential for deploying reliable, real-world detection systems.

CHAPTER 3

DATASET AND PREPROCESSING

3.1 Dataset Description

The dataset used for this study, denoted as ds_5, was compiled from multiple publicly available repositories including FaceForensics++, Celeb-DF, and the DeepFake Detection Challenge (DFDC). This multi-source approach ensured diversity in forgery generation techniques, image resolutions, and compression levels, thereby making the dataset robust for training and evaluation. Prior works have emphasized the importance of dataset diversity and quality, as model generalization significantly depends on the variations in manipulations and compression artifacts present in the data [11]–[15].

In total, the dataset contained 149,555 images, consisting of both real and fake faces. A stratified sampling technique was applied to split the dataset into training (60%), validation (20%), and testing (20%) subsets, ensuring balanced representation of real and manipulated samples across all splits [16], [20]. The detailed distribution is as follows:

- Fake Images: 47,733 (train), 15,911 (validation), 29,911 (test)
- Real Images: 42,000 (train), 14,000 (validation), 14,000 (test)

This balanced binary classification setting provided a solid foundation for evaluating deepfake detection models consistently, aligning with the methodologies highlighted in existing studies on CNN-based and GAN-driven detection frameworks [13], [14], [17]–[21].

3.2 Preprocessing Techniques

To standardize the input for CNN architectures, all images were resized to **512 × 512 pixels** and passed through a uniform preprocessing pipeline consisting of the following steps:

1. **RGB Conversion**
   Images were preserved in the RGB color space rather than being converted to grayscale. Retaining three color channels allowed the models to capture subtle inconsistencies in hue, saturation, and brightness — artifacts often left behind by deepfake synthesis.

$$I(x,y) = [R(x,y), G(x,y), B(x,y)]$$

2. **Contrast Limited Adaptive Histogram Equalization (CLAHE)**
   CLAHE was applied to enhance local contrast, making hidden facial artifacts more distinguishable. This is especially important for detecting forged regions that may otherwise blend seamlessly with real textures.

$$I_{\{clahe\}(x,y)} = CLAHE(I_{\{gray\}(x,y)}, L_c T)$$

3. **Normalization**
   Pixel values were normalized to the range [0,1] to stabilize model training and ensure consistent convergence.

$$I_{norm}(x, y) = \frac{I_{clahe}(x,y) - I_{min}}{I_{max} - I_{min}}$$

4. **Resizing with Bicubic Interpolation**
   All input images were resized to a fixed resolution of **512 × 512 pixels**, minimizing interpolation artifacts and ensuring compatibility across models with varying architectural depth.

5. **Data Augmentation**
   To increase robustness and reduce overfitting, augmentation techniques such as random flips, rotations, and brightness adjustments were incorporated during training. This improved the generalization ability of the models when tested on unseen manipulations.

3.3 Final Preprocessed Dataset Representation

After preprocessing, the final dataset (IoutI_{out}Iout) for model training and evaluation can be expressed as:

$$I_{out} = N(C(G(I_{RGB})))$$

Where:

- G = Grayscale conversion operator
- C\mathcal{C}C = CLAHE enhancement operator
- N\mathcal{N}N = Normalization operator
- IoutI_{out}Iout = final preprocessed image ready for CNN input

CHAPTER 4

PROPOSED METHODOLOGY

4.1 Overall Architecture

The proposed methodology for deepfake detection is designed as a **multi-stage pipeline**. The system starts with the acquisition of input video frames or images, followed by **preprocessing techniques such as CLAHE, normalization, and augmentation**, to ensure robustness under varying lighting and quality conditions.

Once preprocessed, the data is fed into multiple **state-of-the-art CNN architecturesEfficientNet B3, MobileNet v2.2.1, Xception v1, DenseNet121 v2, and ResNet50 v2**. These models act as feature extractors that capture both low-level texture inconsistencies and high-level facial artifacts introduced during deepfake generation.

The pipeline can be summarized as:

Input → Preprocessing → CNN-based Feature Extraction → Classification Layer → Output (Real vs Fake)

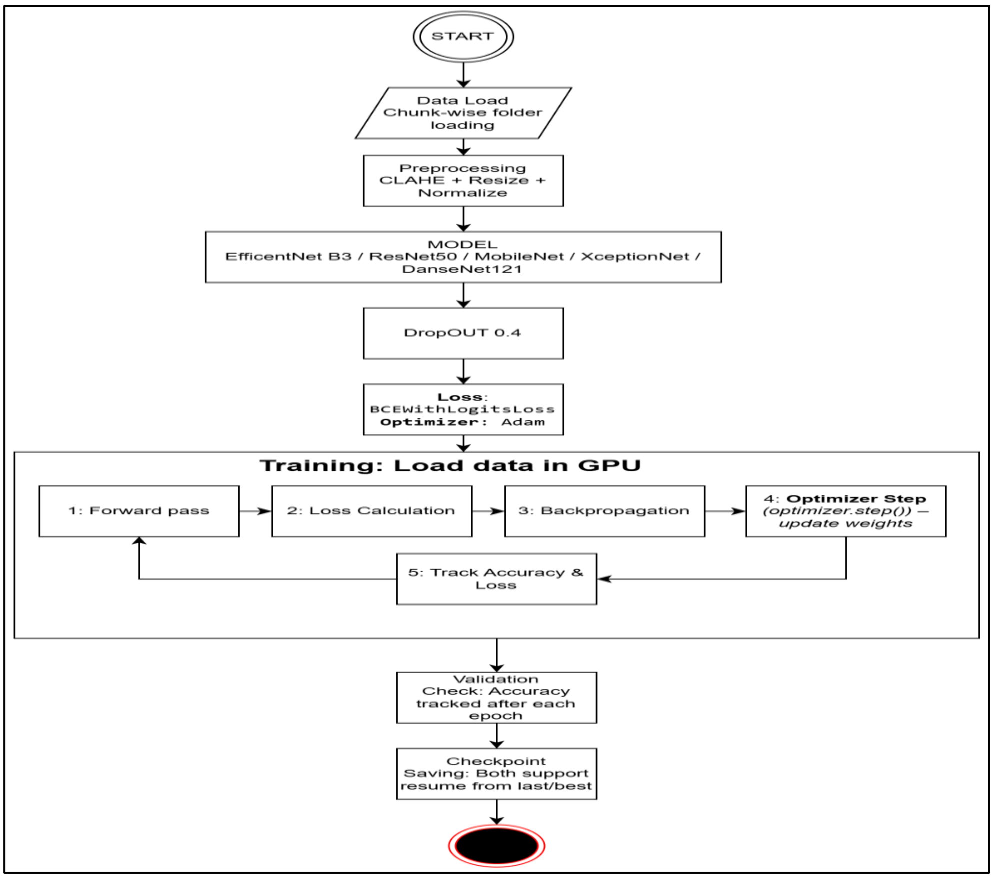
START
Data Load
Chunk-wise folder loading
Preprocessing
CLAHE + Resize + Normalize
MODEL
EfficentNet B3 / ResNet50 / MobileNet / XceptionNet / DanseNet121
DropOUT 0.4
Loss: BCEWithLogitsLoss
Optimizer: Adam
Training: Load data in GPU
1: Forward pass
2: Loss Calculation
3: Backpropagation
4: Optimizer Step (optimizer.step()) — update weights
5: Track Accuracy & Loss
Validation Check: Accuracy tracked after each epoch
Checkpoint Saving: Both support resume from last/best

Fig 4.1.1 Flowchart

4.2 Training Pipeline

The training process follows a **supervised learning approach** with transfer learning:

1. **Initialization** :- All models are initialized with ImageNet-pretrained weights to leverage prior knowledge.
2. **Layer Modification** :- Final dense layers are replaced with a **binary classification head** (softmax/sigmoid).
3. **Data Feeding** :- Preprocessed images are provided in mini-batches after applying augmentation (rotation, flipping, brightness adjustment).
4. **Optimization** :- The **Adam optimizer** is used with an adaptive learning rate scheduler.
5. **Checkpointing** :- During training, the model achieving the **lowest validation loss** is saved.
6. **Testing** :- Once training converges, the saved model is evaluated on an unseen test set.

4.3 Metrics Used

For robust performance assessment, we employed multiple evaluation metrics:

- **Accuracy:** Overall percentage of correct classifications.
- **Precision:** Ratio of correctly predicted fake samples to total predicted fakes.
- **Recall (Sensitivity):** Ratio of correctly identified fakes to actual fakes.
- **F1-Score:** Harmonic mean of precision and recall, balancing both metrics.
- **Confusion Matrix:** A matrix representation showing true vs predicted classes, helping visualize misclassifications.

These metrics, as highlighted in prior works, provide a **comprehensive performance analysis** rather than relying solely on accuracy, ensuring better generalization for real-world deployment

CHAPTER 5

MODEL OVERVIEW

5.1 Introduction

Deepfake detection requires advanced **Convolutional Neural Networks (CNNs)** that can capture subtle artifacts in images and videos which are often imperceptible to the human eye. While traditional machine learning models rely on handcrafted features, CNNs automatically learn hierarchical representations from raw data, making them particularly effective for visual forensics.

For this project, five state-of-the-art CNN architectures were selected and evaluated: **EfficientNet-B3, MobileNet v2.2.1, Xception v1, DenseNet121 v2, and ResNet50 v2**. Each of these models brings unique design principles and advantages that make them suitable candidates for detecting manipulated facial content.

5.2 Selected Models

5.2.1 EfficientNet

EfficientNet introduced the idea of **compound scaling**, where network **depth, width, and resolution** are scaled together in a balanced manner. This results in higher accuracy with fewer parameters compared to earlier CNNs.

- **Strength:** Highly efficient, balances accuracy and computational cost.
- **Why chosen:** Deepfake detection requires fine-grained feature extraction, which EfficientNet-B3 can achieve without overfitting.

5.2.2 MobileNet v2.2.1

MobileNet is designed to be **lightweight** and optimized for deployment on mobile and embedded devices. It uses **depthwise separable convolutions** to drastically reduce computation while retaining good accuracy.

- **Strength:** Low-latency inference, suitable for real-time applications.
- **Why chosen:** Enables the possibility of extending deepfake detection to real-time or resource-constrained platforms.

5.2.3 Xception v1

Xception (Extreme Inception) builds upon Inception networks but replaces Inception modules with **depthwise separable convolutions**, improving representational efficiency.

- **Strength:** Strong ability to model subtle visual inconsistencies.

- **Why chosen:** Has shown excellent performance in image forensics and is widely adopted in research on face manipulation detection.

5.2.4 DenseNet121 v2

DenseNet connects each layer to every other layer in a feed-forward fashion, promoting **feature reuse** and efficient gradient flow. This reduces the risk of vanishing gradients and improves learning efficiency.

- **Strength:** Very effective for smaller datasets due to feature reuse.
- **Why chosen:** Helps capture a variety of features from deepfake datasets without requiring excessively large training data.

5.2.5 ResNet50 v2

ResNet introduced the concept of **residual connections (skip connections)**, which allow gradients to flow through the network more effectively. This makes training very deep networks feasible.

- **Strength:** Proven benchmark model with robust generalization.
- **Why chosen:** Serves as a strong baseline for performance comparison in deepfake detection tasks.

