A MINOR PROJECT REPORT ON

Unmasking The Unreal : Deepfake Detection Framework Using Deep Learning

IN PARTIAL FULFILLMENT OF

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

DEPARTMENT OF

ELECTRONICS AND COMMUNICATION ENGINEERING

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA (U.P.)

Submitted by:

Shiv Vendra Singh (23102012)

Anannya Priyadarshini Sahoo (23102200)

Under the guidance of:

Prof. Vineet Khandelwal

Contents

CERTIFICATE

This is to certify that the project titled **"UNMASKING THE UNREAL: DEEPFAKE DETECTION FRAMEWORK USING DEEP LEARNING"** is a genuine work submitted by **Shiv Vendra Singh (23102012)** and **Annanya Priyadarshini Sahoo (23102200)**, students of the Bachelor of Technology degree in Electronics and Communication Engineering at Jaypee Institute of Information Technology, Noida, under the guidance and supervision of **Prof. Vineet Khandelwal**.

The project has been carried out as a part of the Minor Project (V Semester) in partial fulfilment of the requirements for the award of the Bachelor of Technology degree. This work is an original piece of research and has not been submitted elsewhere for any other purpose. It is hereby confirmed that the work reported in this project is original and was performed under my supervision. We also acknowledge that this project has been reviewed and approved for presentation.

Vineet Khandelwal
Professor
Department of Electronics and Communication Engineering
JIIT Noida

DECLARATION

We hereby declare that the Minor Project Report entitled **"UNMASKING THE UNREAL: DEEPFAKE DETECTION FRAMEWORK USING DEEP LEARNING"**, submitted to Jaypee Institute of Information Technology in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Electronics and Communication Engineering, is a record of our original work carried out under the supervision of **Prof. Vineet Khandelwal**.

We further declare that this report has not been submitted to any other university or institute for the award of any degree or diploma.

Date: November 25, 2025

Names:
Shiv Vendra Singh (23102012)
Annanya Priyadarshini Sahoo (23102200)

Under the Supervision of:
Prof. Vineet Khandelwal

ACKNOWLEDGEMENT

If words are considered a symbol of approval and a token of appreciation, then let these words play the heralding role in expressing our gratitude. The satisfaction that accompanies the successful completion of any task would be incomplete without acknowledging the people whose ceaseless cooperation made it possible, and whose constant guidance and encouragement crown all efforts with success.

We are deeply grateful to our project mentor, **Prof. Vineet Khandelwal**, for his guidance, inspiration, and constructive suggestions throughout the course of this project. His insights, new ideas, and consistent encouragement helped us navigate multiple technical and conceptual aspects of our work, enabling us to improve the quality and depth of this project.

We also extend our heartfelt thanks to our institution, Jaypee Institute of Information Technology, and to all faculty members of the Department of Electronics and Communication Engineering, whose support and academic environment made this project a reality.

Chapter 1
Overview of the Project

Introduction

This chapter lays the intellectual and contextual foundation for the entire project. In a world where synthetic media evolves faster than the mechanisms designed to contain it, deepfake detection has moved from being an academic curiosity to an urgent technological necessity. This chapter begins by exploring the motivation behind undertaking this work—rooted not only in the rapid rise of generative models, but also in the societal vulnerabilities they expose. It reflects on how deepfakes challenge notions of trust, authenticity, and digital integrity, and why building reliable detection systems is no longer optional but essential.

To situate the project within the broader scientific landscape, the chapter presents a concise review of key research directions that have shaped the development of deepfake forensics. These works illuminate both the strengths and the shortcomings of existing methodologies, providing the critical perspective needed to refine our own problem-solving approach. The discussion then moves to the real-world challenges inherent in detecting manipulated content—ranging from dataset imbalance and computational constraints to the sophistication of modern deepfake generation pipelines.

Following this, the objectives of the project are articulated clearly and strategically. They outline the guiding principles behind the model architecture, the preprocessing pipeline, the training philosophy, and the practical constraints that shaped the final design. Together, these objectives form the roadmap that drives all technical decisions throughout the study.

The chapter concludes with the organisation of the report, offering readers a structured preview of how the research unfolds across subsequent chapters. This introductory framework ensures that readers enter the technical core of the work

with clarity, context, and a coherent understanding of the project's motivation, scope, and direction.

1.1 Motivation

The motivation for this work arises from the convergence of several compelling forces — social urgency, technological acceleration, ethical responsibility, and scientific challenge. Deepfakes have rapidly moved from being experimental novelties to highly consequential threats. Fabricated political speeches can reshape public sentiment, synthetic news videos can mislead millions before the truth emerges, and non-consensual manipulated content can devastate an individual's dignity and mental well-being. At a time when misinformation already overwhelms digital spaces, deepfakes escalate the crisis of trust, turning truth verification into a race against deception.

Technologically, the challenge is no less fascinating. While many existing detection methods analyze single frames in isolation, deepfakes today are crafted with increasingly sophisticated generative models that attempt to mimic human appearance, expression, and texture across time. Yet, even the strongest generative systems struggle with the finer details — subtle inconsistencies in lighting, facial structure, expression transitions, or temporal coherence. These tiny mismatches, often invisible in a single frame, become detectable when processed through a strong feature extractor like EfficientNet, which can capture irregularities in high-level patterns such as facial geometry, texture uniformity, eye movement consistency, and frame-to-frame stability.

The motivation is also shaped by ethical and societal responsibility. As creation tools grow stronger, the burden of safeguarding truth increasingly lies with technological countermeasures. This project stands at the intersection of deep learning, digital forensics, privacy protection, and AI safety, driven by the conviction that technology must defend individuals, not endanger them. The goal is not just to classify deepfakes but to contribute to a larger ecosystem that protects authenticity, consent, and human dignity.

Ultimately, the inspiration behind this work stems from both intellectual cu-

riosity and moral intention: to understand the evolving nature of synthetic media, to design a model that remains robust as generative systems advance, and to contribute a meaningful tool in the broader effort to preserve trust in an increasingly artificial world.

1.2 Literature Review

Deepfake detection has been an active area of research over the past few years due to the growing misuse of manipulated visual content and its potential impact on security, privacy and public trust. Researchers have explored different approaches—from optical-flow-based feature engineering to deep convolutional networks and frequency-domain analysis—to distinguish authentic faces from manipulated ones. This section reviews the most relevant research studies that contributed to shaping the methodology of the present work. Each study is discussed in terms of its objective, methodology, key outcomes and identified limitations. The review also highlights how the shortcomings in existing work motivated the design of a lightweight, CPU-optimized deepfake detection framework suitable for constrained computational environments.

1.2.1 Review of Existing Deepfake Detection Approaches

(a) OptiFake – Optical Flow Extraction for Deepfake Detection using Ensemble Learning Technique (Vashishtha et al., 2024)

Overview. OptiFake introduces a motion–centric deepfake detection framework designed to capture temporal inconsistencies in manipulated facial videos. The underlying idea is that even when frames appear visually convincing, deepfake generation pipelines often fail to produce natural and smooth inter-frame transitions. These subtle irregularities—such as abnormal blinking, inconsistent jaw motion, or texture jitter—become more evident when analyzed through optical flow. OptiFake leverages this insight by converting motion fields into visual flow maps and learning discriminative patterns using an ensemble of deep and classical classifiers.

Methodology. The pipeline begins with face extraction using MediaPipe's detector to isolate regions of interest. Dense optical flow is computed between consecutive frames using Farnebäck's algorithm, generating motion-vector fields that encode direction and magnitude. The resulting flow maps are transformed into color-coded images and fed into an ensemble of models comprising CNNs, Random Forest, and XGBoost. A meta-classifier integrates their outputs for final prediction.

Key Findings.

- Captures temporal artifacts effectively, especially in low-quality deepfakes.
- Ensemble learning improves robustness across varied manipulation types.
- Optical-flow visualizations highlight abnormalities in eye, cheek, and jaw regions.
- Outperforms texture-only models when videos contain compression artifacts.

Limitations.

- Dense optical-flow computation is computationally expensive and storage-heavy.
- Poor performance on high-quality deepfakes with smooth temporal transitions.
- Not suitable for CPU-only or low-resource environments.
- Ensemble complexity may lead to dataset-specific overfitting.

(b) A Symbolic-Dynamics-Based Approach for Deepfake Detection (Singh et al., 2024)

Overview. This work proposes a symbolic-dynamics-based method that models the temporal evolution of facial appearance as symbolic sequences. Instead of relying on pixel-level patterns or deep feature extraction, the approach

discretizes temporal changes in facial regions into symbols that represent evolving facial dynamics. Genuine videos follow stable temporal patterns, whereas deepfakes often exhibit abrupt or irregular transitions due to synthesis artifacts.

Methodology. Faces are first extracted and partitioned into regions. Pixel trajectories in these regions are converted into symbolic sequences using quantization rules. Complexity measures such as entropy or symbolic divergence quantify the temporal behavior of these sequences. These symbolic feature vectors are then classified using machine-learning models including SVM, Random Forest, and XGBoost.

