Advanced Deepfake Detection Using Facial Biometrics: A Comparative Study of Face Detection and Deep Learning Models

Jaaie Kadam – 23222441

Department of Computer Science and Information Systems
Faculty of Science and Engineering
University of Limerick

Submitted to the University of Limerick, August 2024 in partial fulfilment of the requirements for the Degree of Master of Science in Artificial Intelligence & Machine Learning

Supervisor: Dr. Tony Scanlan

Abstract

The rapid advancement of deep learning and artificial intelligence has enabled the creation of highly realistic fake videos, commonly known as deepfakes. These manipulated videos pose significant threats to privacy, security, and information integrity, necessitating the development of robust detection mechanisms. This thesis focuses on detecting deepfake videos using facial biometrics, leveraging the FaceForensics++ dataset, a benchmark for studying facial manipulation.

The research integrates two prominent face detection techniques: Multi-Task Cascaded Convolutional Networks (MTCNN) and Faster Region-Based Convolutional Neural Networks (Faster R-CNN). MTCNN is noted for precise facial landmark localization, while Faster R-CNN excels in generating accurate region proposals. These techniques are followed by the application of three state-of-the-art deep learning models—ResNet, EfficientNet, and Xception—to classify detected faces as real or fake.

Experimental results demonstrate that combining advanced face detection techniques with deep learning models significantly improves the system's ability to distinguish between authentic and manipulated facial images. A comparative analysis of MTCNN and Faster R-CNN, in conjunction with ResNet, EfficientNet, and Xception, provides insights into the trade-offs between detection accuracy and computational efficiency.

The system undergoes rigorous evaluation on unseen video data to validate its effectiveness, focusing on predictive accuracy and confidence levels. Confusion matrices and classification reports offer detailed insights into model performance, highlighting strengths and areas for improvement.

The study concludes that facial biometrics, coupled with state-of-the-art machine learning techniques, presents a powerful tool for combating the growing threat of deepfakes, with potential applications in digital forensics, media verification, and security.

DECLARATION

I would like to extend my deepest gratitude to my supervisor, Dr. Tony Scanlan, for his unwavering support and invaluable guidance throughout this project. His consistent advice during our regular meetings was instrumental in navigating the challenges I faced, and his insightful suggestions greatly contributed to the successful completion of this work.

I am especially grateful for his encouragement and mentorship, which have fostered my passion for Computer Vision and Artificial Intelligence, areas I am now more enthusiastic about than ever.

Jaaie Kadam

Limerick, 2024

ACKNOWLEDGMENTS

First and foremost, I would like to express my heartfelt gratitude to Dr. Tony Scanlan for his timely guidance and support throughout the duration of this dissertation. I'd also like to thank him for his valuable suggestions, feedback, and generous advice, which empowered me to push my limits and successfully complete the work. It was an honor to work under his supervision.

I would like to thank Dr. J.J. Collins for teaching us the various Deep Learning concepts. I am also grateful to Dr. Tabea De Wille for her research methodology course, which helped me with my dissertation. Finally, I'd like to thank Dr. Emil Vassev, the course director, and all of the lecturers who helped me understand Artificial Intelligence.

A special thanks to my family and friends for their unwavering support and encouragement. Above all, I am grateful to the Almighty God for his blessings throughout this journey.

Contents

1. Introduction/Background

1.1. Overview

The advent of deep learning and artificial intelligence has revolutionized various domains, leading to unprecedented advancements in fields such as computer vision, natural language processing, and robotics. However, these advancements have also given rise to new challenges, particularly in the area of media manipulation. One of the most significant and concerning developments in this regard is the creation of deepfakes—highly realistic fake videos generated using deep learning techniques. Deepfakes can superimpose one person's face onto another's body, modify facial expressions, and even alter speech, making it appear as though someone said or did something they did not.

The implications of deepfakes are far-reaching and severe. They threaten personal privacy, as individuals' likenesses can be manipulated without consent. They endanger public security, with the potential for deepfakes to be used in disinformation campaigns, fraud, and blackmail. Furthermore, deepfakes undermine the integrity of information, as distinguishing between real and manipulated content becomes increasingly challenging. In this context, developing reliable and effective deepfake detection mechanisms is of paramount importance.

1.2. Problem Statement

Despite the growing prevalence of deepfakes, detecting them remains a significant challenge. Traditional detection methods often fall short due to the high quality and realism of modern deepfakes. Current approaches to deepfake detection typically rely on identifying inconsistencies in visual artifacts, such as unnatural blinking or irregular lighting. However, as deepfake generation techniques improve, these inconsistencies become less noticeable, rendering traditional detection methods less effective.

The core problem this research addresses is the need for more sophisticated detection techniques that can accurately distinguish between real and manipulated facial images, even

when deepfakes are of high quality. This research proposes using facial biometrics—specifically facial detection and deep learning models—to enhance the accuracy and reliability of deepfake detection. By leveraging facial biometrics, this research aims to overcome the limitations of existing detection methods and provide a more robust solution to the deepfake problem.

1.3. Research Aims and Questions

The primary aim of this research is to develop and evaluate a robust deepfake detection framework that integrates facial biometrics with state-of-the-art deep learning models. Specifically, the study seeks to investigate the effectiveness of combining two advanced facial detection techniques—Multi-Task Cascaded Convolutional Networks (MTCNN) and Faster Region-Based Convolutional Neural Networks (Faster R-CNN)—for accurate face detection in video sequences. By applying these detection techniques, the research aims to enhance the reliability of face extraction, which is a crucial step in identifying manipulated content. Following face detection, the study applies three deep learning models—ResNet, EfficientNet, and Xception—to classify the detected faces as real or fake. Through a comprehensive evaluation using the FaceForensics++ dataset, the research compares the performance of these models, with a focus on the trade-offs between detection accuracy and computational efficiency. Ultimately, this study aims to provide insights into how the integration of facial biometrics and deep learning models can improve the accuracy and reliability of deepfake detection, while also identifying the strengths and limitations of the proposed framework. The findings of this research are expected to contribute to the development of more effective tools for combating the growing threat of deepfakes, with implications for digital forensics, media verification, and security.