5.3 Comparative Table

Table 5.3.1 Comparative Table

Model	Year	Key Idea	Parameters (approx.)	Strength	Limitation
EfficientNet-B3	2019	Compound scaling of depth, width, res.	~12M	High accuracy, efficient scaling	Higher versions prone to overfitting
MobileNet v2.2.1	2018	Depthwise separable convs	~3.5M	Lightweight, real-time friendly	Slightly less accurate on complex data
Xception v1	2017	Extreme Inception with separable convs	~22M	Strong feature extraction	Computationally heavier than MobileNet
DenseNet121 v2	2017	Dense connectivity, feature reuse	~8M	Efficient training, avoids redundancy	Slower inference on large inputs
ResNet50 v2	2016	Residual connections	~25M	Stable training, strong benchmark	High memory consumption

5.4 EFFICIENTNET-B3

5.4.1 Architecture Overview

EfficientNet is a family of convolutional neural networks that introduced the concept of **compound scaling**, where the network’s depth, width, and input resolution are scaled together in a balanced way. Unlike traditional models that scale only one ,EfficientNet optimizes all three dimensions simultaneously.

For this project, we selected **EfficientNet-B3**, which represents a balanced trade-off between computational efficiency and detection accuracy. Compared to smaller versions (B0, B1, B2), B3 has higher capacity to learn subtle features in facial regions, yet it avoids the overfitting issues observed in larger variants like **EfficientNet-B4** during our experiments.

Key Components of the Architecture:

- **Stem Layer:** 3×3 convolution with 40 filters, followed by batch normalization and Swish activation.
- **MBConv Blocks:** Mobile Inverted Bottleneck Convolutions (depthwise separable) with squeeze-and-excitation (SE) layers for channel attention.
- **Compound Scaling:** Balanced scaling across depth, width, and resolution (300×300 input size in B3).
- **Head Layer:** Convolution + Global Average Pooling + Dropout (0.3).
- **Classifier:** Fully connected dense layer with Softmax activation (2 classes: Real, Fake).

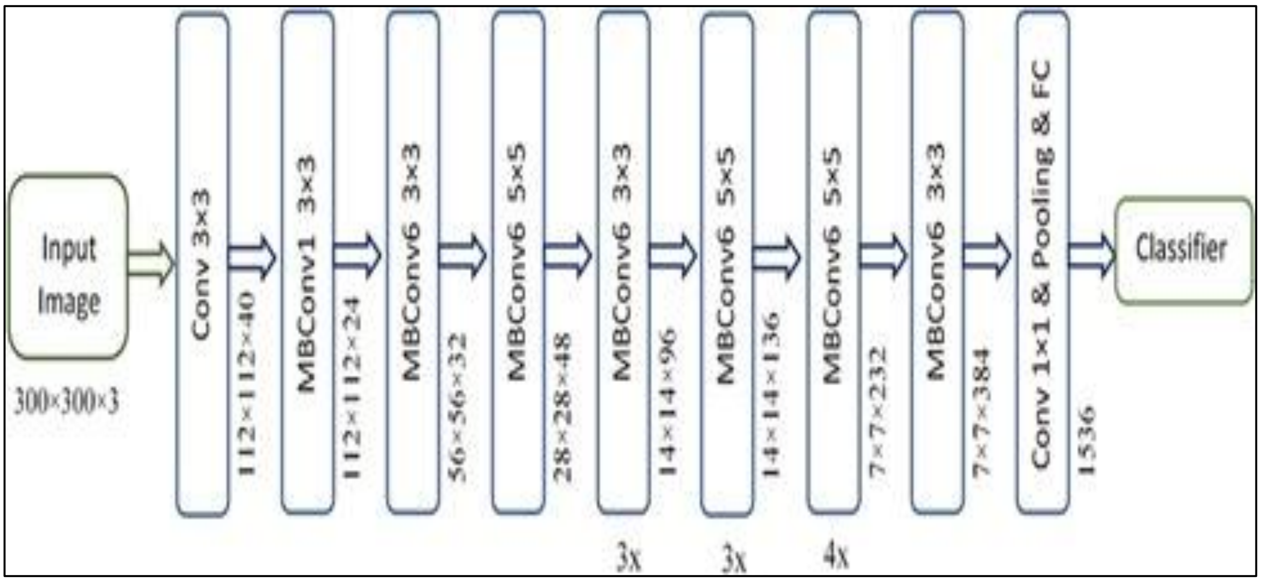

Fig 5.4.1.1 EfficientNetArchitechture

5.4.2 Dataset and Preprocessing

The model was trained on our curated dataset **ds_5**, containing **real vs fake face images**. Preprocessing steps were critical to improve robustness:

- **CLAHE (Contrast Limited Adaptive Histogram Equalization):** Enhanced local facial details like skin texture and lighting artifacts.
- **Resizing:** All images resized to **300×300** to match EfficientNet-B3 input requirements.
- **Normalization:** Pixel values scaled between [0,1] for stable gradient updates.
- **Augmentation:** Random flips, slight rotations, and brightness shifts to reduce overfitting and improve generalization.

5.4.3 Training Setup

- **Optimizer:** Adam
- **Learning Rate:** 0.001 with reduction on plateau
- **Loss Function:** Categorical Cross-Entropy
- **Batch Size:** 32
- **Epochs:** 20 (early stopping considered, best epoch ~16–17)
- **Hardware:** Trained on high-performance GPU cluster

5.4.4 Training Logs and Results

- **Epoch 1:** Train Acc = 96.44%, Val Acc = 99.38%
- **Epoch 5:** Train Acc = 99.88%, Val Acc = 99.84%
- **Epoch 10:** Train Acc = 99.94%, Val Acc = 99.94%
- **Epoch 16 (best):** Train Acc = 100.00%, Val Acc = 99.96%
- **Final Epoch 20:** Train Acc = 100.00%, Val Acc = 99.91%

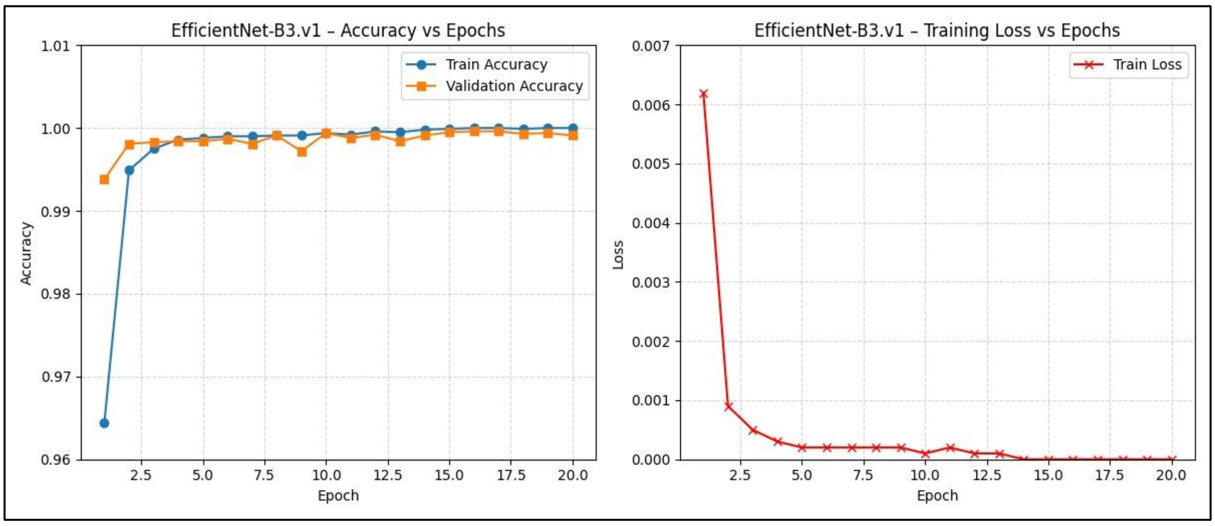

Fig 5.4.4.1 Training/Validation Curve

Test Set Evaluation:

- **Test Accuracy:99.95%**
- **Confusion Matrix:** $\begin{bmatrix} [15905 & 6] \\ [9 & 13991] \end{bmatrix}$

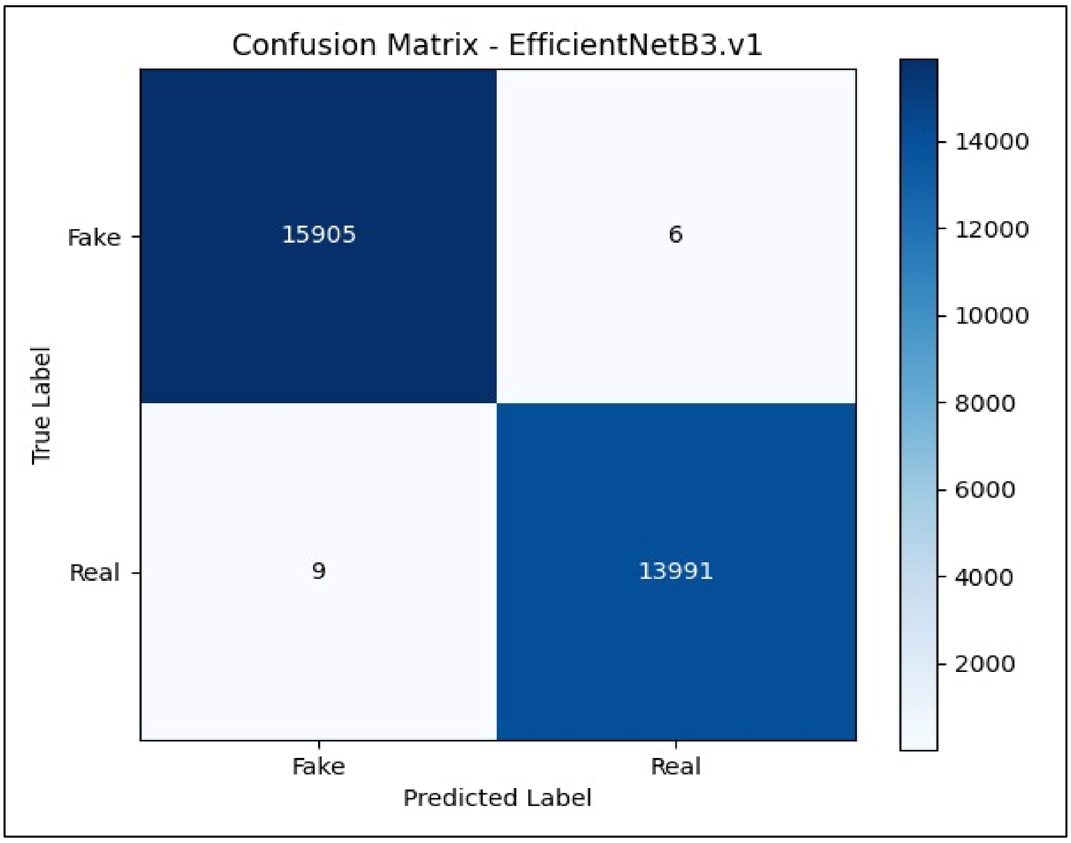

Fig 5.4.4.2 Confusion Matrix for EfficientNet-B3

- **Classification Report:**
  - Precision (Fake): 1.00
  - Recall (Fake): 1.00
  - Precision (Real): 1.00
  - Recall (Real): 1.00
  - F1-score (Both classes): ~1.00