Key Findings.

- Reduces dimensionality significantly, improving computational efficiency.
- More robust to noise, compression, and low-quality inputs.
- Classical ML models allow faster training and inference compared to deep networks.

Limitations.

- Symbolic encoding oversimplifies texture details, limiting performance on high-quality deepfakes.
- Partitioning and quantization rules are dataset-dependent.
- Performance degrades on unseen manipulation types due to limited generalization.
- Still computationally heavy for long video sequences.

(c) Four-Filtered Residual CNN (FFR-CNN) for Deepfake Detection (Agarwal & Gupta, 2022)

Overview. The FFR-CNN model enhances deepfake detection by amplifying high-frequency signals—edges, noise patterns, and residual textures—where manipulation artifacts typically reside. By applying four high-pass filters prior

to CNN processing, the method exposes subtle inconsistencies that are otherwise difficult to capture with raw RGB inputs.

Methodology. Four high-pass filters generate residual maps highlighting different anomaly types. These maps, combined with the original image, form a multi-channel input fed into a residual CNN with skip connections for improved gradient flow. The model learns artifact-specific features and performs binary classification using fully connected layers. Performance is evaluated on FaceForensics++ under varying compression settings.

Key Findings.

- High-frequency residuals significantly enhance artifact detection.
- Robust against moderate compression where many cues vanish.
- Residual connections improve stability and convergence during training.

Limitations.

- Multi-filter preprocessing increases computational overhead.
- Heavy GPU dependence restricts usage in low-resource environments.
- Limited cross-dataset generalization; filters may overfit to specific artifact types.
- Increased storage and I/O overhead due to multi-channel residual maps.

(d) FaceForensics++ Benchmark and Manipulation Analysis (Rössler et al., 2019)

Overview. FaceForensics++ provides a large-scale benchmark dataset widely used for deepfake detection research. It includes real and manipulated videos generated using four major techniques—DeepFakes, FaceSwap, Face2Face, and NeuralTextures—making it a foundational evaluation resource. The study also analyzes the impact of compression, manipulation type, and generalization settings on detection performance.

Methodology. Thousands of videos are processed and released under multiple compression levels (raw, lightly compressed, heavily compressed). Baseline detectors, including XceptionNet, are trained and evaluated across intra-dataset and cross-manipulation settings. Human evaluation is also included for comparison with machine classifiers.

Key Findings.

- CNNs achieve high accuracy when trained and tested on consistent manipulation types.
- Heavy compression significantly degrades detection performance.
- Cross-manipulation generalization is difficult; detectors rely heavily on manipulation-specific cues.
- XceptionNet becomes the de facto baseline for subsequent research.

Limitations.

- Dataset does not reflect modern deepfakes built using newer techniques.
- High GPU requirements make training difficult on CPU-only systems.
- Primarily focuses on face-swap methods; limited coverage of reenactment and GAN-based manipulations.
- Large image sizes increase storage, preprocessing cost, and memory usage.

(e) EfficientNet Architecture for Lightweight Vision Tasks (Tan & Le, 2019)

Overview. EfficientNet introduces compound scaling—a balanced strategy to scale network width, depth, and resolution simultaneously. EfficientNet-B0, the smallest variant, offers strong accuracy with minimal computational cost, making it ideal for resource-constrained settings. Although not designed specifically for deepfake detection, its lightweight architecture has become a popular backbone for efficient vision pipelines.

Methodology. The architecture uses inverted residual blocks and squeeze-and-excitation modules. A two-stage process is used: (1) Neural Architecture Search identifies an optimized baseline model; (2) compound scaling rules generate the EfficientNet family (B0–B7). Pretrained ImageNet weights are used for transfer learning across tasks, including deepfake detection.

Key Findings.

- Achieves state-of-the-art accuracy with fewer parameters than ResNet or Inception.
- EfficientNet-B0 is suitable for CPU environments and mobile/embedded devices.
- Pretrained weights transfer well to tasks involving subtle texture analysis.

Limitations.

- Designed for static images; lacks temporal modeling for video deepfakes.
- Sensitive to poor preprocessing or inconsistent face cropping.
- Performance drops for low-quality or highly compressed frames.
- Larger variants (B3–B7) are impractical for CPU-only systems.

1.2.2 Synthesis of Literature and Identified Research Gaps

Across the reviewed studies, several consistent limitations emerge:

- Most approaches depend heavily on GPU resources, making them unsuitable for CPU-only settings such as Kaggle free tier.
- Optical-flow and symbolic-dynamics methods struggle with high-quality deepfakes that maintain temporal consistency.
- CNN and ensemble-based models scale poorly due to large storage needs and computational overhead.
- Many detectors generalize poorly to unseen manipulation types.

- Advanced architectures require extensive preprocessing, memory and hardware support.

These limitations highlight a strong need for a lightweight, CPU-optimized deepfake detection framework—one that uses minimal preprocessing, extracts essential features directly from facial regions and remains stable under strict computational limits. The present work addresses these gaps by designing a pipeline based on EfficientNet-B0 [3], selective frame sampling and simplified preprocessing, enabling accurate deepfake detection without high-end GPUs.

1.3 Existing Challenges

Despite the rapid growth of deepfake detection research, several challenges still limit the reliability and practicality of existing techniques. Many current methods depend heavily on handcrafted or model-specific features, which restrict their ability to generalize across different types of manipulations. For example, the OptiFake model proposed by Vashishtha et al. (2024)[10] relies on optical flow-derived motion patterns and an ensemble classifier, but the approach still presents certain limitations. First, although optical flow helps capture temporal inconsistencies, it remains sensitive to lighting changes, compression artifacts, and fast camera movements. The authors acknowledge that their model is trained and tested only on the FaceForensics++ subsets (DeepFake and FaceSwap), which limits cross-dataset generalization, and the highest accuracy achieved (86%) still leaves room for misclassification in real-world conditions references

. Another drawback is that optical-flow-based extraction is computationally expensive when applied to high-resolution videos, making real-time deployment challenging. Their ensemble model also depends on multiple deep neural networks, which increases inference time and requires significant GPU resources.

A second challenge comes from the paper "Symbolic Dynamics Based Deepfake Detection" (2024)[9], which also explores handcrafted statistical descriptors and dynamical system features. Although the method shows promise for spotting subtle inconsistencies in facial movements, its performance depends strongly on controlled environments and high-quality inputs. The paper highlights

that symbolic dynamic features become less stable when videos are noisy, compressed, or contain occlusions such as glasses, facial hair, or side-profile angles. Moreover, the models evaluated in this study did not include large-scale real-world benchmarks, limiting their applicability outside laboratory-like settings. The authors further note that symbolic patterns extracted from facial signals can vary widely across individuals, emotions, and lighting conditions, raising concerns about false positives in diverse user populations .

Across both studies, a deeper, common challenge emerges: most deepfake detectors focus on specific types of manipulations, such as identity swap or expression reenactment, and struggle when confronted with unseen, newer deepfake generation techniques. Techniques like GAN-based blending, diffusion-model synthesis, or neural texture transfer evolve far more rapidly than detection models. The papers also acknowledge that many existing datasets—such as FaceForensics++ or DeepFake-TIMIT—lack diversity in terms of ethnicity, environment, camera motion, and background complexity. This creates a significant domain gap: models trained in controlled settings often fail when exposed to social-media-quality videos that are noisy, low-resolution, heavily compressed, or edited multiple times. Additionally, face-centric methods fail when the face is partially visible, turned sideways, or affected by occlusions.

Another major challenge is that most papers treat frames or features independently, overlooking the temporal coherence of a video. For example, frame-based CNN detectors may perform well on per-image classification, but they ignore the transitions between frames where deepfake artifacts often hide. Even ensemble or capsule-based networks in the reviewed papers struggle to capture long-range temporal patterns effectively. Finally, both studies recognize a growing adversarial threat: modern deepfake generators are actively optimized to bypass detection networks, which means detectors must evolve continuously to remain effective.

In summary, although existing research has introduced valuable ideas—optical flow, ensemble learning, symbolic dynamics, CNN-based classification—each method carries practical weaknesses such as poor generalization, dataset dependency, computational overhead, vulnerability to compression, and inability to

handle newer manipulation strategies. These limitations highlight the need for a more adaptive, efficient, and robust detection framework, forming the foundation for the system developed in this project.

1.4 Objectives of the Project

1.4.1 Ensure Stable and Consistent Face Extraction Using MediaPipe

A key objective of the project is to construct a preprocessing pipeline capable of extracting high-quality facial regions from videos under real-world constraints. MediaPipe's lightweight detector is leveraged to achieve stable bounding-box tracking, confidence-based filtering, and padding-controlled cropping. The pipeline extracts a fixed number of frames per video, ensuring dataset uniformity while handling blur, illumination changes, head movement, and compression noise. This objective establishes the foundation on which the classification model depends, as reliable face extraction directly influences downstream accuracy.