The research is guided by the following key questions:

1. How effective are different face detection methods in accurately localizing faces in both real and deepfake videos, particularly when faced with challenges such as varying facial expressions, angles, and lighting conditions?
2. How does the combination of specific face detection methods with deep learning models influence the overall accuracy and reliability of deepfake detection systems?
3. What impact does integrating the FaceForensics++ dataset have on the performance of deepfake detection models, and how does this performance compare when using alternative datasets?

4. What are the strengths and limitations of the proposed deepfake detection framework, and how can it be improved for practical applications?
5. How can the performance of deepfake detection models be effectively evaluated, and what metrics and methodologies are best suited for assessing their accuracy, robustness, and overall effectiveness?

1.3. Thesis Outline

This thesis is structured as follows:

- **Chapter 2: Literature Review**
    - This chapter provides an overview of existing research in the field of deepfake detection, facial biometrics, and related technologies. It examines current methodologies, datasets, and the challenges associated with detecting deepfakes.
- **Chapter 3: Methodology**
    - Details the experimental setup, including face detection techniques (MTCNN, Faster R-CNN) and deep learning models (ResNet, EfficientNet, XceptionNet), along with data preprocessing, training, and evaluation methods.
- **Chapter 4: Results**
    - Presents the experimental results, analyzing detection performance across models and face detection methods, with a focus on trends and comparative analyses.
- **Chapter 5: Conclusion and Future Work**
    - Summarizes the research findings, discusses their significance, and suggests potential future improvements and research directions in deepfake detection.

2. Literature Review

2.1. Deepfake Detection: An Overview

Deepfakes are an advanced form of digital media manipulation, where artificial intelligence (AI), particularly deep learning, is used to create highly realistic but fake content. The term "deepfake" is derived from "deep learning" and "fake," highlighting the use of sophisticated machine learning techniques to produce convincing falsified media. These manipulated videos

often involve seamlessly swapping one person's face with another's in existing footage, making it appear as though the individual is saying or doing something they never did (Ian Goodfellow, 2014)

Origins and Development: The concept of deepfakes emerged in 2017, initially gaining traction on online platforms such as Reddit, where users began sharing videos that convincingly swapped the faces of celebrities. This technology, originally seen as a novel tool, soon garnered widespread attention due to its potential for misuse in creating deceptive media that could easily be mistaken for real footage. (Korshunov & Marcel, 2018)

Public Awareness and Ethical Concerns: The rise of deepfakes has sparked significant ethical and security concerns, primarily because of their potential to spread misinformation, create non-consensual pornography, and defame individuals. The ease with which deepfakes can be created and disseminated online poses serious threats to privacy and the integrity of digital information. As a result, there is an urgent need for public education on the dangers of deepfakes, as well as the development of robust detection technologies to mitigate their Impact. (Rössler, et al., 2019)

Deepfakes have a wide range of applications, both beneficial and potentially harmful. In the entertainment industry, deepfakes are employed to create convincing visual effects, such as de-aging actors, resurrecting deceased performers, or enabling an actor to portray multiple characters in the same scene. These applications demonstrate the creative potential of deepfakes in filmmaking and digital media production (Robert Chesney, 2019). For instance, deepfake technology has been used in high-profile films to bring historical figures back to life or to create entirely new performances by blending the likenesses of various actors.

However, the same technology also poses significant risks, particularly in the realm of misinformation and propaganda. Deepfakes have been increasingly used to fabricate videos that spread false information or defame individuals by making them appear to say or do things they never did. This misuse of deepfake technology has raised concerns about the erosion of trust in digital content, as the line between real and fake becomes increasingly blurred (Marie-Helen Maras, 2019). The potential for deepfakes to influence public opinion, manipulate elections, and exacerbate social tensions highlights the urgent need for effective detection and regulation mechanisms.

Educational Importance: Given their potential impact on society, it is crucial to educate people about how deepfakes are made, how they can be detected, and the broader implications of this technology. Understanding the mechanisms behind deepfakes is essential for those working in

digital media, cybersecurity, or information technology, as these fields are at the forefront of combating the misuse of deepfakes (Tero Karras S. L., 2019).

To combat this growing threat, researchers have developed various deepfake detection mechanisms, leveraging state-of-the-art machine learning models and face detection techniques. This literature review explores the existing research on deepfake detection, focusing on facial biometrics, deep learning models, and related technologies. It also examines current methodologies, datasets, and the challenges associated with detecting deepfakes, providing a foundation for the research presented in this thesis.

2.1. Technological Foundations

The creation of deepfakes is grounded in several advanced technologies within AI, particularly deep learning and computer vision. The two primary techniques used to generate deepfakes are **Generative Adversarial Networks (GANs)** and **Autoencoders**. Both of these methods are based on unsupervised learning, where models learn to replicate patterns found in the training data to generate new, highly realistic content.

Generative Adversarial Networks (GANs): Introduced by Ian Goodfellow and his colleagues in 2014, GANs have revolutionized the field of synthetic media generation. A GAN consists of two neural networks: the **generator** and the **discriminator**. These networks are trained simultaneously through a process known as adversarial training (Ian Goodfellow, 2014).