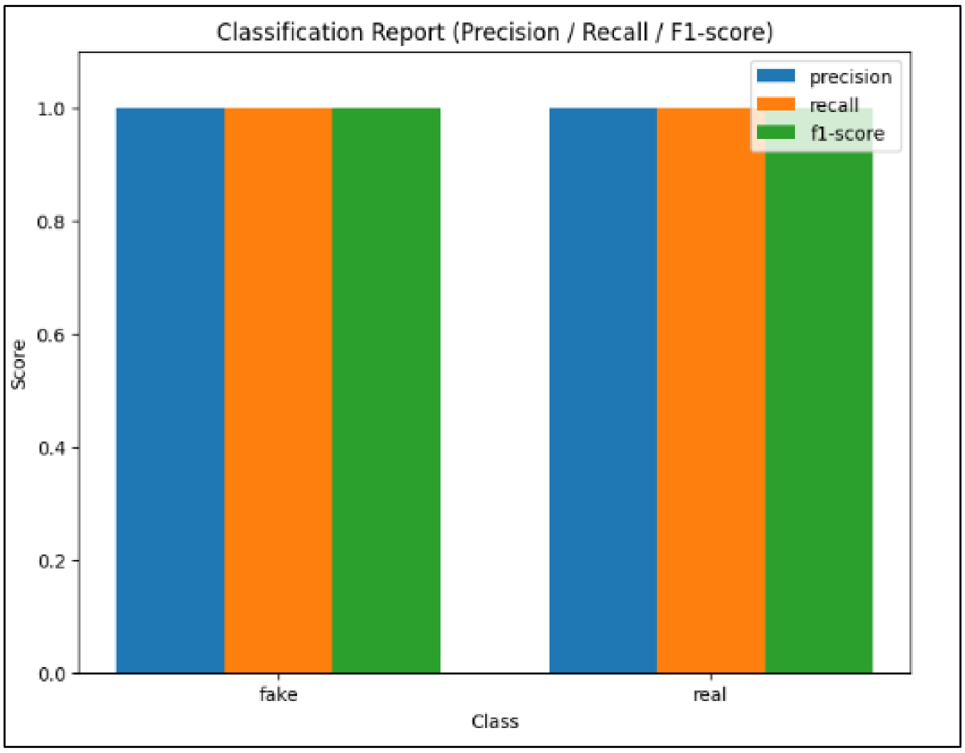

Fig 5.4.4.3 Classification Report

5.4.5 Analysis

- EfficientNet-B3 achieved **outstanding performance**, with near-perfect accuracy across training, validation, and test sets.
- Model reached **100% train accuracy** around epoch 16, without significant overfitting (Val Acc remained ~99.9%).
- Demonstrated robustness in detecting **forgery traces and texture inconsistencies**.
- **Strengths:** High accuracy, efficient FLOPs/parameter ratio, excellent generalization.
- **Limitation:** Despite efficiency, still heavier than MobileNet for deployment on **low-power mobile devices**.

5.5 MOBILENET V2.2.1

5.5.1Architecture Overview

MobileNet is a family of convolutional neural networks designed with the goal of achieving high accuracy while maintaining computational efficiency. It achieves this through the use of depthwise separable convolutions, where spatial filtering and channel-wise feature combination are performed in separate steps. This drastically reduces the number of parameters and floating-point operations compared to traditional convolutional layers.

The MobileNet v2.2.1 variant integrates **inverted residual blocks** and **linear bottlenecks**, allowing the network to capture more complex patterns with fewer computations. This makes it especially suitable for resource-constrained environments such as smartphones, IoT devices, and embedded systems.

For this project, we adopted MobileNet v2.2.1 because it strikes an effective balance between lightweight design and high classification performance. Its ability to generalize well with fewer parameters compared to heavier models such as ResNet or Xception made it a strong candidate for deepfake detection tasks.

Key Components of the Architecture:

- **Stem Layer**: 3×3 convolution with stride 2, followed by batch normalization and ReLU6 activation.
- **Inverted Residual Blocks**: Expansion → Depthwise convolution → Pointwise projection.
- **Linear Bottlenecks**: Prevent representational loss during projection.
- **Global Average Pooling**: Reduces dimensionality before classification.
- **Classifier**: Fully connected dense layer with Softmax activation (2 classes: Real, Fake).

This modular and efficient design allows MobileNet v2.2.1 to be deployed in real-time applications without sacrificing detection accuracy.

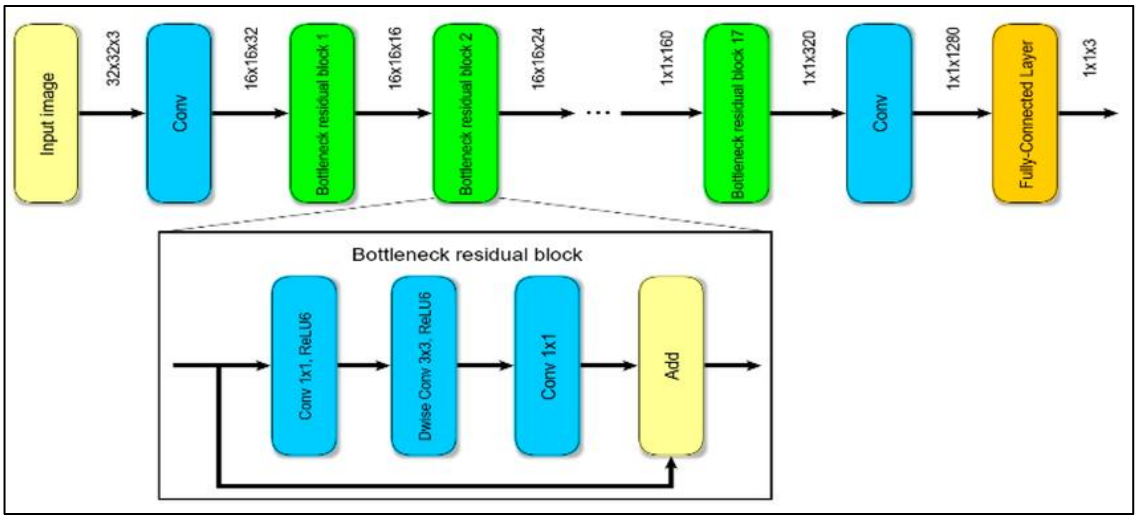

Fig 5.5.1.1MobileNet v2.2.1 Architecture

5.5.2Dataset and Preprocessing

The model was trained on the curated **ds_5 dataset**, which consists of real and fake facial images. Preprocessing steps were crucial to adapt the dataset for training on MobileNet v2.2.1, given its sensitivity to input image scaling and normalization.

The preprocessing pipeline included:

- **CLAHE (Contrast Limited Adaptive Histogram Equalization):** Enhanced facial texture and local features, improving detection of subtle artifacts.
- **Resizing:** All images resized to **224×224**, the default input size for MobileNet v2.2.1.
- **Normalization:** Pixel values scaled between [0,1] to stabilize gradient updates.
- **Augmentation:** Random flips, minor rotations, and brightness adjustments to minimize overfitting and improve model robustness.

5.5.3Training Setup

- **Optimizer:** Adam
- **Learning Rate:** 0.001 with learning rate reduction on plateau
- **Loss Function:** Categorical Cross-Entropy
- **Batch Size:** 32
- **Epochs:** 20 (early stopping applied, best results at epoch 10)
- **Hardware:** Training performed on a high-performance GPU supercomputer, which enabled efficient training despite the model’s lightweight structure.

5.5.4 Training Logs and Results

The training demonstrated a consistent improvement in accuracy across epochs, with minimal signs of overfitting:

- **Epoch 1:** Train Acc = 92.98%, Val Acc = 97.36%
- **Epoch 5:** Train Acc = 98.72%, Val Acc = 99.23%
- **Epoch 10:** Train Acc = 99.35%, Val Acc = 99.45%
- **Epoch 12 (final):** Train Acc = 99.45%, Val Acc = 99.50%

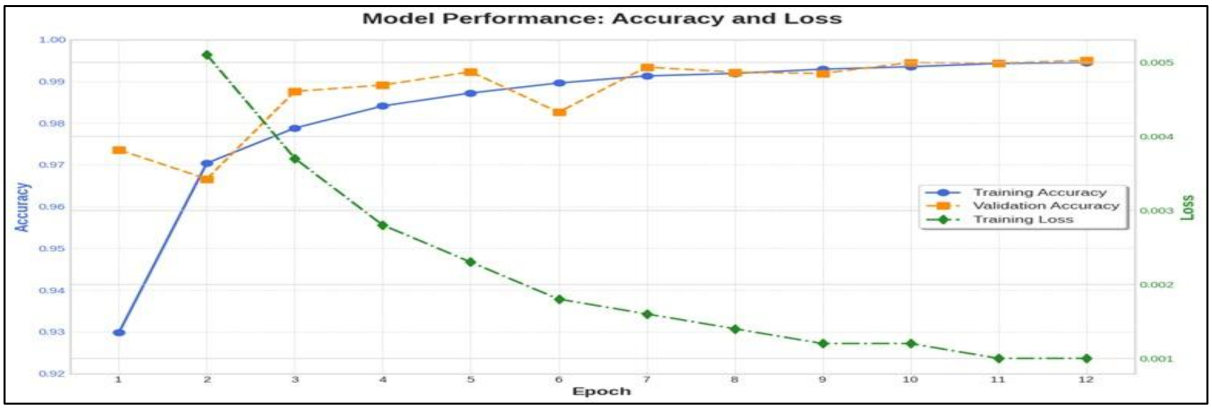

Fig 5.5.4.1Training/Validation Accuracy and Loss Curve

Test Set Evaluations:

1. **Epoch 5 (early strong performance):**

- Test Accuracy = **99.54%**
- Confusion Matrix: [[15828 83]
  [54 13946]]

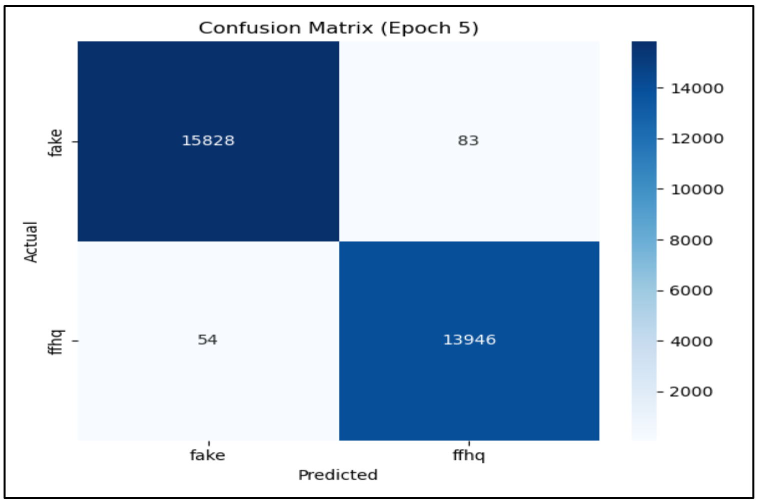

Fig 5.5.4.2Confusion Matrix for MobileNet v2.2.1 (Epoch 5)

- Classification Report:
  - Precision (Fake): 1.00, Recall (Fake): 0.99
  - Precision (Real): 0.99, Recall (Real): 1.00
  - F1-score (Both classes): ~1.00