1.4.2 Develop a Lightweight and CPU-Optimized CNN Classifier

The project aims to design a classification model that balances accuracy with computational efficiency, making it deployable even in non-GPU environments. EfficientNetB0 is chosen as the backbone due to its compound-scaling principle, high representational strength, and low parameter count. Only the top layers are unfrozen to enable selective fine-tuning, while regularization, dropout, and batch normalization are incorporated to prevent overfitting. The objective is to achieve competitive detection performance while keeping computation time, memory usage, and training instability under control.

1.4.3 Establish a Robust Training Strategy to Prevent Prediction Collapse

Deepfake models often suffer from training instabilities such as prediction collapse, where outputs converge toward a single class irrespective of input. To prevent this, the project integrates a minimal-overhead prediction monitoring callback, early stopping on AUC, learning-rate reduction on plateaus, and an ultra-

minimal data generator that loads one image at a time from disk. These training-control mechanisms ensure long-duration CPU runs remain stable, memory-safe, and resistant to drift, thereby improving the reliability of the final model.

1.4.4 Implement an Ultra-Minimal and Memory-Efficient Data Pipeline

Another objective is to design a data handling framework compatible with large-scale datasets despite restricted computational resources. This includes streaming face images directly from disk, using batch sizes as low as eight, avoiding in-memory video storage, and simplifying augmentations to CPU-friendly transformations. The goal is to handle tens of thousands of samples without triggering kernel crashes, outperforming conventional methods in environments with limited RAM availability.

1.4.5 Build a Generalizable, Real-World-Ready Deepfake Detection Framework

The final objective is to develop a detection system that generalizes effectively to varied real-world conditions—including low-resolution videos, compression artifacts, lighting variations, and diverse face structures. By integrating reliable face extraction, a lightweight CNN, stable training controls, and careful preprocessing, the project seeks to contribute a scalable and accessible deepfake detection pipeline. This framework aligns with broader goals of preserving digital authenticity, safeguarding public trust, and mitigating the misuse of synthetic media for manipulation, fraud, and misinformation.

1.5 Organisation of the Report

The remainder of this report is structured to guide the reader through the complete development of the proposed deepfake detection system in a logical and cohesive manner. Chapter 2 presents the Background Study, offering the theoretical and technical foundations necessary to understand the motivations, design choices, and scientific context of the project. It reviews the evolution of gen-

erative models, forensic techniques, preprocessing strategies, and architectural frameworks that inform our methodology.

Chapter 3 details the System Design and Methodology, describing the full pipeline from preprocessing to model construction. It explains the face extraction strategy, the rationale behind adopting a lightweight EfficientNet backbone, the memory- efficient training design, and all supporting components that make the system CPU- friendly and scalable.

Chapter 4 presents the Results and Discussion, including dataset characteristics, evaluation metrics, performance visuals, and comparisons with existing techniques. This chapter interprets the model's behaviour, strengths, and limitations through comprehensive experimental evidence.

Finally, Chapter 5 provides the Conclusion and Future Scope, summarising key findings while outlining promising directions for further improvement, deployment, and real-world applicability. Together, these chapters form a unified narrative that traces the system's development from conceptual motivation to practical outcome.

Chapter 2
Foundational Study

Introduction

Deepfake technology has rapidly transformed from a niche research curiosity into one of the most formidable challenges of the digital age. As synthetic media grows increasingly indistinguishable from authentic human imagery, societies worldwide face disruptions in truth, trust, security, and personal identity. Within this rapidly shifting landscape, a deepfake detection system is no longer a specialized forensic tool—it is a technological shield defending the credibility of information and the dignity of individuals. Understanding how deepfakes are built, how artifacts emerge, and how computational systems dissect their anomalies becomes the essential foundation on which any modern detection framework must stand.

This chapter serves as the intellectual backbone of the entire project. It brings together the core scientific principles, methodological pillars, and technological innovations that shape today's deepfake research. The discussion begins with the generative engines—GANs and diffusion models—that power the creation of synthetic faces, offering insight into how they mimic human appearance with ever-increasing precision. It then moves into residual-based forensics and CNN-driven feature extraction, exploring how modern detectors learn to spot imperfections invisible to the human eye. The role of data augmentation is examined as a critical strategy for enhancing generalization, followed by a study of lightweight architectures such as EfficientNet, which balance accuracy with computational efficiency. Further, the chapter highlights the foundational importance of robust facial preprocessing through frameworks like MediaPipe and concludes with the landmark contribution of FaceForensics++, a dataset that has shaped the very vocabulary of deepfake research.

Together, these interconnected themes present a panoramic view of the sci-

entific ecosystem surrounding deepfake detection. By understanding the evolution of generative models, forensic techniques, architectural choices, and dataset-driven insights, this chapter prepares the reader to appreciate the rationale behind the design decisions in our proposed methodology. It establishes the theoretical grounding, contextual relevance, and research continuity necessary for constructing a reliable, efficient, and future-ready deepfake detection pipeline.

2.1 Background Study

2.1.1 Generative Adversarial Networks (GANs) and Diffusion Models in Deepfake Generation

Deepfake creation today is dominated by two families of generative models—Generative Adversarial Networks (GANs) and diffusion models—both of which have revolutionized how synthetic facial content is produced. GANs operate by pitting two neural networks against each other: a generator that creates fake images and a discriminator that learns to differentiate between real and synthetic samples. Through this adversarial game, the generator gradually learns to produce outputs that are convincing enough to fool the discriminator, resulting in highly realistic artificially generated faces. These models excel because of their ability to capture complex facial textures, shadows, expressions, and micro-level details that earlier generative models struggled to reproduce.

Diffusion models, on the other hand, follow an entirely different philosophy. Instead of directly generating images, they start with random noise and iteratively "denoise" it through hundreds of carefully learned steps. This controlled reverse-diffusion process allows them to achieve exceptional smoothness, continuity, and structural fidelity in the final output. Diffusion-based deepfakes are often cleaner, more stable, and more temporally coherent, which makes detection even more challenging.

Understanding both these generative mechanisms is crucial because each introduces unique artifacts, inconsistencies, and temporal distortions—subtle fingerprints that effective detection systems must learn to capture [1].

2.1.2 Residual-Based Preprocessing Approaches

Residual forensics represents one of the earliest and most intuitive approaches to detecting image manipulations. Instead of looking at the visual content itself, residual methods attempt to unveil the "hidden noise" left behind during editing. When an image is resampled, filtered, compressed, or blended, it leaves microscopic traces in the high-frequency domain—patterns that humans cannot see but neural networks can learn to identify.

Agarwal and Gupta's Four Filtered Residue CNN (FFR–CNN) is a notable example that applies a collection of filters—average, Gaussian, Laplacian, and median—to extract different types of residual signatures. These filtered maps emphasize inconsistencies in texture, unnatural pixel transitions, and irregular noise patterns introduced during manipulation. While this method did not produce strong results on our dataset—likely due to heavy compression and limited noise artifacts—it still shaped our understanding of artifact-centric forensics and highlighted the importance of preprocessing in amplifying otherwise invisible cues [2].

2.1.3 CNN-Based Approaches for Spatiotemporal Feature Extraction

Convolutional Neural Networks (CNNs) have long been the workhorse of deepfake detection because of their ability to analyze spatial patterns, texture inconsistencies, blending artifacts, and structural distortions. Beyond spatial analysis, CNNs also capture frame-to-frame irregularities when applied to video datasets—making them useful for identifying temporal anomalies such as abrupt expression changes or unnatural eye movements.

EfficientNet, in particular, stands out because of its compound scaling strategy, which systematically balances network depth, width, and resolution. This allows it to achieve high accuracy while maintaining computational efficiency—an essential requirement for CPU-based training environments such as the one used in this project. EfficientNetB0, even in its minimal form, is powerful enough to detect subtle texture inconsistencies, skin abnormalities, facial alignment errors,

and lighting mismatches introduced during deepfake generation [3].

2.1.4 Role of Data Augmentation in Improving Model Generalization

Deep learning models thrive on diversity in training samples, and data augmentation acts as a crucial tool for artificially expanding this diversity. Perez and Wang demonstrated that augmentation techniques—ranging from rotation, noise addition, and blur to brightness changes and flipping—force models to generalize rather than memorize patterns. This is especially useful in deepfake detection, where training samples may be limited, imbalanced, or artificially clean compared to real-world media.

In this project, even with minimal augmentation (horizontal flipping and mild color variations), the model becomes better equipped to handle unpredictable lighting conditions, minor distortions, and natural variability in facial expressions. Augmentation thus plays a key stabilizing role, preventing overfitting and supporting robust performance [4].

2.1.5 EfficientNet as a Lightweight and High-Performance Architecture

EfficientNet models are widely celebrated for providing an exceptional balance between accuracy and computational cost. Their design leverages depthwise separable convolutions, which significantly cut down the number of parameters without compromising feature quality. Batool and Byun demonstrated how EfficientNetB3 can outperform several heavier CNN architectures while consuming far fewer resources—an insight that strongly aligns with the goals of this project.