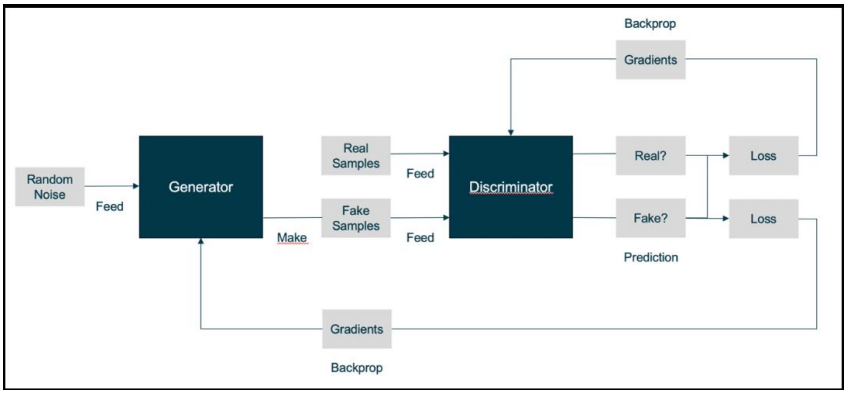


Figure 1: Generative Adversarial Networks Architecture

- **Generator**: The generator network takes random noise as input and produces an image (or frame of a video) as output. The goal of the generator is to create an image that is realistic enough to deceive the discriminator.
- **Discriminator**: The discriminator network, on the other hand, takes an image as input and classifies it as either real (from the original data) or fake (produced by the

generator). The discriminator provides feedback to the generator, which uses this information to improve its output.

This adversarial process forces the generator to create images so realistic that even the discriminator struggles to differentiate them from real ones. In the context of deepfakes, GANs are employed to generate fake video frames that mimic the target person's facial features, expressions, and movements, ultimately creating videos that appear authentic (Ian Goodfellow, 2014) (Tero Karras S. L., 2019)

Application in Deepfakes: GANs are particularly effective in deepfake creation because they can generate high-quality, detailed images that can seamlessly replace the target person's face in a video without noticeable artifacts. The iterative process of adversarial training enables the generation of realistic video frames that are difficult to distinguish from real footage, making GANs the backbone of many deepfake systems (Alec Radford, 2015).

Autoencoders: Another core technology in deepfake creation is the autoencoder, which is particularly effective for face-swapping tasks. An autoencoder is a type of neural network that compresses input data into a latent space and then reconstructs it. In deepfake creation, autoencoders are trained on a large set of images to learn how to represent a person's face in a compressed form (Diederik P Kingma, 2024).

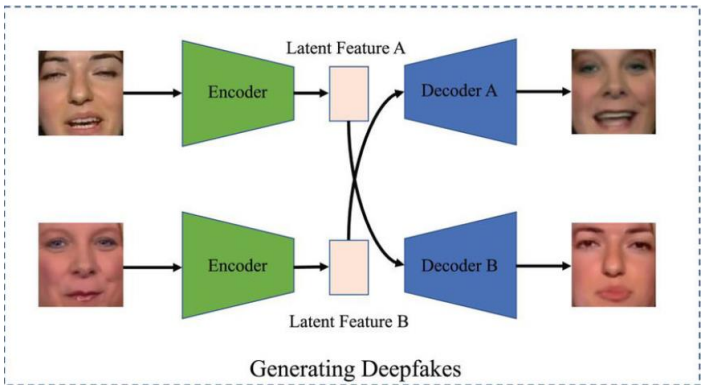


Figure 2: Deepfake Generation Process Using Autoencoders

- **Encoder**: The encoder compresses the input image into a lower-dimensional representation (latent space) that captures essential features of the face.
- **Decoder**: The decoder then reconstructs the image from this latent representation. For deepfakes, two decoders are typically used: one for the source face and one for the target face. By encoding the source face and decoding it with the target's decoder, the system effectively swaps the faces while maintaining realistic expressions and lighting.

Advantages and Challenges: GANs and autoencoders each offer unique advantages for deepfake creation. GANs excel in producing high-quality, detailed images but are notoriously difficult to train due to the delicate balance required between the generator and discriminator. Autoencoders, on the other hand, are easier to train and can be used in real-time applications, although they may not capture as much detail as GANs (Alec Radford, 2015) (Tero Karras S. L., 2019).

Recent Advances: Recent developments have focused on enhancing the capabilities of GANs and autoencoders. For instance, **StyleGAN**, developed by Nvidia, allows for greater control over the generated images, enabling the creation of deepfakes with higher precision and more customizable features. Additionally, the integration of **Variational Autoencoders (VAEs)** has improved the diversity and realism of generated faces by allowing for more complex latent space representations.

Challenges in Training: Training deepfake models is resource-intensive, requiring substantial computational power and large datasets. The process involves fine-tuning the model to balance the generation of realistic content while minimizing detectable artifacts. The success of a deepfake model often depends on the quality of the dataset used for training, as well as the computational resources available for running complex neural networks. (Tero Karras S. L., 2019)

2.2. Categories of Facial Manipulation

Facial manipulation refers to a range of techniques used to alter facial images or videos, often for the purpose of creating realistic but fake representations. These techniques have evolved from simple digital edits to sophisticated manipulations using biometric data and deep learning algorithms. In the context of your thesis on deepfake detection using facial biometrics, it is essential to understand the two primary categories of facial manipulation: **Traditional Facial Manipulation** and **Biometric-Based Facial Manipulation**.

2.2.1. Traditional Facial Manipulation

Traditional facial manipulation involves techniques that rely on basic digital editing tools or simple computer graphics. These methods are often used in entertainment, advertising, and media but can also be applied in malicious contexts, such as creating fake images or videos.

1. **Face Swapping:** Face swapping is one of the most common traditional manipulation techniques, where the face of one person is replaced with another in an image or video.

This method gained popularity through mobile apps and software like Snapchat and Photoshop. The process typically involves detecting facial landmarks—such as the eyes, nose, and mouth—on both faces, aligning them, and then blending the swapped face seamlessly onto the original body (Justus Thies, 2020).