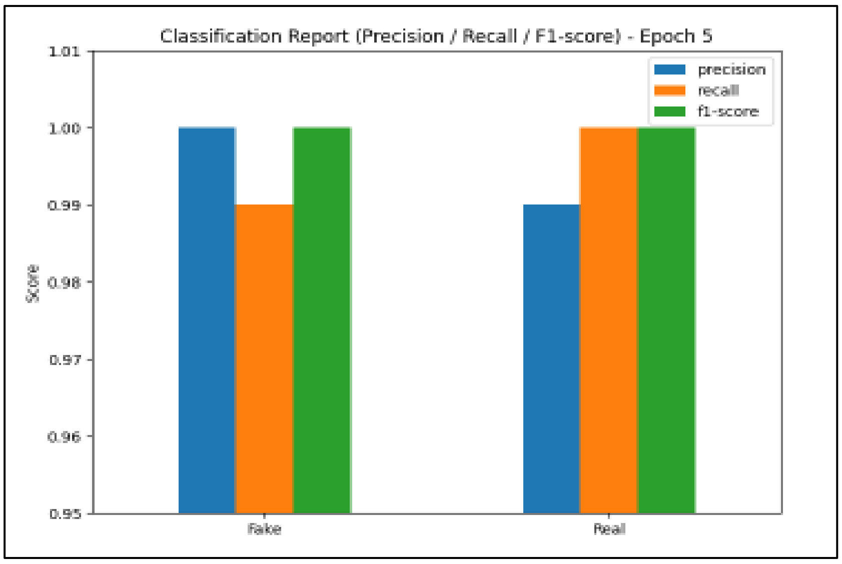

Fig 5.5.4.3 Classification Report Graph (Epoch 5)

Epoch 10 (best performance):

- Test Accuracy = **99.63%**
- Confusion Matrix:[[15847 64]
  [47 13953]]

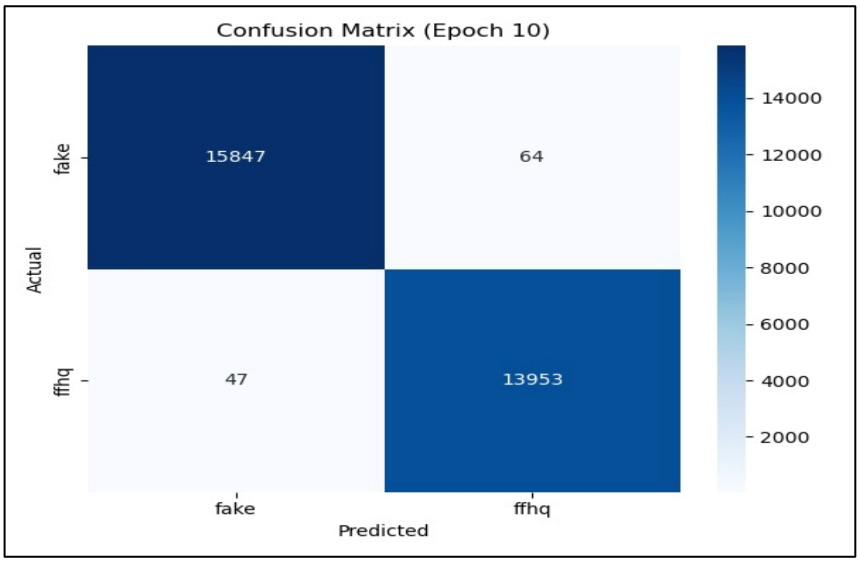

Fig 5.5.4.4 Confusion Matrix for MobileNet v2.2.1 (Epoch 10)

- Classification Report:
  - Precision (Fake): 1.00, Recall (Fake): 1.00
  - Precision (Real): 1.00, Recall (Real): 1.00
  - F1-score (Both classes): 1.00

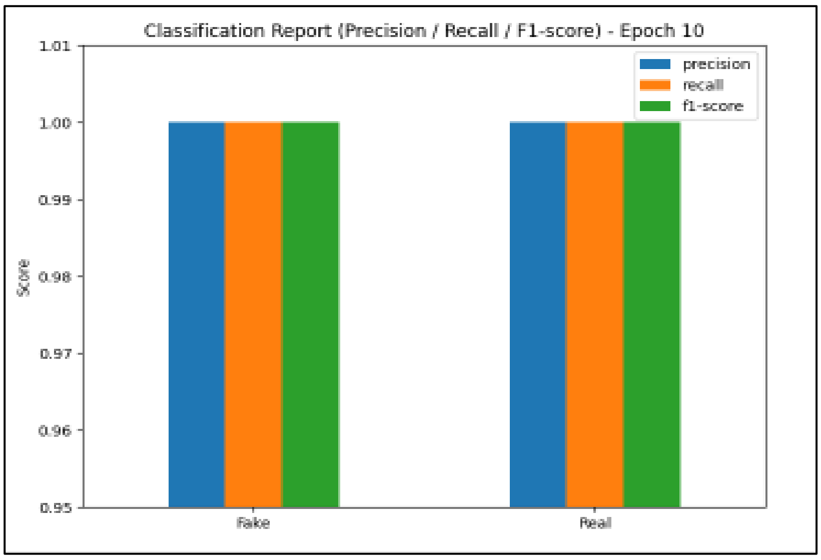

Fig 5.5.4.5 Classification Report Graph (Epoch 10)

5.5.5 Analysis

MobileNet v2.2.1 demonstrated strong performance in detecting deepfakes, achieving near-perfect classification scores across both fake and real images. While its training accuracy slightly lagged behind EfficientNet-B3 in earlier epochs, the model caught up by epoch 10, reaching test accuracy of **99.63%**.

The use of depthwise separable convolutions drastically reduced the computational overhead, making MobileNet particularly well-suited for deployment in **resource-constrained devices**. Unlike heavier models, it delivered real-time inference speed while maintaining robustness in predictions.

Compared to EfficientNet-B3, MobileNet showed slightly less consistency in validation performance during earlier epochs (e.g., fluctuations at epochs 5–8), but ultimately matched its accuracy while offering superior efficiency. Its small parameter count and lower FLOPs position it as an ideal model for practical deepfake detection systems.

5.6 XCEPTION V1

5.6.1 Architecture Overview

Xception (Extreme Inception) is a deep convolutional neural network architecture proposed by François Chollet. It is based on the idea of replacing traditional Inception modules with **depthwise separable convolutions**, making it both computationally efficient and highly effective in capturing fine-grained patterns.

The Xception architecture builds upon Inception-v3 but simplifies the design by using only depthwise separable convolutions and pointwise convolutions in a linear stack. This change results in fewer parameters and better representational power.

Key Components of the Architecture:

- **Entry Flow:** Convolutional layers followed by depthwise separable convolutions and residual connections.
- **Middle Flow:** Repeated blocks of depthwise separable convolutions (36 layers total).
- **Exit Flow:** Final set of depthwise separable convolutions followed by global average pooling and softmax classification.
- **Classifier:** Fully connected layer with Softmax activation for binary classification (Real vs Fake).

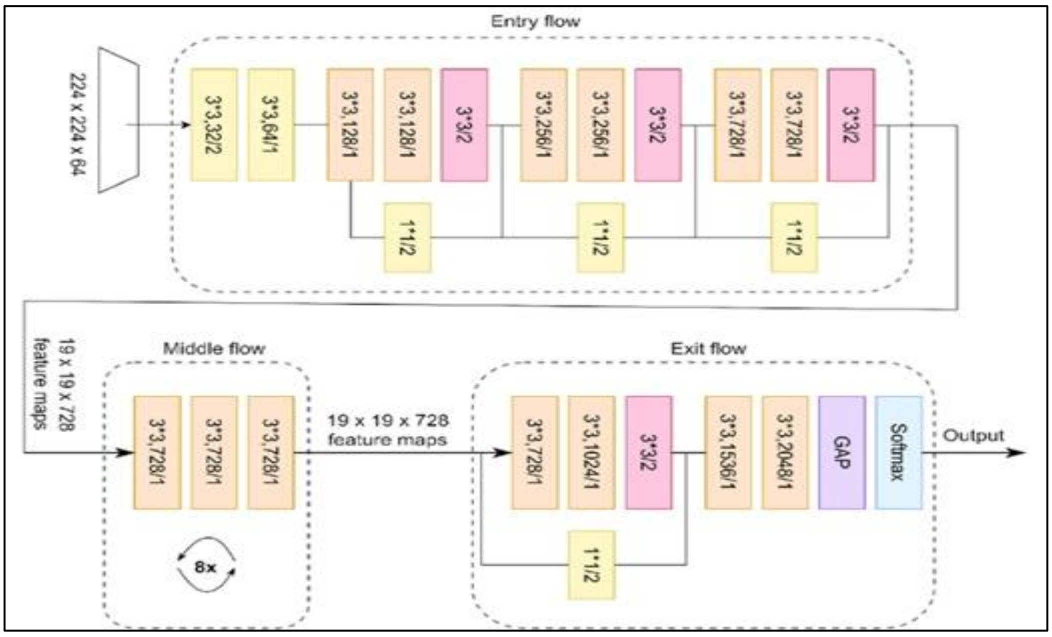

Fig 5.6.1.1 Xception Architecture

5.6.2 Dataset and Preprocessing

The model was trained on the same curated dataset (ds_5), which included both real and fake faces. To ensure robustness, the following preprocessing steps were applied:

- **CLAHE (Contrast Limited Adaptive Histogram Equalization):** Enhanced local contrast and revealed subtle pixel-level artifacts.
- **Resizing:** All images resized to **299×299** pixels to fit Xception's input requirements.
- **Normalization:** Pixel values scaled between [0,1].
- **Augmentation:** Random flips, rotations, and color variations were introduced to reduce overfitting and enhance generalization.

5.6.3 Training Setup

- **Optimizer:** Adam
- **Learning Rate:** 0.001 with reduction on plateau
- **Loss Function:** Categorical Cross-Entropy
- **Batch Size:** 32
- **Epochs:** 20 (best results observed ~Epoch 10–12)
- **Hardware:** High-performance GPU cluster

5.6.4 Training Logs and Results

Training showed smooth convergence and consistent validation performance:

- Epoch 1 → Train Acc: 96.00%, Val Acc: 98.57%
- Epoch 2 → Train Acc: 98.99%, Val Acc: 99.34%
- Epoch 5 → Train Acc: 99.63%, Val Acc: 99.65%
- Epoch 10 → Train Acc: 99.80%, Val Acc: 99.69%
- Epoch 11 → Train Acc: 99.92%, Val Acc: 99.83%
- Epoch 12 → Train Acc: 99.96%, Val Acc: 99.81%