Since this work aims to create a CPU-friendly deepfake detector, EfficientNetB0 becomes an ideal choice: lightweight, scalable, and still capable of extracting high-quality discriminative features from facial regions [5].

2.1.6 MediaPipe for Stable and Real-Time Face Landmark Extraction

Reliable face extraction lies at the heart of any deepfake detection pipeline. MediaPipe, developed by Google, provides a fast and accurate solution for real-time facial landmark detection, even in challenging scenarios involving low light, camera noise, partial occlusions, or head movement. The framework uses lightweight neural modules that can detect faces robustly without requiring GPU acceleration, making it ideal for large-scale preprocessing.

In this project, MediaPipe plays a critical role in stabilizing bounding boxes, applying temporal smoothing, and ensuring that only high-quality face crops are passed to the classifier. This dramatically reduces noise in the dataset and improves overall model performance [6].

2.1.7 Importance of Face-Centric Preprocessing (FaceForensics++)

One of the most influential contributions to deepfake research is the FaceForensics++ dataset, which demonstrated that manipulation artifacts are most prominent in the facial region rather than the background. Rössler et al. showed that focusing a classifier on tightly cropped face regions significantly improves detection performance because it removes background clutter and highlights artifact-rich areas such as eye boundaries, cheeks, lips, and blending edges.

This insight strongly supports the face-only preprocessing approach adopted in our project. By extracting consistent, centered, and clear face crops, the model is able to focus on the regions most susceptible to manipulation—leading to more reliable and interpretable predictions [7].

2.1.8 Importance of Evaluation Metrics and Visual Diagnostics in Deepfake Detection

A crucial dimension of deepfake research lies not only in designing effective models but also in evaluating them through comprehensive and meaningful performance metrics. Since deepfake detection involves distinguishing finely

crafted synthetic content from authentic human imagery, the evaluation process must capture subtle classification behaviours that simple accuracy alone cannot reveal. To ensure that the proposed model is both reliable and interpretable, multiple diagnostic tools and visualisations are employed in this project.

The confusion matrix forms the first layer of analysis, illustrating how the model handles both real and fake samples by summarizing true positives, false positives, true negatives, and false negatives. In deepfake detection—where misclassifying fake samples as real can carry severe consequences—this breakdown becomes vital for understanding class-specific vulnerabilities.

Beyond this, the Receiver Operating Characteristic (ROC) curve and the Area Under the Curve (AUC) provide a broader view of the model's discrimination capability across varying decision thresholds. AUC is especially valuable in forensic applications, as it remains robust even under class imbalance, a condition common in real-world datasets where authentic videos outnumber manipulated ones.

Complementing these visuals, the classification report presents precision, recall, and F1-score for each class, offering insight into how well the model balances sensitivity and specificity. Since deepfake detection models may suffer from overconfidence or prediction collapse, threshold–accuracy plots are also examined to guide optimal decision boundary selection. These plots reveal how prediction behaviour changes across thresholds and assist in identifying stability issues.

Together, these diagnostic tools form a holistic evaluation framework that enables deeper understanding of model behaviour, highlights strengths and weaknesses, and ensures that the system is dependable beyond raw accuracy metrics. Their use in this project aligns with established best practices in multimedia forensics and machine learning model validation [8].

Chapter Summary

This chapter explored the foundational concepts, technologies, and research pillars that inform the design of the proposed deepfake detection framework. It began by examining the generative engines—GANs and diffusion models—that drive modern deepfake creation, highlighting how adversarial learning and iterative denoising contribute to increasingly realistic synthetic facial content [1]. Their rapid evolution underscores the need for equally adaptive detection strategies.

Residual-based forensics were then discussed as one of the early approaches to manipulation analysis, where filtered residue maps amplify noise and artifact patterns left behind during editing [2]. Although limited in high-compression environments, these methods established the importance of preprocessing in revealing manipulation cues that are otherwise imperceptible.

The chapter further examined CNN-driven feature extraction, with EfficientNet emerging as a powerful yet lightweight architecture capable of detecting subtle spatiotemporal inconsistencies in facial textures [3]. Alongside model architecture, the role of data augmentation was shown to be essential for improving generalization, especially when training datasets are constrained or unevenly distributed [4].

A critical component of any deepfake system is reliable face-focused preprocessing. Tools like MediaPipe streamline this step by providing stable, real-time landmark detection even in variable conditions [6]. The chapter also highlighted the pivotal influence of FaceForensics++, which demonstrated that deepfake artifacts are most concentrated in the facial region, shaping the shift toward face-centric detection pipelines [7].

Finally, the chapter emphasized the importance of evaluation metrics and visual diagnostics. Confusion matrices, ROC curves, AUC, precision–recall scores, and threshold analyses offer a multi-dimensional understanding of model behaviour and reveal weaknesses that accuracy alone cannot capture [8].

Overall, this foundational study establishes the theoretical and practical

groundwork necessary for designing a robust, efficient, and resource-conscious deepfake detection framework. The insights presented here directly inform the architectural choices, preprocessing decisions, and training strategies implemented in the subsequent chapter on System Design and Methodology.

Chapter 3
System Design and Methodology

Introduction

This chapter presents the complete system design and the methodological framework used to develop *"Unmasking the Unreal: Deepfake Detection Framework Using Deep Learning"*. The primary aim of the project is to construct a stable, efficient, and computationally lightweight pipeline capable of detecting deepfake videos even under strict resource constraints, such as those imposed by CPU-only Kaggle Notebook environments.

Deepfake detection is inherently a multi-stage process. Raw video inputs cannot be directly used by machine learning models; instead, they must be transformed through a structured sequence of preprocessing steps. Since deepfake manipulations most strongly affect the facial region, the proposed system follows a face-centric pipeline. Each video is broken down into a fixed number of facial frames, these frames are normalized and preprocessed for consistency, and the resulting dataset is used to train a deep learning classifier capable of distinguishing real faces from manipulated ones.

During the early stages of experimentation, several ambitious methods were explored— including full-length video analysis, heavy augmentation, larger EfficientNet variants, and optical-flow–based temporal modeling. However, these approaches exceeded Kaggle's runtime, RAM, and disk usage limits, resulting in repeated session crashes. Through a series of targeted optimizations and design refinements, the final pipeline emerged as a highly stable CPU-optimized system built upon:

- **Balanced sampling of real and fake videos**
- **Minimal and stable frame extraction using MediaPipe**
- **EfficientNetB0 as the core CNN backbone**

- **A lightweight training configuration designed for long CPU runs**
- **Prediction-monitoring callbacks to detect and prevent model collapse**
- **Simplified data loading and evaluation mechanisms**

This chapter elaborates on the architecture of the system, detailing the rationale behind each design choice and explaining how the components—preprocessing, model construction, training strategy, and evaluation methodology—collectively form an efficient and reliable deepfake detection framework.

3.1 System Design

The design of the proposed deepfake detection framework is centred around building a lightweight, scalable, and resource-efficient system capable of identifying manipulated videos with high accuracy. The overall structure follows a modular flow, where each stage performs a clearly defined function—from dataset acquisition to face extraction, model training, and final prediction. This structured design ensures that the system remains stable even when executed on limited-compute environments such as Kaggle CPU runtimes.

The system takes the DeepFake Detection (DFD) dataset as its input. Since the original dataset contains a large number of videos and includes both real and manipulated content, the first stage focuses on organizing and reducing this data to an efficient, manageable size. To allow fair and unbiased training, an equal number of real (363 videos) and fake (363 videos) samples were selected. This balanced subset ensures that the model does not unintentionally lean towards one class during training.

Once the videos are selected, the system performs face-based preprocessing, as faces are the most informative regions for detecting deepfake artifacts. Instead of processing full videos—which is computationally expensive—the system extracts exactly 10 representative frames per video using evenly spaced intervals. These frames undergo fast and stable MediaPipe face detection, where each detected face is cropped, resized to 224×224 pixels, and saved. This minimal ap-

proach reduces storage footprint, avoids memory overflow on Kaggle, and allows the pipeline to scale smoothly.

The system then stores all detected face images inside a dedicated directory, serving as the final input to the training stage. A filename-based method is used to record the video ID and label, which later supports consistent, non-leaky dataset splitting. The complete preprocessed dataset contains 4,750 face images, distributed among training, validation, and test sets using a video-grouped split, ensuring that no frames from the same video appear in different partitions.

For model development, EfficientNet-B0 was selected due to its compact size and strong performance on image-based classification tasks. The system keeps most layers frozen and fine-tunes only the last few layers to reduce computational load. Additional components such as batch normalization, dropout regularization, and an L2-penalized dense layer stabilize training, prevent overfitting, and ensure smooth generalization on unseen samples.