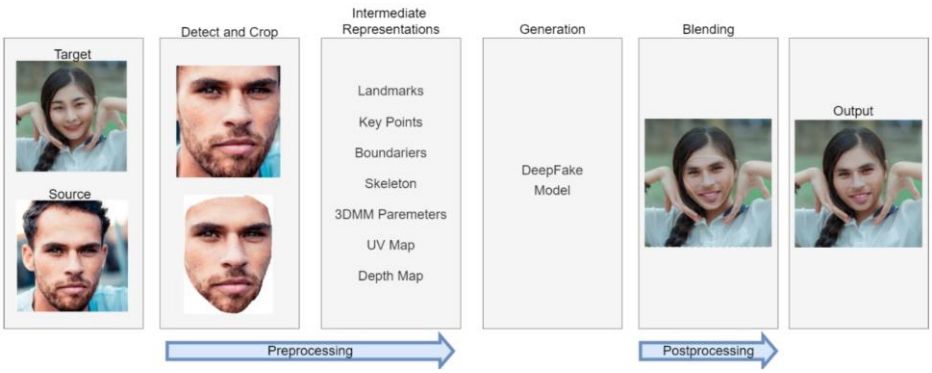


Figure 3: Face-Swapping process

2. **Facial Reenactment:** Facial reenactment allows the transfer of facial expressions from one person (the source) to another (the target) in real-time. This technique is often used in video production and special effects. An example of this is the "Face2Face" system, which captures the facial expressions of a source actor and maps them onto a target actor in a video, preserving the target's identity while altering their expressions (Justus Thies, 2020).

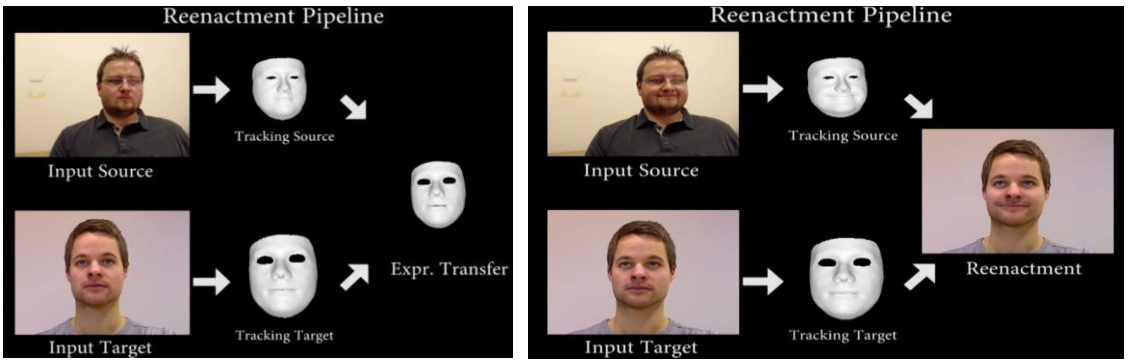


Figure 4: Facial Reenactment Process

2.2.2 Biometric-Based Facial Manipulation

Biometric-based facial manipulation is more advanced and involves the use of facial biometric data for both creating and detecting manipulated media. This category is particularly relevant to your thesis, as it covers the techniques used to create deepfakes and the methods employed to detect them.

1. **Facial Recognition and Biometric Analysis:** Facial recognition systems use biometric data to identify or verify individuals based on their facial features. These systems analyze the geometry of the face, such as the distance between the eyes, the width of the nose, and the shape of the cheekbones. This biometric data can be manipulated to create highly realistic fake images or videos. For example, facial recognition technology can be repurposed to generate synthetic faces or alter facial expressions, contributing to the creation of deepfakes (Schroff, Kalenichenko, & Philbin, 2015).

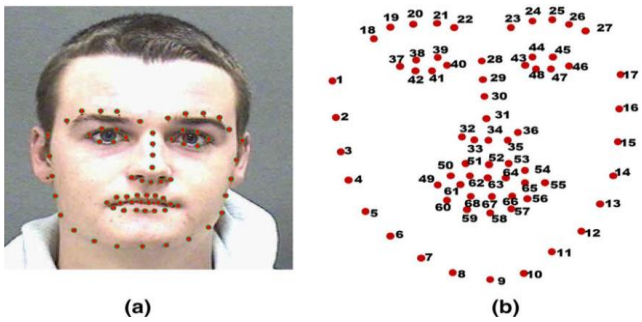


Figure 5: Facial Landmark Detection: (a) Facial landmarks identified on a subject's face, (b) Corresponding numbered landmark points used for facial feature mapping in deepfake detection.

2. **Deepfake Technology:** Deepfakes are one of the most notorious applications of biometric-based facial manipulation. They involve using deep learning algorithms, such as Generative Adversarial Networks (GANs) or autoencoders, to create realistic fake videos where one person's face is superimposed onto another's body. The process involves training a neural network on a large dataset of images of the target person's face, allowing the model to generate convincing fake images or videos that appear real.

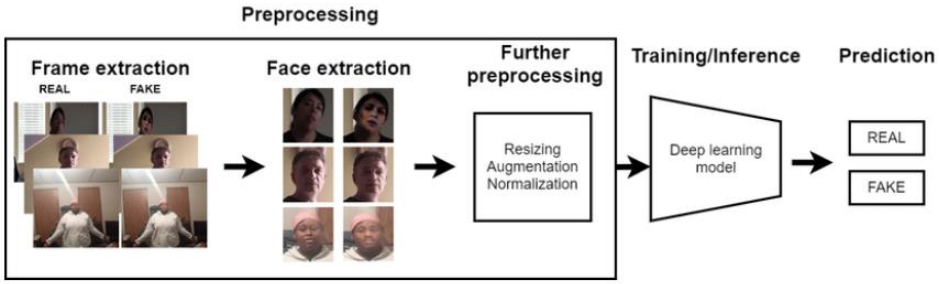


Figure 6: Deepfake Detection Pipeline

3. **Facial Feature Analysis:** Facial feature analysis involves examining specific facial characteristics, such as texture, color, and micro-expressions, to detect manipulations. This technique is particularly effective in identifying deepfakes, as it can detect subtle inconsistencies that are not easily noticeable to the human eye. Methods like Multi-task Cascaded Convolutional Networks (MTCNN) and Faster R-CNN are commonly used

to detect and analyze these features, helping in the accurate identification of manipulated media (Zhang, Zhang, Li, & Qiao, 2016)

In our thesis, Facial (Feature Analysis) Biometrics are used to enhance deepfake detection by integrating facial landmark detection and deep learning models. These models—ResNet50, EfficientNet-B0, and XceptionNet—are trained to recognize subtle patterns and anomalies in facial data, making them effective in distinguishing real from fake images.