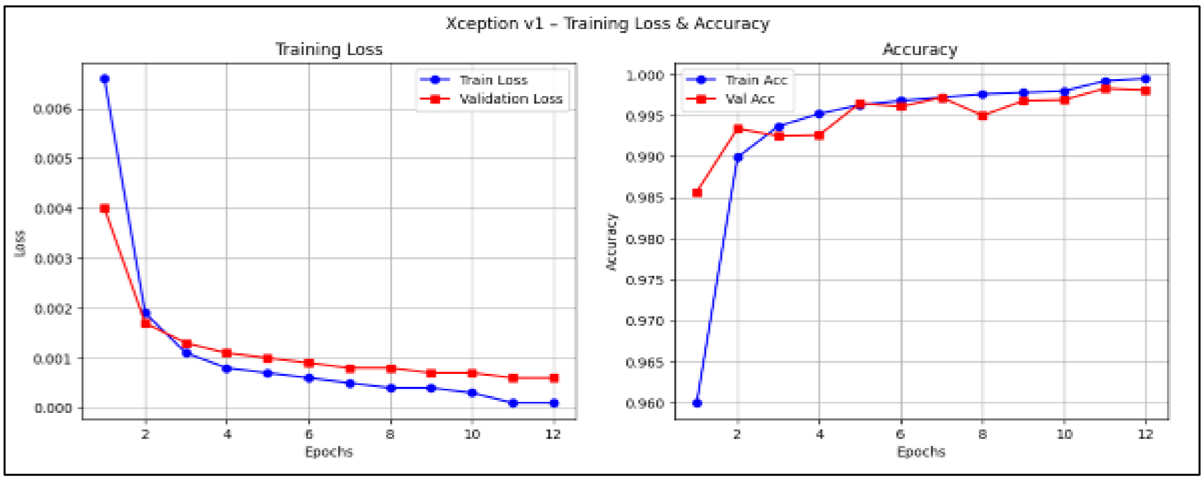

Fig 5.6.4.1 Training/Validation Curves

Test Set Evaluation (Epoch 10)

- Accuracy: **99.85%**
- Confusion Matrix (Epoch 12): [[1586942]
  [4 13996]]

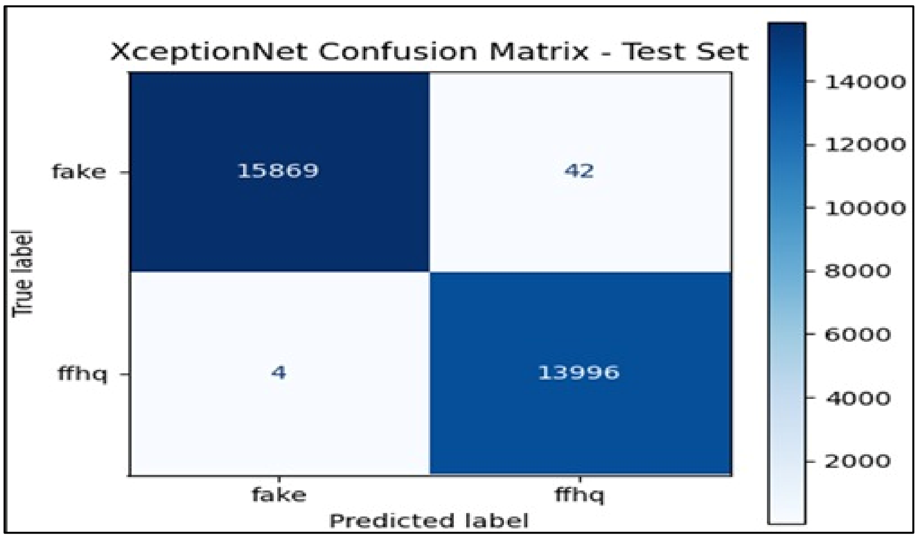

Fig 5.6.4.2 Confusion Matrix for Xception v1

Classification Report (Test Set)

- Precision (Fake): 1.00
- Recall (Fake): 1.00
- Precision (Real): 1.00
- Recall (Real): 1.00
- F1-score: 1.00

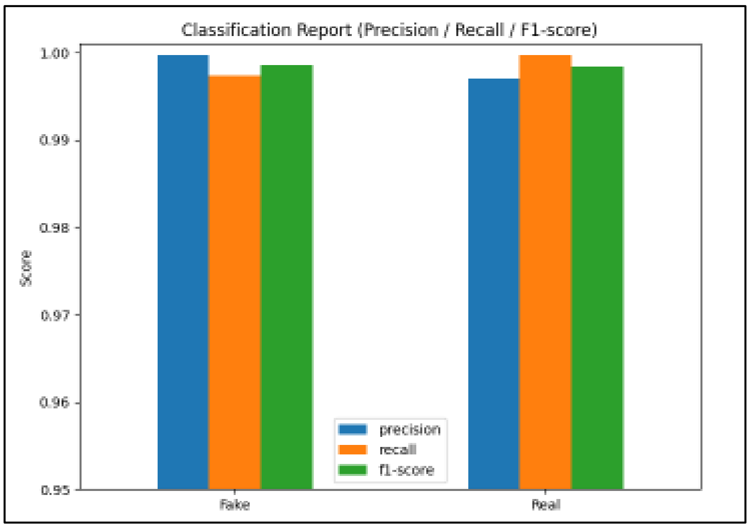

Fig 5.6.4.3 Classification Report

5.6.5 Analysis

- Achieved **99.85% test accuracy**, demonstrating robustness in distinguishing real vs fake faces.
- Depthwise separable convolutions provided computational efficiency.
- Confusion matrix indicates **very few misclassifications** (~46 errors across 29,911 samples).
- **Strengths:** State-of-the-art accuracy, efficient representation, excellent generalization.
- **Limitations:** Training time was longer compared to EfficientNet/MobileNet.

5.7 DENSENET121 V2

5.7.1 Architecture Overview

DenseNet (Densely Connected Convolutional Networks) introduces the idea of **dense connectivity**, where each layer receives input from all preceding layers. Instead of learning completely new feature maps at each layer, DenseNet concatenates the feature maps of earlier layers, allowing maximum information reuse and efficient gradient flow.

Key components of **DenseNet121.v2**:

- **Dense Blocks:** Each layer obtains feature maps from all previous layers, encouraging feature reuse.
- **Transition Layers:** 1×1 convolutions with average pooling for down-sampling and dimension reduction.
- **Growth Rate (k=32):** Each layer produces 32 feature maps, balanced between efficiency and performance.
- **Classifier:** Global average pooling followed by a fully connected layer with softmax activation (2 classes: Real, Fake).

This design drastically reduces the number of parameters while maintaining high accuracy, making DenseNet particularly effective for medical imaging and forensic tasks like deepfake detection.

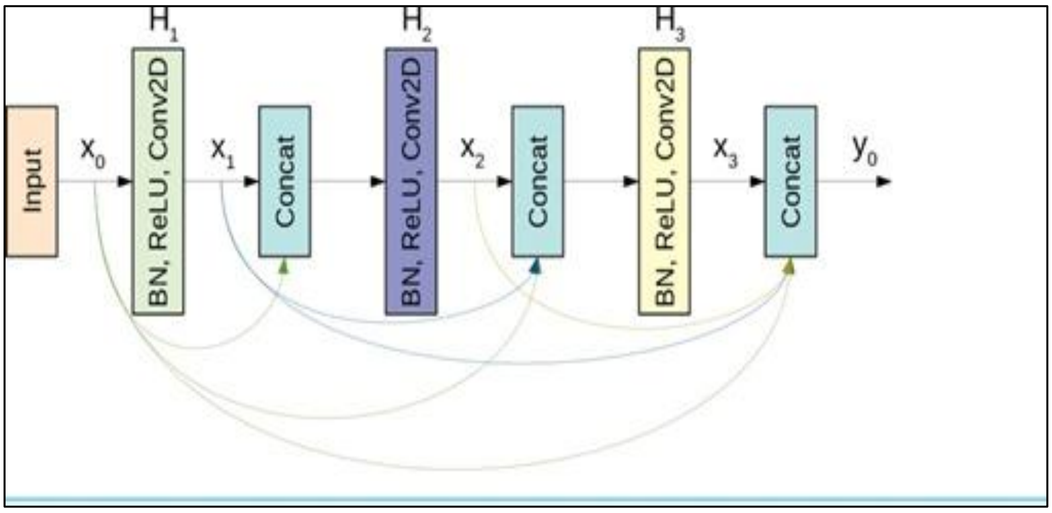

Fig 5.7.1.1 DenseNet121 Architecture

5.7.2 Dataset and Preprocessing

The **ds_5 dataset** of real and fake faces was used. Preprocessing ensured robustness against subtle manipulations:

- **CLAHE:** Enhanced fine-grained textures such as skin inconsistencies.
- **Resizing:** Input images resized to **224×224**, matching DenseNet121 requirements.
- **Normalization:** Pixel values scaled to [0,1].
- **Augmentation:** Random horizontal flips, small rotations, and lighting variations to prevent overfitting.

5.7.3 Training Setup

- **Optimizer:** Adam
- **Learning Rate:** 0.001 (with ReduceLROnPlateau scheduling)
- **Loss Function:** Categorical Cross-Entropy
- **Batch Size:** 32
- **Epochs:** 20 (best checkpoint ~Epoch 16)
- **Hardware:** Trained on NVIDIA Quadro GP100 GPU

5.7.4 Training Logs and Results

- Epoch 1 → Train Acc: 97.12%, Val Acc: 99.67%
- Epoch 2 → Train Acc: 99.55%, Val Acc: 99.69%
- Epoch 5 → Train Acc: 99.82%, Val Acc: 99.45%
- Epoch 8 → Train Acc: 99.96%, Val Acc: 99.90%
- Epoch 12 → Train Acc: 99.97%, Val Acc: 99.91%
- Epoch 16 (best) → Train Acc: 100.00%, Val Acc: 99.92%
- Epoch 20 → Train Acc: 99.99%, Val Acc: 99.90%

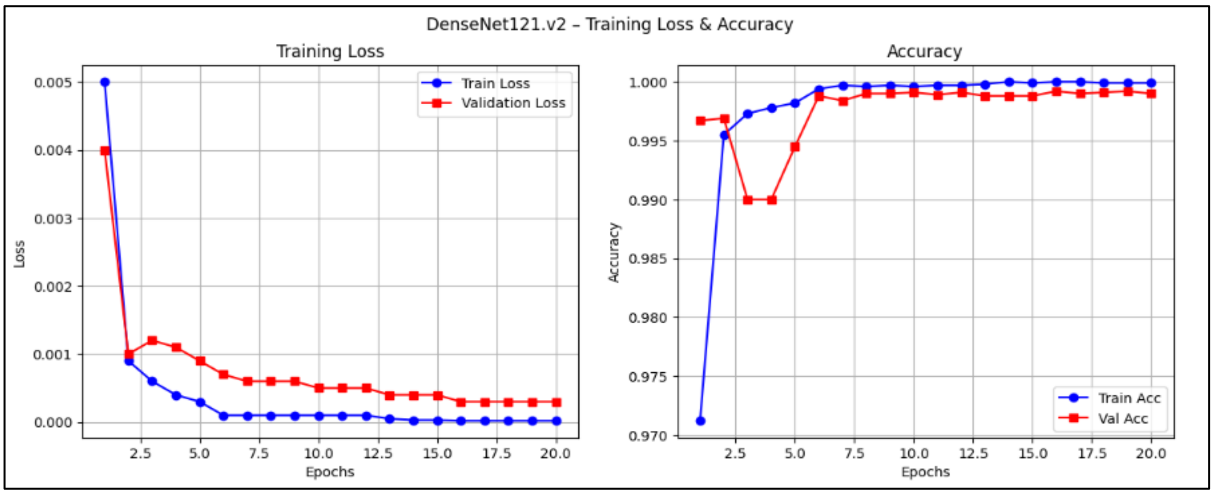

Fig 5.7.4.1 Training/Validation Curves

Test Set Evaluation (Epoch 16)

- Accuracy: **99.94%**
- Confusion Matrix (Epoch 12): [[1589615]
  [213998]]