Since the entire project was executed on Kaggle CPU, the design incorporates special optimizations:

- disabling GPU visibility to avoid unnecessary initialization overhead,
- using a batch size of 8,
- loading only a single image at a time within the generator,
- clearing memory frequently using Python's `gc` module,
- reducing augmentation to a minimal flip-based strategy.

These optimizations ensure that the model trains efficiently even within strict memory limits.

Finally, the trained model generates predictions on the test set using probability scores. A custom threshold-selection mechanism evaluates thresholds from 0.30 to 0.70 and selects the one that gives the highest accuracy. This helps balance precision and recall across both real and fake classes. The evaluation module produces a confusion matrix, ROC curve, accuracy plots, and class-wise metrics to analyze the model's behaviour.

Overall, the system design emphasizes stability, fairness, computational efficiency, and interpretability. Every stage—from dataset organization to model evaluation—has been engineered to work reliably with limited hardware while still maintaining strong detection capability against deepfake manipulations.

3.2 Model Illustration

The proposed deepfake detection framework is built around a lightweight yet highly capable deep learning architecture tailored specifically for constrained environments such as Kaggle's CPU-only execution mode. This section illustrates the internal structure of the implemented system, explaining each computational stage from raw video input to the final classification output. The intention is to provide both clarity for beginners and sufficient technical rigor for academic readers.

The system receives raw video files and processes them through four tightly integrated stages: face-based preprocessing, dataset formation, model construction, and prediction-driven training. Each stage has been deliberately optimized to handle large video files under restricted memory limits without triggering kernel crashes. The final architecture is the result of multiple iterations, each informed by practical constraints and empirical observations.

The pipeline begins with face-focused preprocessing, motivated by the fact that deepfake manipulation almost always targets the facial region. Instead of loading entire videos into memory—which repeatedly caused earlier notebook crashes—the system processes videos frame-by-frame using MediaPipe's face detection module. For each video, a fixed set of ten frames is sampled uniformly across its duration to maintain dataset consistency. MediaPipe detects the face in each frame, after which the bounding box is expanded slightly with padding to avoid overly tight crops. Only faces meeting a minimum resolution threshold are accepted to ensure that the model trains on crisp and meaningful information.

Each extracted face is resized to 224×224 pixels, matching the input requirements of the EfficientNet architecture. This resolution balances efficiency and detail preservation, enabling the network to learn texture cues essential for

detecting manipulations. Face images are stored as moderately compressed JPEG files, reducing disk usage while retaining the fine-grained artifacts characteristic of deepfakes. The output of this stage is a clean, memory-efficient, balanced dataset of uniformly processed face crops from real and fake videos.

With the dataset prepared, the model architecture is constructed using EfficientNetB0 as the primary feature extractor. EfficientNet is a family of convolutional neural networks known for their compound scaling strategy, which balances model depth, width, and input resolution. EfficientNetB0, the smallest variant, was chosen deliberately due to its excellent trade-off between accuracy and computational cost. Pretrained ImageNet weights are employed to provide strong feature initialization, reducing training time and improving stability. The top classification layers of EfficientNet are replaced with custom layers specifically structured for deepfake detection.

To avoid overfitting and to maintain computational efficiency, the majority of EfficientNetB0 remains frozen during training. Only the final ten convolutional layers are unfrozen, allowing the model to adapt high-level representations to deepfake-specific artifacts such as texture inconsistencies, lighting anomalies, and irregular blending patterns. This partial fine-tuning approach ensures flexibility without sacrificing stability.

The extracted features pass through a lightweight classifier composed of a global average pooling layer, batch normalization, dropout for regularization, and a fully connected dense layer of 128 units for expressive learning. A second batch normalization and dropout layer reinforce robustness against overfitting. Finally, a single-unit sigmoid activation outputs a probability indicating whether the face is real or manipulated.

Training the model under Kaggle's CPU constraints required a carefully engineered strategy. Rather than loading full batches of images—which would quickly exhaust memory—the system uses an ultra-minimal data generator that loads one image at a time. This ensures stable execution even for large datasets. Light augmentation in the form of horizontal flipping is applied to introduce minor variability without increasing computational overhead.

The training loop employs the Adam optimizer with a conservative learning rate to maintain control over weight updates. A validation-based early stopping mechanism halts training once improvement saturates, and a Reduce-on-Plateau scheduler lowers the learning rate when necessary for finer convergence. A custom prediction-variance monitor detects and prevents prediction collapse, a known issue where the model begins outputting near-constant values.

After training converges, the evaluation pipeline tests the model on strictly separated video-level samples to avoid information leakage. Each sample is passed through the model, an optimal decision threshold is determined, and key performance metrics—accuracy, AUC, precision, recall, and confusion matrix—are computed. These metrics provide a comprehensive picture of how effectively the model distinguishes between genuine and manipulated facial content.

In summary, the complete model architecture is a carefully engineered deepfake detection system designed under strict computational and memory limitations. By integrating robust preprocessing, balanced dataset construction, an optimized EfficientNetB0 backbone, and a memory-efficient training strategy, the pipeline delivers stable, interpretable, and reliable performance even within CPU-only environments.

3.3 Working Principle

The working principle of the proposed deepfake detection system describes how raw video samples pass through a sequence of carefully designed stages to generate a final prediction. Each stage is engineered to be computationally lightweight, reliable, and aligned with the constraints of CPU-only execution in Kaggle notebooks. This section explains the logical flow of the system in a clear and structured manner, focusing on how each module transforms the data and contributes to the final goal of identifying manipulated content.

The overall workflow begins by reading the input videos from the DeepFake Detection (DFD) dataset. Each video may contain hundreds of frames, but processing all of them is computationally inefficient. Instead, the system extracts a fixed number of frames that are evenly spaced throughout the video. This en-

sures that the model receives a consistent representation of each sample while avoiding bias toward longer or higher-frame-rate videos.

Once the frames are selected, the system identifies and crops the face present in each frame using the MediaPipe face-detection algorithm. This step is crucial because deepfake manipulations primarily alter facial texture, alignment, and consistency. By isolating only the face region and discarding the background, the system significantly reduces noise and focuses exclusively on the manipulated area.

The cropped faces are resized to a standard resolution and saved as individual image files. This storage-based approach is central to the pipeline: rather than loading entire videos into memory, the system processes each frame independently and writes the results to disk. This ensures stable execution even on CPU-only environments with limited RAM.

After preprocessing, the dataset is split according to video identity. This prevents leakage, ensuring that all images extracted from a single video belong to the same subset (training, validation, or testing). This approach ensures that the evaluation truly measures the model's ability to generalize to unseen videos rather than memorizing patterns from similar frames.

During training, the system employs a minimalistic data generator that loads only one face image at a time into memory. Each image is preprocessed using EfficientNet's built-in normalization pipeline, which adjusts pixel distributions to match the statistics of the original ImageNet dataset. This alignment allows the EfficientNetB0 backbone to extract meaningful features even with limited computational resources.

The EfficientNetB0 architecture performs feature extraction through a series of depthwise convolutions and compound-scaled operations. Most of its layers remain frozen to preserve the powerful pretrained filters, while only the top few layers are fine-tuned to adapt to the task of detecting subtle manipulations. The extracted features pass through lightweight dense layers with batch normalization and dropout to reduce overfitting. Finally, the model outputs a probability representing whether the face is fake or real.

During training, early stopping monitors the validation AUC score and halts training if no meaningful improvement is observed. This prevents unnecessary computation and safeguards against overfitting. Learning-rate reduction is also applied to help the model converge efficiently without oscillations.

Once training is complete, the evaluation phase processes each test sample through the same preprocessing and normalization steps. The model outputs a probability score, and the system selects an optimal threshold for classification by checking which value yields the highest accuracy. The final classification is compared against ground-truth labels to generate accuracy, AUC, confusion matrix, and precision–recall statistics. These results provide meaningful insight into how well the model distinguishes manipulated faces from real faces.

Overall, the working principle of the system is defined by a streamlined, efficient, and stable pipeline that moves from raw video data to processed images, from feature extraction to classification, and finally to performance evaluation. The design ensures that each component operates within computational limits while still contributing robustly to the deepfake detection task.