2.3 Face Detection

Face detection is a computer vision technology that identifies and locates human faces within digital images or video frames. Unlike face recognition, which attempts to match a detected face to a database of known faces, face detection simply detects the presence and position of faces. It is a foundational step in many applications, including facial recognition systems, emotion detection, augmented reality, and, importantly for your thesis, deepfake detection.

In the context of deepfake detection, face detection plays a critical role in preprocessing the data for further analysis. Before any analysis or classification can be done to determine whether an image or video is real or fake, the system needs to accurately identify and isolate the facial regions. These detected faces are then analyzed using various deep learning models to check for inconsistencies or artifacts indicative of manipulation. Effective face detection ensures that the subsequent deepfake detection process is accurate and reliable.

In our thesis, MTCNN and Faster R-CNN detection techniques are used.

2.3.1. MTCNN (Multi-task Cascaded Convolutional Networks)

MTCNN is a robust and widely used face detection method that excels at both detecting faces and aligning facial landmarks. It was specifically designed to handle face detection in unconstrained environments, such as varied lighting conditions, poses, and occlusions. MTCNN is often favored in tasks that require high accuracy in detecting facial features, such as deepfake detection, where precise facial alignment is crucial.

MTCNN is particularly effective for deepfake detection because of its ability to accurately detect facial landmarks, even in challenging conditions. This accuracy in landmark detection is vital when analyzing facial regions for signs of manipulation. The method's robustness ensures that the detected faces are correctly aligned, which is essential for the subsequent deep learning models to analyze the facial features consistently.

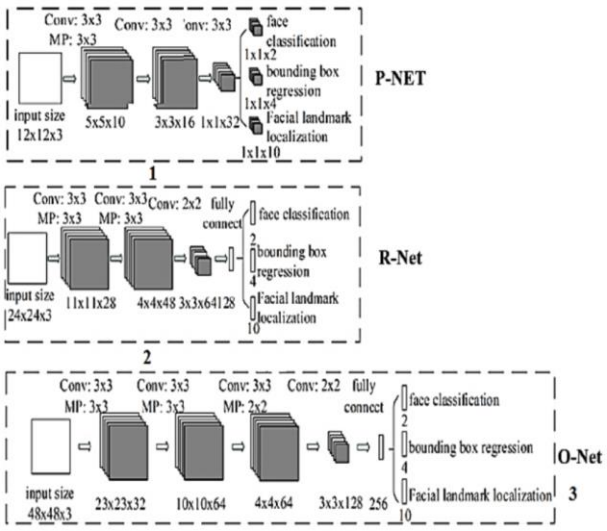


Figure 7: Architecture of the MTCNN (Multi-task Cascaded Convolutional Networks)

Architecture of MTCNN consists of three stages, each progressively refining the detection and alignment of faces:

1. **P-Net (Proposal Network):** The P-Net scans the image and generates candidate windows that potentially contain faces. It performs a rough detection by sliding a small window over the image, producing bounding box proposals with corresponding scores.
2. **R-Net (Refine Network):** The R-Net takes the proposals from the P-Net and refines them by rejecting a large number of false positives. It further adjusts the bounding box coordinates for a more accurate detection.
3. **O-Net (Output Network):** The O-Net performs the final refinement, producing high-confidence face detections and precise facial landmark locations (e.g., eyes, nose, and mouth corners). The O-Net is also responsible for further refining the bounding boxes.

Each stage involves convolutional layers that extract features from the input images, followed by fully connected layers that make predictions. The cascading structure allows MTCNN to be both efficient and accurate, making it well-suited for real-time applications like deepfake detection.

*2.3.*2. Faster R-CNN (Region-based Convolutional Neural Networks)

Faster R-CNN is an advanced object detection framework that significantly improves the speed and accuracy of detecting objects in images. In the context of face detection, Faster R-CNN is used to locate and classify faces within an image, producing bounding boxes that enclose detected faces. Unlike traditional R-CNN methods, Faster R-CNN integrates a Region Proposal

Network (RPN) that allows the system to propose regions likely to contain objects (or faces) and then classify these regions in a single, unified framework.

Faster R-CNN is favored in deepfake detection for its high accuracy in detecting faces even in complex scenarios where faces might be partially obscured or presented in unusual poses. Its ability to handle large images and detect multiple faces in a single image makes it an ideal choice for processing video frames, where the position and appearance of faces can vary significantly. The speed of Faster R-CNN is also a significant advantage in real-time applications, ensuring that face detection does not become a bottleneck in the deepfake detection pipeline.