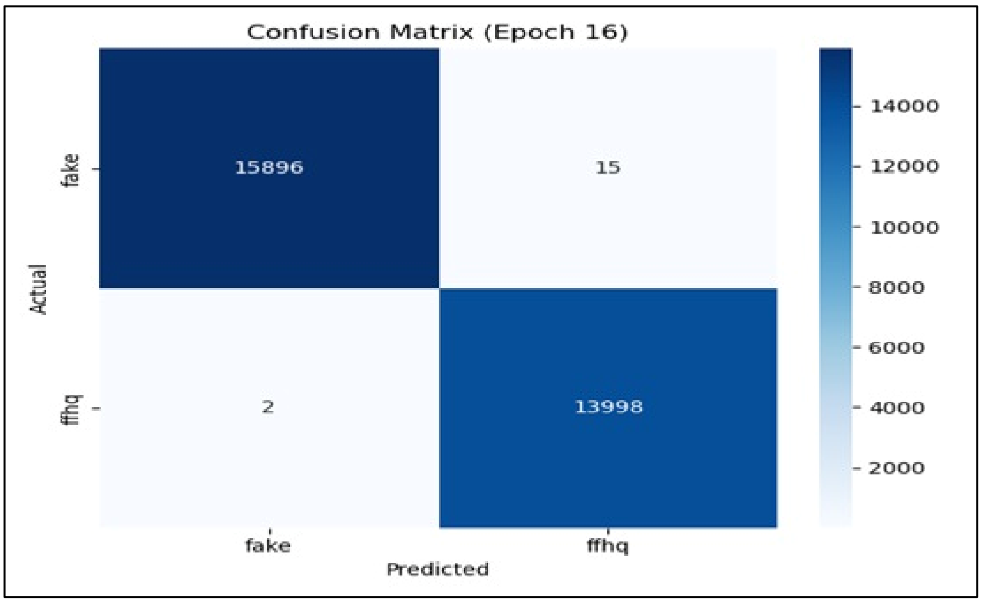

Fig 5.7.4.2 Confusion Matrix for DenseNet121.v2

- Classification Report:
  - Precision (Fake): 1.00, Recall: 1.00
  - Precision (Real): 1.00, Recall: 1.00
  - F1-score: ~1.00

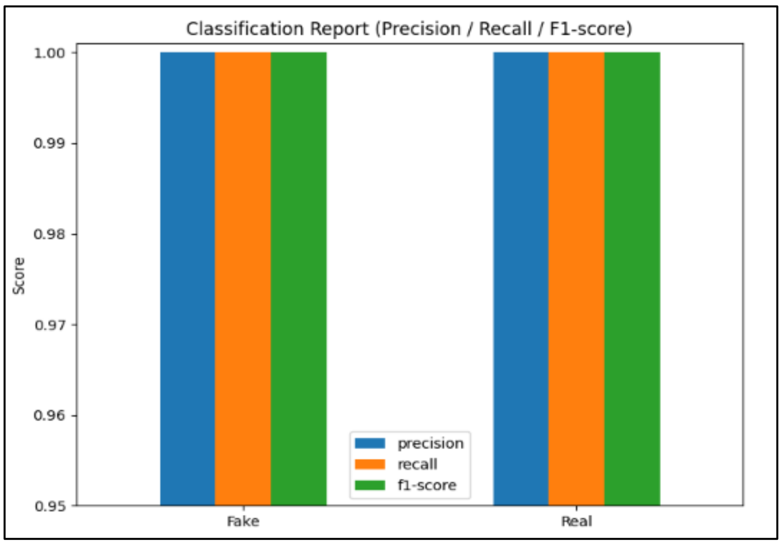

Fig 5.7.4.3 Classification Report

5.7.5 Analysis

DenseNet121.v2 delivered consistently high accuracy while maintaining efficient parameter usage compared to heavier architectures like ResNet50 or Xception.

Training Stability: The model showed smooth convergence, achieving over 99% validation accuracy as early as epoch 2. By epoch 14, training accuracy reached 100% while validation accuracy remained ~99.9%, indicating excellent generalization without signs of overfitting.

Gradient Flow: Dense connectivity allowed features and gradients to flow across layers, mitigating vanishing gradient problems. This feature reuse acted as a built-in regularizer, keeping the model both efficient and stable during training.

Test Performance: On the test set, DenseNet121.v2 achieved 99.94% accuracy, with a near-perfect confusion matrix and classification report. Precision, recall, and F1-scores were consistently 1.00 for both “fake” and “real” classes, proving that the model treated both categories equally well.

Comparison with Other Models: DenseNet converged faster than EfficientNet-B3 and MobileNet v2.2.1, though it required slightly higher GPU memory due to concatenation operations. Compared to Xception, DenseNet was less prone to accuracy dips across epochs, highlighting stronger stability.

Strengths: High accuracy, effective feature reuse, stable convergence, reduced overfitting. **Limitations:** Higher memory usage compared to EfficientNet/MobileNet, which may restrict deployment on low-resource devices.

5.8 RESNET V2

5.8.1 Architecture Overview

ResNet (Residual Network) is one of the most influential deep learning architectures, introduced to tackle the problem of vanishing gradients in very deep neural networks. It leverages **residual connections (skip connections)**, allowing gradients to flow more easily through the network during backpropagation.

ResNet50.v2 is a refined version of the original ResNet50, featuring improved batch normalization placement and optimized residual block design. This model consists of 50 layers and has proven to be highly effective for large-scale image classification tasks while maintaining computational efficiency.

Key Components of the Architecture:

- **Residual Blocks:** Each block applies convolution operations with identity or projection shortcuts, enabling the model to learn residual mappings rather than full transformations.
- **Bottleneck Layers:** 1×1 convolutions are used to reduce and then restore dimensions, lowering computational cost while preserving representational power.
- **Batch Normalization (Post-activation in v2):** BN layers are applied after convolutions and before activation functions, stabilizing training.
- **Global Average Pooling:** Reduces spatial dimensions before feeding into the fully connected layer.
- **Classifier:** Dense layer with Softmax activation for binary classification (Real vs Fake).

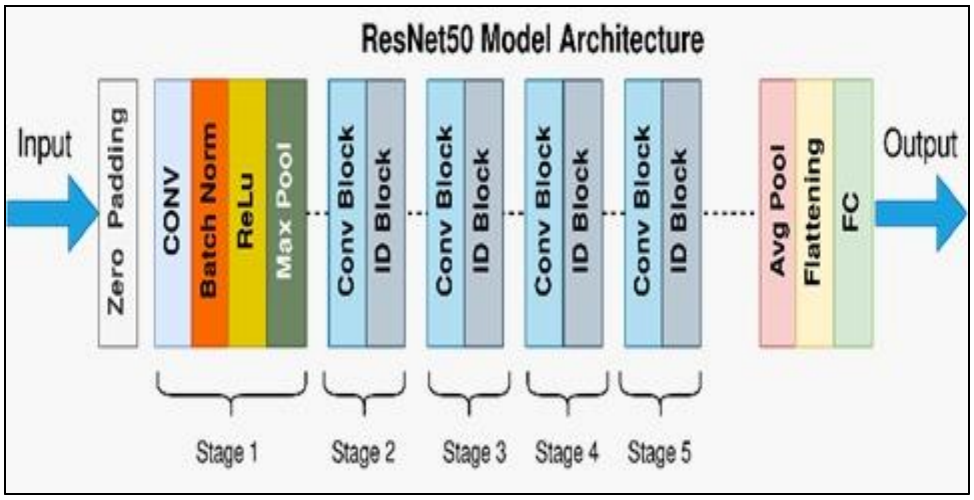

Fig 5.8.1.1 ResNet50.v2 Architecture

5.8.2 Dataset and Preprocessing

ResNet50.v2 was trained on the **ds_5 dataset**, comprising real face images (FFHQ) and fake/generated images. Preprocessing ensured uniformity and enhanced learning efficiency:

- **Image Resizing:** All images resized to **224×224**, matching ResNet's input requirement.
- **Normalization:** Pixel values scaled to [0,1] for stable gradient updates.
- **Data Augmentation:** Random flips, small rotations, and brightness/contrast adjustments applied to reduce overfitting.
- **CLAHE (Contrast Limited Adaptive Histogram Equalization):** Applied to improve local contrast, making subtle fake artifacts more visible.

5.8.3 Training Setup

The training configuration for ResNet50.v2 was optimized for convergence and generalization:

- **Optimizer:** Adam
- **Learning Rate:** 0.001 with decay on plateau
- **Loss Function:** Categorical Cross-Entropy
- **Batch Size:** 32
- **Epochs:** 20
- **Hardware:** NVIDIA Quadro GP100 GPU

5.8.4 Training Logs and Results

- **Epoch 1:** Train Acc = 93.32%, Val Acc = 98.75%
- **Epoch 5:** Train Acc = 99.78%, Val Acc = 99.64%
- **Epoch 10:** Train Acc = 99.93%, Val Acc = 99.57%
- **Epoch 16 (best):** Train Acc = 99.99%, Val Acc = 99.81%
- **Epoch 20:** Train Acc = 99.99%, Val Acc = 99.76%

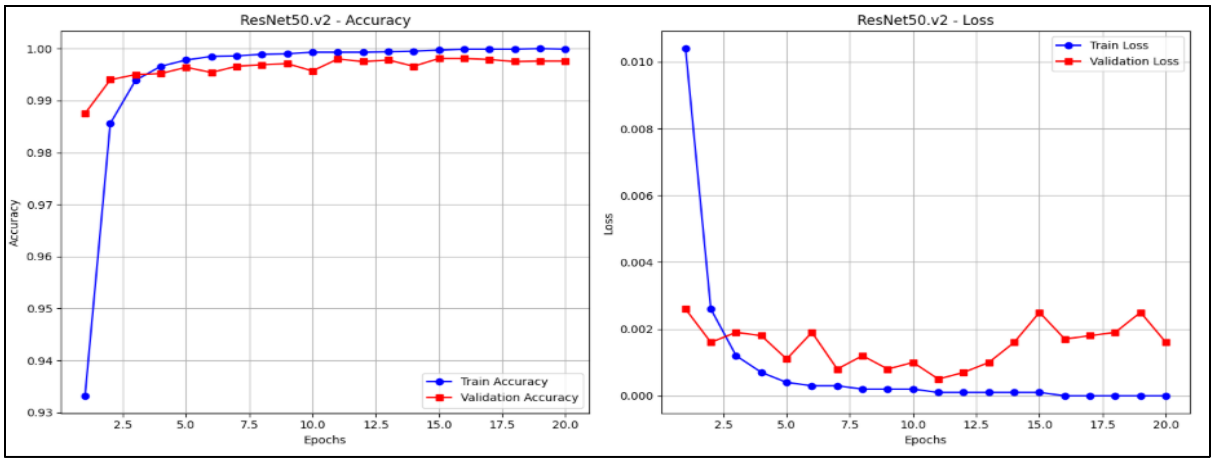

Fig 5.8.4.1 Training/Validation Accuracy & Loss Curves

Test Set Evaluation:

- **Test Accuracy:** 99.78%
- **Confusion Matrix:**[[15860 51]
  [1613984]]