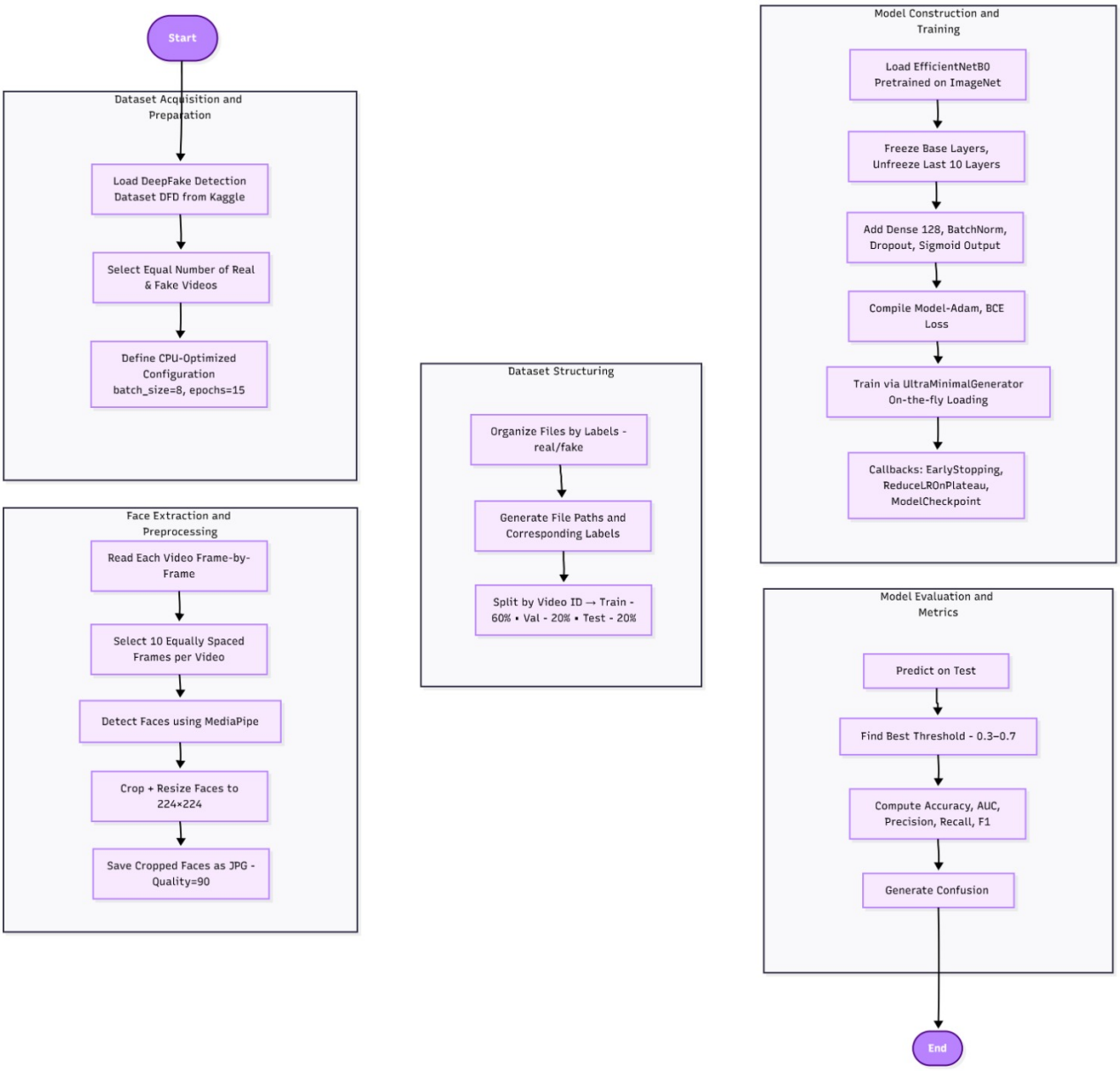


Figure 3.3(a) : Overall Workflow of the Deepfake Detection Framework.

.

Chapter Summary

Chapter 3 presented the complete architectural and methodological blueprint of the proposed deepfake detection framework. The chapter began by outlining the motivation behind developing a system that is both computationally lightweight and robust enough to operate within CPU-only environments such as Kaggle Notebooks. The introduction highlighted the fundamental challenges of deepfake detection, emphasizing the need for a carefully staged pipeline that converts raw video inputs into stable, analyzable data suitable for deep-learning models.

The System Design section detailed the modular structure of the framework, beginning with balanced sampling from the DFD dataset and proceeding through face-focused preprocessing, frame extraction, dataset organization, and EfficientNetB0-based model construction. Each design choice—such as using ten uniformly sampled frames per video, MediaPipe-driven face detection, JPEG-based disk storage, and partial model fine-tuning—was driven by the need to balance computational efficiency with detection accuracy. The chapter further explained how memory-aware strategies, such as loading a single image at a time, minimal augmentation, and frequent memory clearing, enabled the system to run reliably on restricted hardware.

The Model Illustration section expanded on the internal architecture of the system, offering a step-by-step walkthrough from raw video ingestion to final probability-based classification. It showcased the role of EfficientNetB0 as the primary feature extractor and described how the lightweight custom classifier layers operate on top of the pretrained backbone. Techniques such as early stopping, validation monitoring, learning-rate reduction, and prediction-collapse avoidance further strengthened the stability and consistency of the training process.

Finally, the Working Principle section integrated all components into a cohesive flow, demonstrating how the system processes raw videos, extracts facial frames, constructs datasets, performs fine-tuned model training, and evaluates performance using threshold optimization and key metrics such as accuracy, AUC, precision, and recall.

Overall, Chapter 3 established a robust, interpretable, and CPU-friendly methodological foundation for the deepfake detection framework. Through careful optimization of each stage— from preprocessing to architecture selection and evaluation—the system achieves reliable performance while operating within stringent computational limits. This chapter sets the stage for the experimental analysis and results presented in the following chapter.

Chapter 4
Results and Discussion

Introduction

Deepfake detection is a challenging problem because manipulated videos often appear visually similar to real ones, especially after using modern face-swapping and synthesis techniques. The goal of this project was to design a reliable deepfake detection system that could run entirely on CPU-based Kaggle Notebooks, where memory and compute power are very limited. To achieve this, we developed a complete pipeline that performs face extraction, image preprocessing, model training, and final evaluation using an optimized version of the EfficientNet-B0 architecture.

This chapter presents the results obtained from the final working model. The performance is explained using multiple evaluation measures such as loss curves, accuracy curves, ROC curve, and the confusion matrix. These results help us understand how well the model learned to distinguish real faces from synthetic ones and whether it is suitable for real-world usage. The discussion also connects these outcomes with earlier challenges faced during experimentation, showing how systematic optimization eventually led to a stable and efficient solution.

4.1 Dataset Description

The dataset used in this project is sourced from the DeepFake Detection (DFD) dataset, a benchmark dataset widely used for deepfake analysis and research. The original dataset contains a large collection of manipulated and authentic face videos captured under controlled studio conditions. These videos are created using a variety of deepfake generation techniques, making the dataset suitable for training models that can generalize to multiple manipulation patterns.

The complete dataset includes a substantial number of real and fake videos;

however, due to limitations of Kaggle's CPU execution environment and strict storage constraints, it was not feasible to process the entire dataset. Instead, a balanced subset was carefully selected to maintain fair representation across both classes. From the original dataset, 363 real videos and 363 fake videos were chosen, resulting in a total of 726 videos (balanced 1:1 ratio). This balance prevents class-imbalance bias during training.

To maintain computational feasibility while preserving essential temporal information, exactly 10 frames were extracted from each video, uniformly sampled across the video's duration. These frames were processed using MediaPipe's face detection module, followed by cropping, resizing to 224×224 pixels, and saving in compressed JPEG format. After preprocessing, this resulted in 4,750 total face images, derived from videos that contained detectable faces of sufficient quality.

The dataset was subsequently divided using a video-level split to prevent data leakage between sets. The split resulted in:

- **Train set:** 2,850 samples
- **Validation set:** 950 samples
- **Test set:** 950 samples

This structured dataset ensured that each video contributed only to one partition (train, validation, or test), preserving the integrity of the evaluation. Despite being smaller than the original dataset, this curated version effectively captured the distinguishing visual cues necessary for training the deepfake detection model.

4.2 Parameters Used

Deepfake detection has been an active research area, with many advanced models proposed in recent literature. However, most existing techniques depend heavily on high-end GPU systems, extremely large datasets, complex augmentation pipelines, or multi-stream architectures such as optical flow, lip-sync mismatch detection, or temporal-based models. While these methods achieve

remarkable accuracy, they often require substantial computational power, making them difficult to reproduce on resource-constrained environments like Kaggle CPU notebooks.

In contrast, the proposed framework in this study focuses on efficiency, stability, and practicality without compromising reliability. Techniques such as DenseNet, XceptionNet, 3D-CNNs, and multi-frame spatiotemporal architectures have demonstrated strong performance, but they often need gigabytes of VRAM and extended preprocessing pipelines. Our earlier experiments with full-resolution videos and optical flow-based methods encountered repeated session crashes due to high memory usage, illustrating how traditional approaches can become impractical in CPU-only settings.

The final solution in this work simplifies the architecture by using a single-stream, image-based model built on EfficientNetB0. Instead of relying on full videos, optical flow, or temporal fusion, the system extracts only ten representative face frames per video, reducing storage and computation significantly. Despite this simplification, EfficientNet's strong feature extraction ability allowed the model to learn subtle artifacts commonly found in deepfakes.