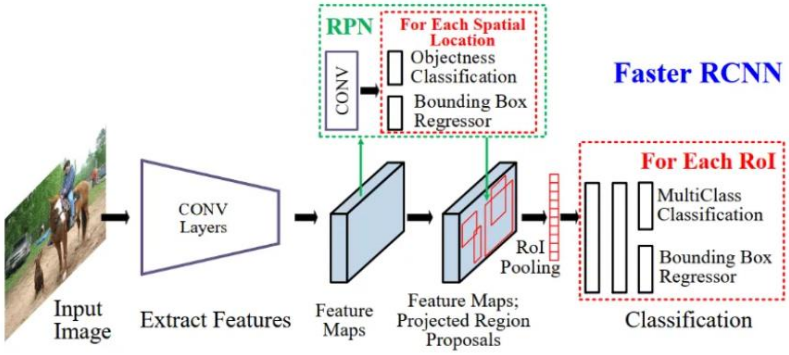


Figure 8: Architecture of the Faster R-CNN Model

Architecture of Faster R-CNN consists of two main components:

1. **Region Proposal Network (RPN):** The RPN is responsible for generating region proposals where objects (in this case, faces) are likely to be found. It slides a small network over the feature map produced by the backbone CNN (e.g., ResNet) and predicts the bounding boxes for objects along with scores indicating the likelihood of an object being present.

2. **Fast R-CNN:** Once the RPN generates the proposals, they are fed into the Fast R-CNN module, which classifies each region and refines the bounding boxes. This module also uses ROI (Region of Interest) pooling to extract fixed-size feature maps from each proposal, which are then passed through fully connected layers for classification and bounding box regression.

The combination of the RPN and Fast R-CNN allows for an end-to-end training process, making Faster R-CNN both fast and effective at detecting faces in diverse conditions.

2.4. Overview of Deep Learning Models

Deep learning models are a subset of machine learning algorithms that use artificial neural networks with many layers—hence the term "deep." These models are designed to automatically learn features from data through multiple layers of abstraction, allowing them to excel in complex tasks such as image recognition, natural language processing, and more. Each layer in a deep learning model transforms the input data into a higher level of abstraction, enabling the model to capture intricate patterns and relationships within the data.

In deepfake detection, deep learning models are essential because they can learn subtle differences between real and manipulated images or videos. By training on large datasets, these models can identify minute inconsistencies that are often imperceptible to humans.

Deep learning models used in deepfake detection work by analyzing images or video frames to identify telltale signs of manipulation. The process generally involves:

1. **Data Preprocessing:** Images or frames are preprocessed, which may include resizing, normalizing, and augmenting data to enhance the model's ability to generalize.
2. **Feature Extraction**: The model automatically extracts features from the input data. These features may include edges, textures, or more complex patterns specific to human faces.
3. **Classification:** The model classifies each image or frame as either "real" or "fake" based on the features extracted. During this phase, the deep learning model leverages patterns learned during training to make predictions.
4. **Model Training and Optimization**: The model is trained on a labeled dataset of real and fake images/videos using backpropagation and optimization algorithms like stochastic gradient descent to minimize the error in predictions.
5. **Prediction and Evaluation**: After training, the model is used to predict the authenticity of new, unseen images or videos. Its performance is evaluated using metrics such as accuracy, precision, and recall.

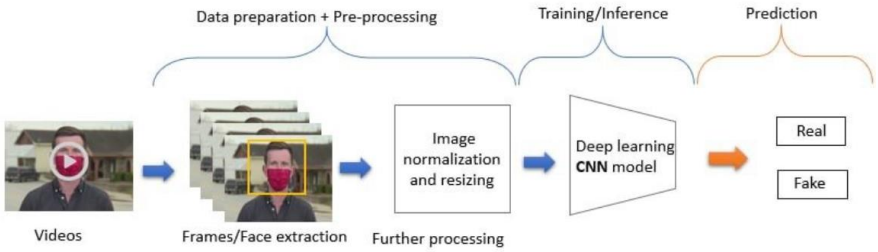


Figure 9: Deepfake Detection Workflow

We have used three different Deep Learning models in our thesis, ResNet-50, EfficientNet-B0 and XceptionNet:

2.4.1. ResNet-50

ResNet-50 is a variant of the ResNet (Residual Network) family, which was introduced by He et al. in 2015. The "50" in ResNet-50 refers to the depth of the network, i.e., it has 50 layers. ResNet-50 is one of the most popular deep learning models for image classification tasks due to its ability to overcome the vanishing gradient problem using residual connections.

ResNet-50 is effective in deepfake detection because of its ability to learn complex features from images while maintaining high accuracy. Its residual connections allow the network to train deeper layers without degradation in performance, making it well-suited for detecting subtle inconsistencies in deepfake videos.

Architecture of ResNet-50:

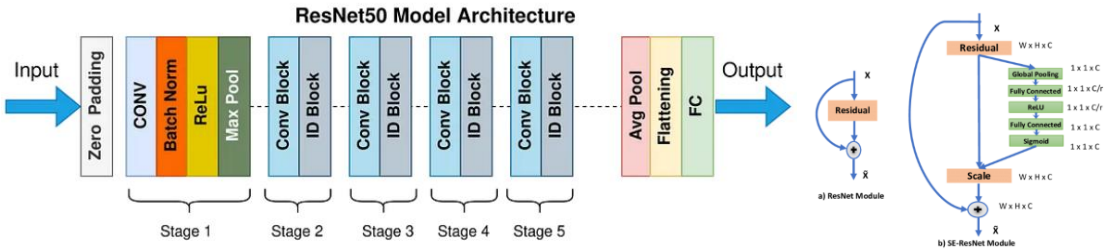


Figure 10: ResNet-50 architecture diagram and Residual Block diagram

- **Residual Blocks:** ResNet-50 is composed of multiple residual blocks, each containing convolutional layers, batch normalization, and ReLU activation functions. The key innovation is the skip connection, which adds the input of a layer to the output of a deeper layer, helping to preserve gradient flow and enabling the network to learn better.
- **Convolutional Layers:** These layers apply filters to the input image, capturing essential features like edges and textures.

- **Pooling Layers:** Used to downsample the image dimensions while preserving critical features.
- **Fully Connected Layer:** At the end of the network, fully connected layers map the learned features to output classes, which, in this case, are "real" or "fake."