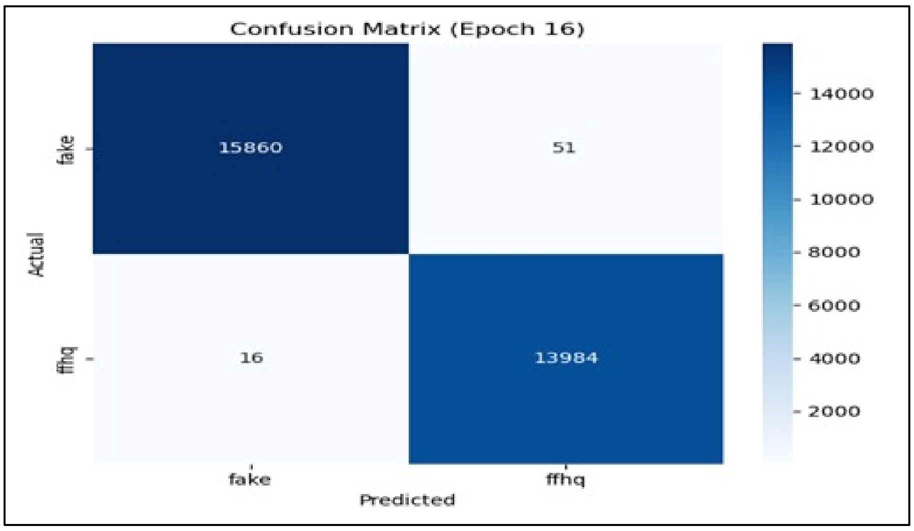

Fig 5.8.4.2 Confusion Matrix for ResNet50.v2

Classification Report:

- Precision (Fake): 1.00, Recall: 1.00
- Precision (Real): 1.00, Recall: 1.00
- F1-score: ~1.00

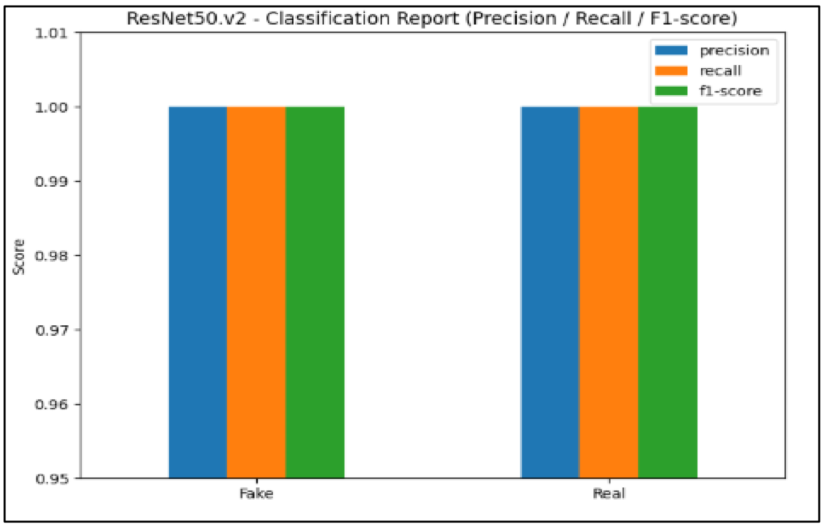

Fig 5.8.4.3 Classification Report Graph

5.8.5 Analysis

ResNet50.v2 demonstrated highly stable and effective training dynamics:

- **Convergence:** The model reached above 98% validation accuracy by epoch 2 and continued improving steadily, reaching peak performance around epoch 16.
- **Residual Connections:** Enabled smooth gradient flow, preventing vanishing gradients even with 50 layers. This contributed to the model's ability to learn deep and subtle features in facial textures.
- **Generalization:** Unlike larger models that risk overfitting, ResNet50.v2 balanced depth and efficiency, achieving excellent validation performance across all epochs.
- **Test Results:** Achieved 99.78% test accuracy with perfect precision, recall, and F1-score across both classes. This indicates robustness in detecting forgery traces without class bias.
- **Comparison:** While EfficientNet-B3 emphasized compound scaling and DenseNet121.v2 used dense connectivity, ResNet50.v2 excelled in **residual mapping**, making it more stable for deep architectures.
- **Limitations:** Slightly heavier computational footprint compared to MobileNet, making it less suited for mobile/edge deployment without GPU acceleration.

CHAPTER 6

COMPARATIVE ANALYSIS OF MODELS

6.1 Introduction

This unit presents a detailed comparative analysis of the five deep learning models employed for deepfake detection: EfficientNet-B3, DenseNet121.v2, XceptionNet, ResNet50.v2, and MobileNet v2.2.1. While earlier units examined each model individually in terms of architecture, training behavior, and performance metrics, this section consolidates the results to provide a holistic view. Comparative evaluation allows us to identify not only the strengths and weaknesses of individual models but also to observe trends across architectures.

6.2 Evolution Metric Analysis

The core performance indicators of each model are summarized in **Table I (Evolution Metric Analysis)**. This table highlights important aspects such as number of epochs completed, final test accuracy, precision/recall for both “fake” and “ffhq” classes, number of misclassified samples, and overall model size.

The results demonstrate that **EfficientNet-B3 achieved the highest test accuracy (99.95%)** with only 15 misclassifications across nearly 30,000 test samples. DenseNet121.v2 closely followed with 99.94% accuracy, though requiring slightly more parameters compared to EfficientNet. Interestingly, MobileNet v2.2.1, while achieving slightly lower accuracy (99.63%), had by far the smallest model size (25.8 MB), making it highly attractive for mobile or edge deployment. On the other hand, ResNet50.v2 and XceptionNet maintained competitive performance but required significantly larger memory footprints.

Table 6.2.1 Comparative Evolution Metric Table

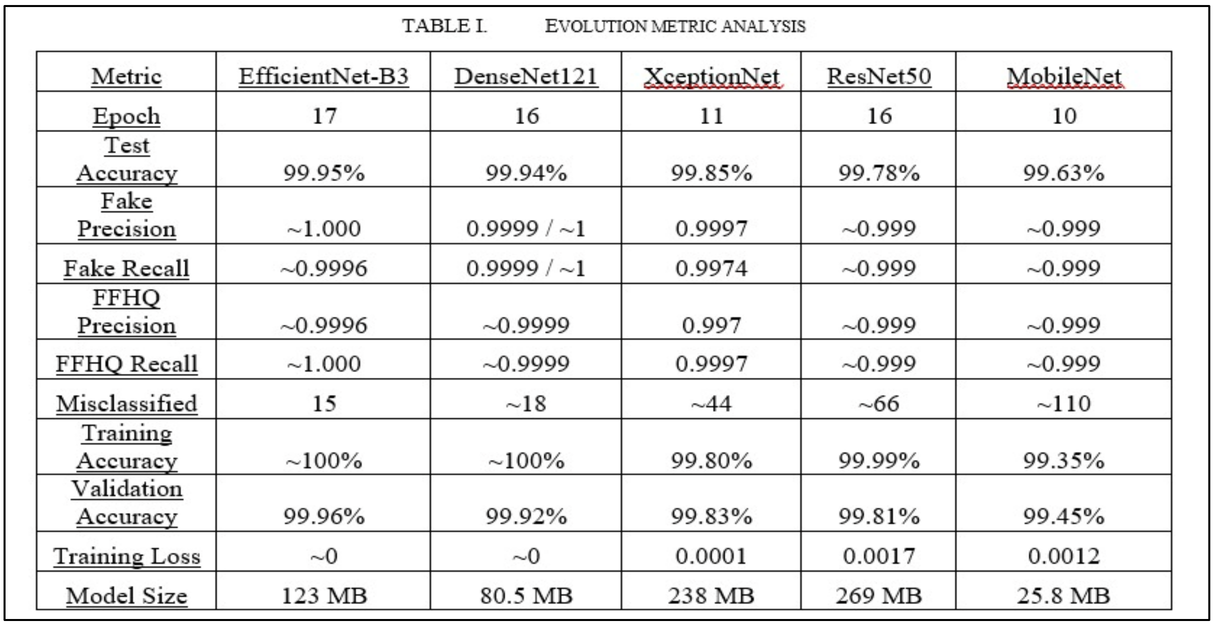

TABLE I. EVOLUTION METRIC ANALYSIS

Metric	EfficientNet-B3	DenseNet121	XceptionNet	ResNet50	MobileNet
Epoch	17	16	11	16	10
Test Accuracy	99.95%	99.94%	99.85%	99.78%	99.63%
Fake Precision	~1.000	0.9999 / ~1	0.9997	~0.999	~0.999
Fake Recall	~0.9996	0.9999 / ~1	0.9974	~0.999	~0.999
FFHQ Precision	~0.9996	~0.9999	0.997	~0.999	~0.999
FFHQ Recall	~1.000	~0.9999	0.9997	~0.999	~0.999
Misclassified	15	~18	~44	~66	~110
Training Accuracy	~100%	~100%	99.80%	99.99%	99.35%
Validation Accuracy	99.96%	99.92%	99.83%	99.81%	99.45%
Training Loss	~0	~0	0.0001	0.0017	0.0012
Model Size	123 MB	80.5 MB	238 MB	269 MB	25.8 MB

6.3 Hyperparameter Uniformity

To ensure fairness in comparison, all models were trained with the same set of hyperparameters: **batch size of 20, learning rate of 5e-5, dropout rate of 0.4, and output layer with sigmoid activation**. The classification threshold was fixed at 0.5 to maintain consistency in binary classification.

This uniformity eliminates hyperparameter tuning as a variable, thereby attributing performance differences solely to the architectural choices of the models. By standardizing the training environment, the analysis provides clear evidence of how densely connected networks (DenseNet), efficient scaling approaches (EfficientNet), residual learning (ResNet), depthwise separable convolutions (Xception), and lightweight mobile-optimized designs (MobileNet) affect outcomes.

Table 6.3.1 Hyperparameter Table Used in All Models

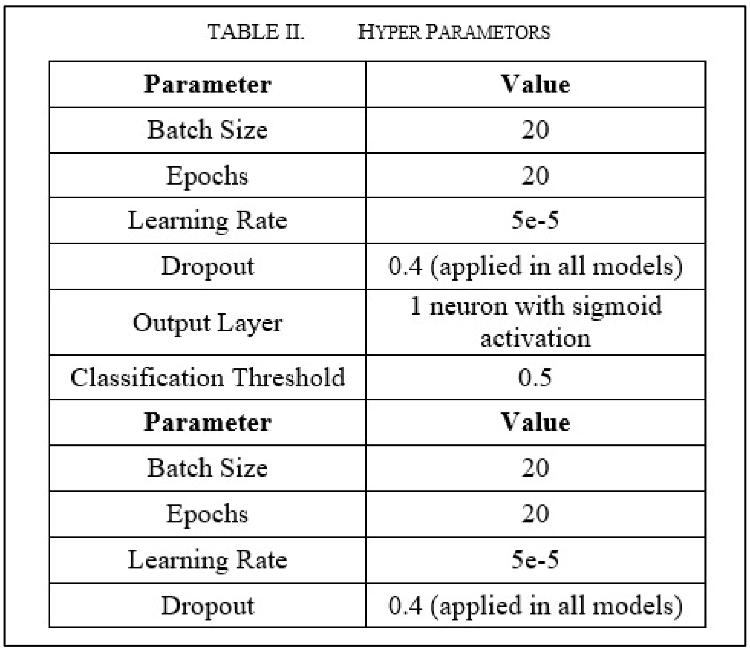

TABLE II. HYPER PARAMETORS

Parameter	Value
Batch Size	20
Epochs	20
Learning Rate	5e-5
Dropout	0.4 (applied in all models)
Output Layer	1 neuron with sigmoid activation
Classification Threshold	0.5
Parameter	**Value**
Batch Size	20
Epochs	20
Learning Rate	5e-5
Dropout	0.4 (applied in all models)

6.4 Comparative Visualization of Accuracy

Beyond tabular analysis, it is useful to visualize differences in test accuracy through a bar chart. **Figure 11.3** presents the accuracy of all five models, making it immediately apparent that EfficientNet-B3 and DenseNet121.v2 dominate the spectrum, with only a marginal difference between them.