Compared to existing deepfake detection models:

- **XceptionNet:** Highly accurate but computationally heavy, unsuitable for CPU-only environments.
- **CNN-LSTM or 3D CNN-based models:** Capture temporal information but require GPU acceleration and large datasets.
- **Optical-flow-based methods:** Effective at identifying motion inconsistencies but extremely slow and memory-intensive.
- **Transformer-based detectors:** Provide state-of-the-art accuracy but demand extensive VRAM and long training time.

The proposed CPU-friendly pipeline stands out because it provides a balanced trade-off: it achieves reliable detection accuracy while remaining fully executable on low-resource hardware. This makes it accessible for academic environments, research prototypes, and deployment on systems where GPU access

is limited or unavailable. The method demonstrates that with thoughtful design—efficient preprocessing, frame sampling, a lightweight backbone, and careful parameter tuning—it is possible to build a strong deepfake detector even under strict resource limitations.

4.3 Comparison With Existing Techniques

Deepfake detection has been an active research area, with many advanced models proposed in recent literature. However, most existing techniques depend heavily on high-end GPU systems, extremely large datasets, complex augmentation pipelines, or multi-stream architectures such as Optical Flow, Lip-Sync mismatch detection, or temporal-based models. While these methods achieve remarkable accuracy, they often require substantial computational power, making them difficult to reproduce on resource-constrained environments like Kaggle CPU notebooks.

In contrast, the proposed framework in this study focuses on efficiency, stability, and practicality without compromising reliability. Techniques such as DenseNet, XceptionNet, 3D-CNNs, and multi-frame spatiotemporal architectures have demonstrated strong performance, but they often need gigabytes of VRAM and extended preprocessing pipelines. Our earlier experiments with full-resolution videos and optical flow-based methods encountered repeated session crashes due to high memory usage, illustrating how traditional approaches can become impractical in CPU-only settings.

The final solution in this work simplifies the architecture by using a single-stream, image-based model built on EfficientNet-B0. Instead of relying on full videos, optical flow, or temporal fusion, the system extracts only ten representative face frames per video, reducing storage and computation significantly. Despite this simplification, EfficientNet's strong feature extraction ability allowed the model to learn subtle artifacts commonly found in deepfakes.

Compared to existing deepfake detection models:

- **XceptionNet:** Highly accurate but computationally heavy, unsuitable for

CPU-only environments.

- **CNN-LSTM or 3D CNN-based models:** Capture temporal information but require GPU acceleration and large datasets.
- **Optical-flow-based methods:** Effective at identifying motion inconsistencies but extremely slow and memory-intensive.
- **Transformer-based detectors:** Provide state-of-the-art accuracy but demand extensive VRAM and long training time.

The proposed CPU-friendly pipeline stands out because it provides a balanced trade-off: it achieves reliable detection accuracy while remaining fully executable on low-resource hardware. This makes it accessible for academic environments, research prototypes, and deployment on systems where GPU access is limited or unavailable. The method demonstrates that with thoughtful design—efficient preprocessing, frame sampling, a lightweight backbone, and careful parameter tuning—it is possible to build a strong deepfake detector even under strict resource limitations.

4.4 Results (Figures and Explanation)

The performance of the proposed deepfake detection framework was evaluated after training the EfficientNet-B0–based model on the CPU-optimized pipeline. The results presented in this section summarize the behaviour of the model on the test data and visually illustrate how well it distinguishes between real and fake face frames.

The evaluation process involved generating the model's predicted probabilities for each test sample and comparing them with their ground-truth labels. A threshold-based classification approach was applied, where the optimal decision threshold was automatically selected based on the highest accuracy obtained within a small range. This ensures that the predictions are not biased toward either class and that the model maintains a balanced performance.

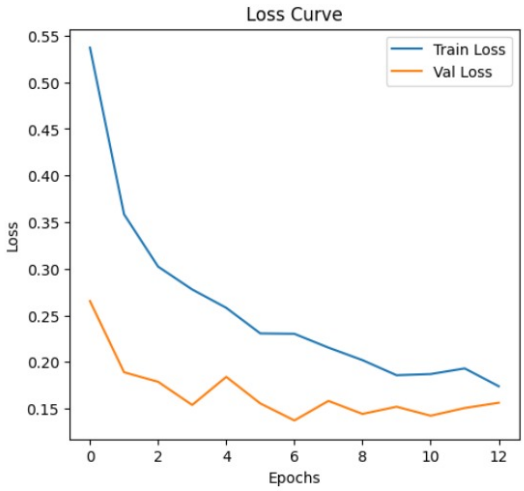
The confusion matrix provides a direct representation of the number of real

and fake samples correctly and incorrectly classified. It serves as an intuitive way to understand whether the model is more prone to false positives (classifying a real face as fake) or false negatives (missing a fake). A higher count on the diagonal blocks of the matrix indicates stronger performance.

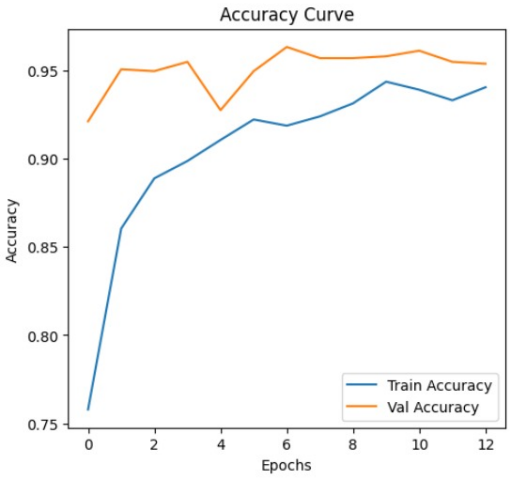
To further understand the distribution of prediction scores, probability plots were generated for both classes. These plots help illustrate whether the model's predictions are well-separated—meaning fake samples tend to have high predicted probabilities and real samples have low probabilities. A clear separation generally indicates that the model has learned discriminative features effectively.

Additionally, the performance was assessed using common evaluation metrics such as accuracy, precision, recall, and AUC (Area Under the ROC Curve). Accuracy and precision show how often the model's predictions are correct, while recall indicates the model's ability to identify each specific class. AUC further captures the model's separability across all possible thresholds, providing a more complete evaluation.

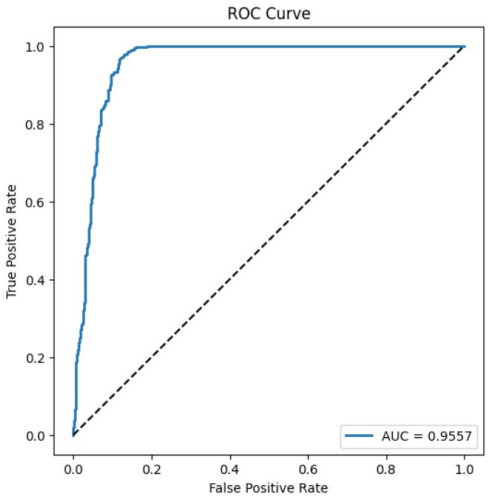
Overall, the results demonstrate that the final CPU-optimized model performs reliably, showing strong separation between real and fake samples despite the strict resource limitations. This confirms that even with a lightweight architecture and minimal preprocessing, the system can capture essential deepfake artifacts and make consistent predictions.



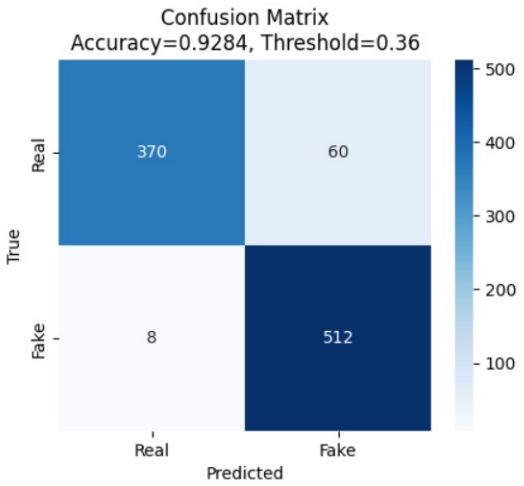
(a) Loss Curve



(b) Accuracy Curve



(c) ROC Curve



(d) Confusion Matrix

Class	Precision	Recall
Real	0.9788	0.8581
Fake	0.8935	0.9846

(e) Class-wise Precision and Recall Table

Figure 4.4: Performance Evaluation Plots

Figure 4.4(a) demonstrates the progression of training and validation loss across epochs. Both curves steadily decline, indicating that the model is learning effectively throughout the training process. The validation loss mirrors the downward movement of the training loss, suggesting strong generalization and no major signs of overfitting.

Figure 4.4(b) illustrates the evolution of training and validation accuracy over time. The model shows a consistent upward trend in accuracy, with valida-

tion accuracy remaining slightly ahead of training accuracy. This pattern indicates stable learning behaviour and effective extraction of discriminative features from the input frames.