2.4.2. EfficientNet-B0

EfficientNet-B0 is the base model of the EfficientNet family, introduced by Tan and Le in 2019. EfficientNet models are designed to achieve high accuracy while being computationally efficient. They use a compound scaling method that uniformly scales the depth, width, and resolution of the network, leading to better performance with fewer resources.

EfficientNet-B0 is used in deepfake detection due to its ability to balance performance and computational cost. It achieves high accuracy with relatively low computational power, making it ideal for real-time applications where resources may be limited.

Architecture of EfficientNet-B0:

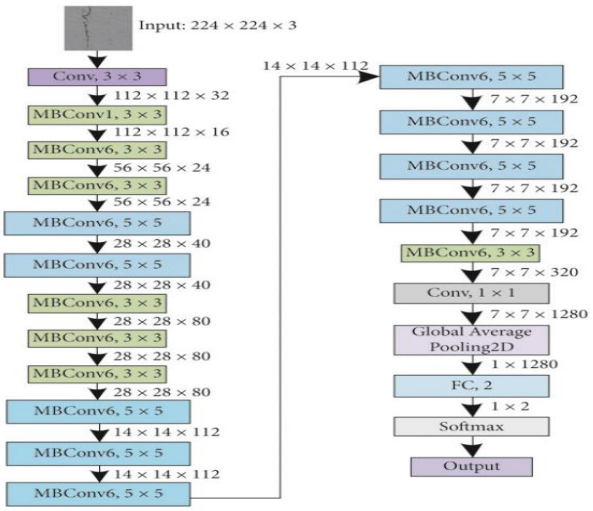


Figure 11: EfficientNet-B0 architecture diagram

- **Mobile Inverted Bottleneck Convolution (MBConv):** EfficientNet-B0 uses MBConv layers, which are lightweight convolutional layers designed for mobile and edge devices. These layers consist of depthwise separable convolutions that reduce the number of parameters and computational load.
- **Swish Activation Function:** EfficientNet-B0 uses the Swish activation function, which has been shown to perform better than ReLU in certain deep learning tasks.

- **Compound Scaling:** The network scales in a balanced way by increasing the depth (number of layers), width (number of neurons in each layer), and resolution (input image size), resulting in better accuracy without excessive resource use.

2.4.3. XceptionNet

XceptionNet, introduced by François Chollet in 2017, is an advanced deep learning model based on the Inception architecture but with depthwise separable convolutions. The name "Xception" stands for "Extreme Inception," as it takes the Inception modules to their logical extreme by completely separating cross-channel and spatial convolutions.

XceptionNet is highly effective in deepfake detection because of its ability to learn fine-grained features through depthwise separable convolutions. This architecture excels in identifying subtle anomalies in images, which is crucial for distinguishing between real and manipulated content.

Architecture of XceptionNet:

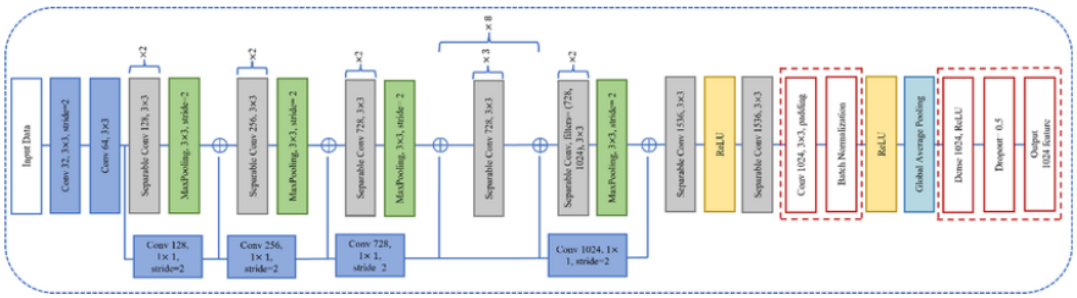


Figure 12: XceptionNet architecture diagram

- **Depthwise Separable Convolutions:** Unlike traditional convolutions, XceptionNet separates the spatial convolution and the depthwise convolution, which improves efficiency and performance.
- **Inception Module:** XceptionNet replaces the Inception modules with depthwise separable convolutions, simplifying the architecture while maintaining the ability to capture complex patterns.
- **Linear Stack of Convolutional Layers:** The network consists of 36 convolutional layers, which are arranged in a linear stack rather than the traditional Inception modules.

2.4.4. Integration of Face Detection Methods with Deep Learning Models

Integrating face detection methods with deep learning models is a powerful approach for enhancing the accuracy and efficiency of tasks like deepfake detection, facial recognition, and identity verification. The integration involves using a face detection algorithm to accurately locate and crop the face from an image or video frame, followed by feeding this cropped region into a deep learning model for further analysis, such as classification, feature extraction, or manipulation detection.

1. What is Integration of Face Detection with Deep Learning?

The integration process typically involves two main stages:

- **Face Detection**: In this stage, algorithms like MTCNN or Faster R-CNN are used to detect the presence and location of faces in an image or video. These algorithms identify facial landmarks and bounding boxes around faces, providing crucial data for the next stage.
- **Deep Learning Analysis**: Once the face is detected and localized, the cropped facial region is passed to a deep learning model, such as ResNet50, EfficientNet-B0, or XceptionNet. These models are trained to analyze facial features, classify images, or detect anomalies (like deepfakes).

This integration ensures that only the relevant parts of the image are processed by the deep learning model, significantly improving the model's performance and reducing computational costs.

2. Detailed Examples of Integration

Example 1: MTCNN + XceptionNet:

- **MTCNN (Multi-task Cascaded Convolutional Networks)**: MTCNN is a popular face detection method that detects facial landmarks through a three-stage process: Proposal Network (P-Net), Refine Network (R-Net), and Output Network (O-Net). MTCNN not only detects faces but also aligns them, making it ideal for integration with deep learning models that require precise input.
- **XceptionNet**: XceptionNet is a deep learning model that uses depthwise separable convolutions to efficiently extract features from the input images. It's known for its high

performance in image classification and anomaly detection, making it suitable for tasks like deepfake detection.