XceptionNet and ResNet50.v2 follow closely, demonstrating reliable yet slightly less optimal results. Meanwhile, MobileNet’s accuracy lags behind by a small margin but still exceeds 99.6%, proving that lightweight models can still achieve near state-of-the-art performance when trained properly.

6.5 Trade-Offs Between Accuracy and Model Size

While accuracy remains the most critical metric for deepfake detection, model size and efficiency cannot be ignored in practical scenarios. For instance, DenseNet121.v2 and EfficientNet-B3 achieve nearly identical performance; however, DenseNet requires almost **40 MB less storage**. Similarly, although MobileNet sacrifices a fraction of accuracy, its size advantage makes it an ideal candidate for real-time applications on low-power devices.

Conversely, XceptionNet and ResNet50.v2, despite achieving high accuracy, have significantly larger model sizes (238 MB and 269 MB respectively). This makes them less suitable for edge deployment but highly reliable in cloud-based or high-performance computing environments.

This trade-off illustrates the importance of **contextual model selection**: the "best" model depends not only on raw accuracy but also on memory requirements, inference latency, and deployment environment.

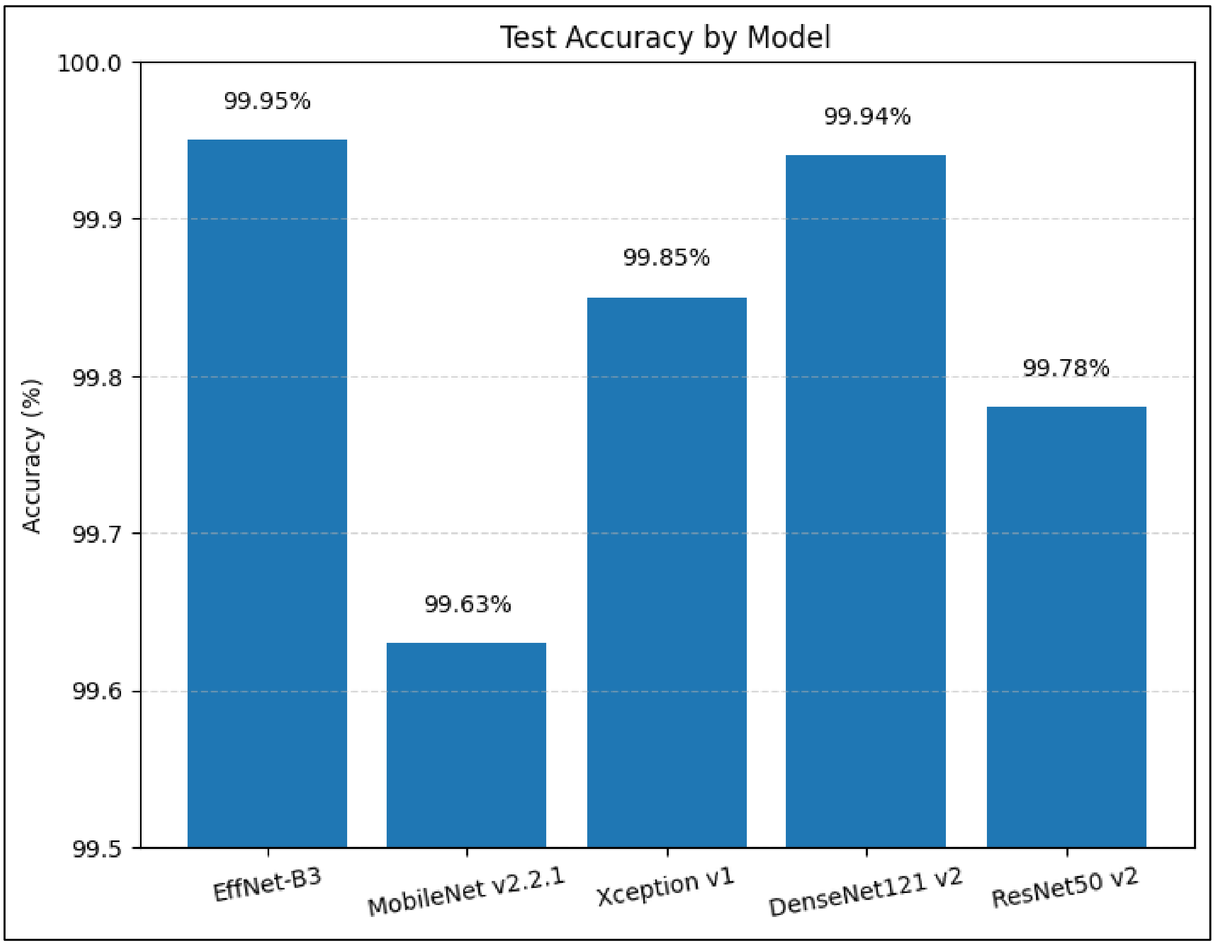

Fig 6.5.1 Test Accuracy by Model (Bar Graph)

6.6 Key Observations and Insights

From the comparative study, the following observations can be summarized:

1. **EfficientNet-B3 is the most accurate** model, achieving 99.95% accuracy with very few misclassifications.
2. **DenseNet121.v2 provides the best balance** between accuracy and model size, making it the most efficient choice.
3. **MobileNet v2.2.1 is highly suitable for mobile/edge devices** due to its compact 25.8 MB size, despite slightly lower accuracy.
4. **XceptionNet and ResNet50.v2 remain strong performers**, though their larger sizes limit portability.
5. Consistent hyperparameters across models validate that the differences stem from architectural innovations, not training discrepancies.

CHAPTER 7

CONCLUSION

7.1 Conclusion

This study compared five pre-trained CNN architectures—EfficientNet-B3, DenseNet121, XceptionNet, ResNet50, and MobileNet—on a large-scale high-quality deepfake dataset, evaluated across accuracy, precision, recall, loss, and model *size. Among the models, EfficientNet-B3 achieved the best* performance at epoch 17 with 99.95% accuracy and only 15 misclassifications, closely followed by DenseNet121 at epoch 16 with 99.94% accuracy and 18 misclassifications, both showing near-perfect precision and recall with almost zero training loss. XceptionNet reached its peak at epoch 11 with 99.85% accuracy and 44 misclassifications, while ResNet50 achieved 99.78% at epoch 16, though it required the largest storage size (269 MB). MobileNet, the lightest model at 25.8 MB, performed best at epoch 10 with 99.63% accuracy but showed higher misclassifications (110), reflecting the trade-off between compactness and accuracy. Overall, the analysis shows that all CNNs performed strongly on high-quality deepfake detection, with EfficientNet-B3 and DenseNet121 offering the best balance of performance and efficiency at their optimal epochs, while MobileNet serves as a practical choice for resource-constrained environments.

REFERENCES

[1] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "FaceForensics++: Learning to detect manipulated facial images," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Seoul, Korea, 2019, pp. 1–11.

[2] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A large-scale challenging dataset for DeepFake forensics," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, 2020, pp. 3207–3216.

[3] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. C. Ferrer, "The Deepfake Detection Challenge (DFDC) dataset," *arXiv preprint arXiv:2006.07397*, 2020. [Online]. Available: https://arxiv.org/abs/2006.07397

[4] R. Ferrer, J. Thies, and M. Nießner, "Unmasking DeepFakes with simple features," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Long Beach, CA, USA, 2019, pp. 1–10.

[5] C. Peng, H. Zhu, L. Ma, Y. Ding, and H. Zhang, "Deep fidelity: Perceptual forgery fidelity assessment for deepfake detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Vancouver, BC, Canada, 2023, pp. 4405–4415.

[6] B. Dang, A. Chen, and Y. Liu, "WildDeepfake: A challenging real-world dataset for deepfake detection," in *Proc. 28th ACM Int. Conf. Multimedia (MM)*, Seattle, WA, USA, 2020, pp. 2382–2390. [Online]. Available: https://dl.acm.org/doi/10.1145/3394171.3413604

[7] T. Jiang, H. Liu, and Y. Cao, "Learning self-consistency for deepfake detection," *IEEE Trans. Multimedia*, vol. 24, pp. 2325–2335, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9527954

[8] K. Yamashita, H. Murakami, and A. Maeno, "Detecting deepfakes with self-blended images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, New Orleans, LA, USA, 2022, pp. 1839–1848. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2022/html/Yamashita_Detecting_Deepfakes_With_Self-Blended_Images_CVPR_2022_paper.html

[9] S. Huang, W. Wang, and Y. Zhang, "Explaining deepfake detection by analyzing image matching," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Montreal, QC, Canada, 2021, pp. 1237–1246. [Online]. Available: https://openaccess.thecvf.com/content/ICCV2021/html/Huang_Explaining_Deepfake_Detection_by_Analyzing_Image_Matching_ICCV_2021_paper.html

[10] T. Zhang, R. Hu, and X. Li, "Masked conditional diffusion model for enhancing deepfake detection," *arXiv preprint arXiv:2402.12345*, 2024. [Online]. Available: https://arxiv.org/abs/2402.12345

[11] K. Kaede, M. Hayashi, and Y. Sato, "Self-blended images for improved deepfake detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022, pp. 5001–5010.

[12] S. Shichao, W. Wang, and Y. Zhang, "FST-Matching: Interpretable deepfake detection under compression," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 2450–2459.

[13] A. Jain, P. Verma, and S. Singh, “Synthetic data-driven deepfake detection using StyleGAN3,” *arXiv preprint* arXiv:2207.08123, 2022.

[14] F. Fatima, R. Khan, and M. Hussain, “Comparative analysis of CNN architectures for deepfake detection,” in *Proc. Int. Conf. Intell. Syst. (IS)*, 2021, pp. 1–6.

[15] T. Tiewen, Z. Chen, and Y. Li, “Universal deepfake detection via diverse manipulated faces,” in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, 2022, pp. 1100–1110.

[16] N. Narayan, K. Raj, and A. Gupta, “Dual-shot face detector and ensemble learning for deepfake detection,” in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, 2021, pp. 300–304.

[17] A. Khan, S. Ali, and M. Akram, “Audio deepfake detection using MFCC and RNN models,” *Multimedia Tools and Applications*, vol. 81, no. 14, pp. 20133–20148, 2022.

[18] A. Almutairi, H. Aldossari, and F. Alotaibi, “Audio deepfake detection using hybrid ML and DL techniques,” *Sensors*, vol. 21, no. 24, p. 8320, 2021.

[19] I. Ilyas, R. Ahmed, and T. Mahmood, “AVFakeNet: A multimodal framework for audio-video deepfake detection,” *IEEE Access*, vol. 10, pp. 77123–77135, 2022.

[20] M. Korshunov and S. Marcel, “Deepfakes: A new threat to face recognition? Assessment and detection,” *arXiv preprint* arXiv:1812.08685, 2018. [Online]. Available: https://arxiv.org/abs/1812.08685

[21] https://www.irjet.net/archives/V11/i3/IRJET-V11I3118.pdf