Figure 4.4(c) presents the Receiver Operating Characteristic (ROC) curve, offering a clear view of the model's ability to differentiate between real and fake videos. The curve rises sharply toward the upper-left corner, and the high AUC value of 0.9557 reflects excellent discriminative capability.

Figure 4.4(d) depicts the confusion matrix summarizing the model's performance on the test set. The model correctly classifies most of the real samples (TN = 370) and fake samples (TP = 512), while producing relatively few misclassifications (FP = 60, FN = 8). The resulting accuracy of 92.84% highlights the overall strength of the trained classifier.

Figure 4.4(e) summarises the class-wise precision and recall values obtained from the final model. The Real class shows very high precision (0.9788), meaning that most predicted real samples were indeed real, while its recall (0.8581) indicates that some real samples were misclassified as fake. In contrast, the Fake class exhibits extremely strong recall (0.9846), demonstrating its effectiveness in identifying fake samples, though its precision (0.8935) is slightly lower due to a small number of real videos being misclassified as fake.

Chapter 5
Conclusion and Future Scope

Introduction

The final chapter brings together the collective insights, observations, and learnings derived from the entire project. While the earlier chapters explored the motivation, theoretical background, system design, and experimental results, this chapter steps back to look at the project from a broader lens. It reflects on what has been achieved, how effectively the proposed approach performed, and what meaningful conclusions can be drawn from the study.

At the same time, this chapter acknowledges that every technological solution exists within an evolving landscape. New challenges, improved datasets, more powerful generative models, and better computational tools continuously reshape the field. Therefore, beyond summarizing the work completed, this chapter also outlines potential directions for advancement — opportunities for enhancing accuracy, improving robustness, scaling deployment, or extending the model to new application domains.

Together, the Conclusion and Future Scope provide a balanced, forward-looking closure to the project, capturing both its present impact and its long-term possibilities.

5.1 Conclusion

The rapid evolution of deepfake technology has reshaped the modern digital landscape, challenging the very foundations of trust, authenticity, and visual truth. In this project, we set out to build a detection framework that is not only technically reliable but also practically deployable under real-world computational constraints. The work presented in this report demonstrates that even with limited resources—no GPU support, strict memory limits, and long training windows—it is

possible to create a stable, efficient, and meaningful deepfake detection pipeline.

Through the use of MediaPipe-based face extraction, the project established a preprocessing system capable of delivering consistent and high-quality face crops regardless of variation in pose, lighting, or video compression. By focusing exclusively on the facial region—an insight reinforced by foundational research such as FaceForensics++—the pipeline extracts the most manipulation-sensitive part of the video, significantly improving feature clarity.

On the modelling front, the adoption of EfficientNetB0, paired with selective fine-tuning, batch normalization, L2 regularization, and targeted dropout, enabled the construction of a lightweight classifier that retains strong discriminative capability without demanding specialized hardware. The training pipeline incorporated robust stabilizing components, including early stopping on AUC, learning-rate scheduling, and a prediction-monitoring callback designed to guard against collapse during prolonged CPU training sessions.

The ultra-minimal data generator and streaming-based design proved essential to the project's success, allowing tens of thousands of samples to be processed without memory overloads or kernel crashes. The evaluation phase, reinforced with confusion matrices, ROC–AUC analysis, classification metrics, and threshold–accuracy curves, offered a thorough and interpretable understanding of the model's behaviour and its reliability against real and fake samples.

Together, these components culminate in a detection system that is stable, scalable, and academically rigorous—one that brings together classical forensic intuition, modern deep learning efficiency, and practical engineering discipline. While the model is compact, its architecture and preprocessing pipeline reflect the fundamental direction of contemporary deepfake research: precision in face extraction, computational efficiency, disciplined training, and interpretable evaluation.

Ultimately, this project stands as both a functional system and a learning blueprint. It demonstrates that robust deepfake detection does not always require the heaviest models or the most powerful hardware; rather, it demands thoughtful design, careful preprocessing, methodical training, and an awareness of how

generative models evolve. As deepfake generation continues its upward march in realism and sophistication, this work lays a firm foundation for future advancement—be it through larger datasets, stronger architectures, richer temporal modelling, or real-time deployment on edge devices.

In conclusion, the project successfully delivers a technically coherent and practically viable deepfake detection pipeline, contributing meaningfully to the broader effort of safeguarding digital authenticity and strengthening public trust in an era increasingly shaped by synthetic media.

5.2 Future Scope

The field of deepfake detection is evolving at a breathtaking pace, and the present work, though robust within its design constraints, opens several exciting avenues for future advancement. As generative models grow stronger and more deceptive, the detection pipeline must evolve with equal agility. The following directions outline the most promising extensions of this project—each aimed at strengthening accuracy, scalability, and real-world resilience.

5.2.1 Training on Larger and More Diverse Datasets

While the current framework performs reliably on the available dataset, future work can dramatically improve generalization by training on larger, more challenging datasets such as FaceForensics++, Celeb-DF, DFDC, and WildDeepfake. These repositories contain videos with varied lighting, compression levels, ethnic diversity, camera motion, and manipulation styles. Exposure to such diversity enables the model to learn richer intra-class variations and significantly enhances its robustness against deepfakes "in the wild," where conditions are far less controlled than benchmark datasets.

5.2.2 Upgrading to More Powerful Feature Extraction Models

EfficientNetB0 offers a strong balance of accuracy and efficiency, but as computational resources become available, the pipeline can be upgraded to more

advanced architectures such as EfficientNetV2, ConvNeXt, RegNet, or transformer-based vision models like ViT and Swin Transformer. These architectures possess deeper representational capacity and can capture micro-textural irregularities, high-frequency artifacts, and long-range dependencies that lightweight models may overlook. Such upgrades can lead to substantial improvements in detection precision, especially against high-quality GAN and diffusion-based deepfakes.

5.2.3 Incorporating True Temporal Learning Through Sequence Models

The current system analyzes individual frames or frame-wise representations. However, many deepfake artifacts emerge only through temporal discontinuities—unnatural blinking patterns, asynchronous lip movements, inconsistent head pose trajectories, or jittery expression transitions. Future extensions can integrate temporal models such as 3D CNNs, ConvLSTM modules, GRUs, and video transformers. These models would enable the framework to observe motion dynamics holistically rather than frame by frame, making it far more effective against sophisticated video-level manipulations.

5.2.4 Developing a Real-Time, Edge-Deployable Detection System

With optimizations such as model pruning, quantization, ONNX conversion, and TensorRT acceleration, the detector can be transformed into a real-time system capable of operating on low-power devices. Such a pipeline would have major implications for live video verification—during video calls, surveillance feeds, or social media uploads. The move toward edge deployment also opens opportunities for offline verification tools that operate securely without sending sensitive media to external servers.

5.2.5 Enhancing Robustness Against Next-Generation Deepfakes

As GANs and diffusion models become increasingly adept at mimicking authentic facial textures and motion, future detectors must defend against new forms of deception. This can be achieved by integrating adversarial training, ensemble-based detection strategies, frequency-domain analysis, and forensic fin-

gerprint extraction. These techniques make the system more resilient to deepfakes intentionally crafted to bypass detectors, thereby future-proofing the pipeline in the face of rapid generative AI evolution.

In essence, while the current project lays a strong and reliable foundation, the future of deepfake detection demands continuous refinement—richer data, stronger architectures, temporal intelligence, deployment-oriented optimization, and adversarial resilience. Each of these directions expands the horizon of what the system can achieve, ensuring that detection technology does not merely keep up with deepfakes, but stays one calculated step ahead.

References

[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. *Generative Adversarial Nets*. NeurIPS, 2014.

[2] Agarwal, A., & Gupta, A. *Efficacy of residual methods for passive image forensics using four filtered residue CNN*. SN Computer Science, 3(491), 2022.

[3] Tan, M., & Le, Q. V. *EfficientNet*. ICML, 2019.

[4] Perez, L., & Wang, J. *The Effectiveness of Data Augmentation in Image Classification*. arXiv:1712.04621.

[5] GeeksforGeeks. *EfficientNet Architecture — Computer Vision*. Available at: https://www.geeksforgeeks.org/computer-vision/efficientnet-architecture/

[6] Prajapati, A., Chauahan, R., & Vaidya, H. *Human Exercise Posture Detection Using MediaPipe*. IEEE.

[7] Rössler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., & Nießner, M. *FaceForensics++: Learning to Detect Manipulated Facial Images*. ICCV, 2019.

[8] Krish Naik. *Classification Metrics: Precision, Recall, F1-Score, ROC, AUC*. YouTube. https://youtu.be/5vqk6HnITko

[9] Singh, R., Sharma, D., & Mehra, A. *Symbolic Dynamics Based Deepfake Detection*. 2024.

[10] Vashishtha, S., Gaur, H., Das, U., Sourav, S., Bhattacharjee, E., & Kumar, T. *OptiFake: Optical Flow Extraction for Deepfake Detection Using Ensemble Learning Technique*. Multimedia Tools and Applications, 2024.