System Architecture:

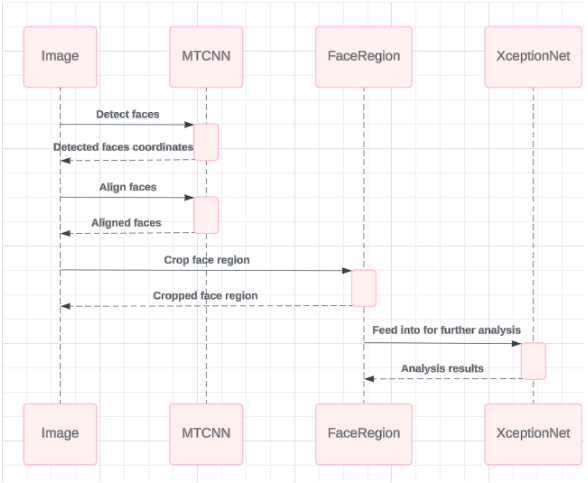


Figure 13: Integrated MTCNN XceptionNet Sequence Diagram

- **Step 1: Face Detection**: MTCNN is applied to each frame or image to detect faces and facial landmarks. It generates bounding boxes around the detected faces.
- **Step 2: Face Alignment**: MTCNN aligns the face by correcting the orientation based on the detected landmarks.
- **Step 3: Cropping the Face Region**: The face is cropped out of the image based on the bounding box generated by MTCNN.
- **Step 4: Feature Extraction with XceptionNet**: The cropped and aligned face is passed to XceptionNet, which extracts high-dimensional features.
- **Step 5: Classification/Detection**: XceptionNet analyzes the extracted features to classify the face as real or fake (in the case of deepfake detection) or perform other relevant tasks.

Example 2: Faster R-CNN + ResNet50

- **Faster R-CNN (Region Convolutional Neural Network)**: Faster R-CNN is an object detection algorithm that uses region proposal networks (RPNs) to generate candidate object regions (in this case, faces) and then classifies these regions. It is highly accurate and efficient for detecting faces in complex images.

- **ResNet50**: ResNet50 is a deep residual network that is widely used for image classification and feature extraction. It excels in handling deep layers by using skip connections to avoid the vanishing gradient problem, making it suitable for complex tasks like deepfake detection.

System Architecture:

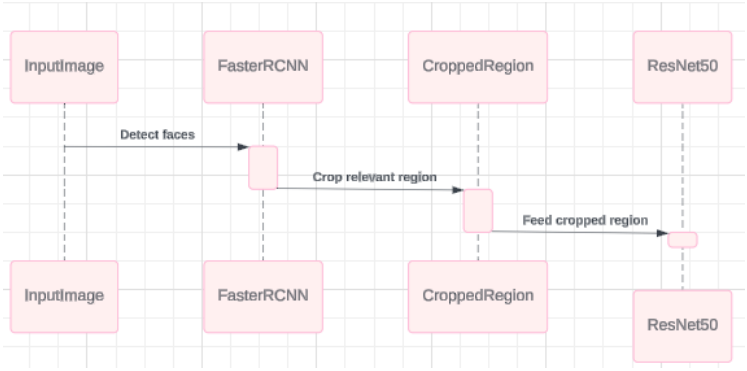


Figure 14: Integrated Faster R-CNN ResNet50 Sequence Diagram

- **Step 1: Face Detection with Faster R-CNN**: The input image or video frame is processed by Faster R-CNN, which uses its RPN to generate potential face regions. These regions are then classified, and the most relevant bounding box (the face) is selected.
- **Step 2: Cropping the Face Region**: The face is cropped based on the bounding box coordinates provided by Faster R-CNN.
- **Step 3: Feature Extraction with ResNet50**: The cropped face is then passed through ResNet50, which extracts features like texture, color, and geometry.
- **Step 4: Classification/Detection**: ResNet50 processes the features to determine whether the face is real or fake (in deepfake detection) or for other tasks.

2.5 Transfer Learning

2.5.1. Introduction to Transfer Learning

Transfer learning is a machine learning technique where a model developed for a particular task is reused as the starting point for a model on a second task. This method is especially effective when the second task has limited data. Instead of training a model from scratch, transfer learning allows you to leverage the pre-trained weights of a model, usually trained on large datasets like ImageNet, to perform well on a different but related task.

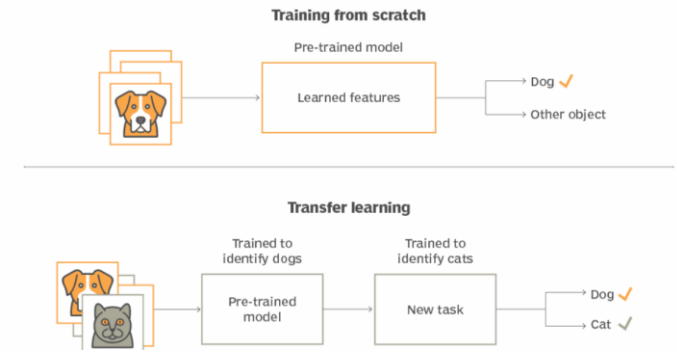


Figure 15: Transfer Learning Workflow

In essence, transfer learning transfers knowledge from one domain (e.g., image classification on ImageNet) to another domain (e.g., deepfake detection). This is particularly useful in cases where collecting and labeling a large dataset for training is challenging or impractical.

2.5.2. Deepfake Detection Context

In the context of deepfake detection, transfer learning is employed due to its ability to overcome the limitations of small and specialized datasets. In this project, transfer learning was applied to both ResNet50 and XceptionNet. Deepfake detection often requires distinguishing between subtle differences in real and fake images, which can be difficult without a large amount of training data. By using models like ResNet50 and XceptionNet, which are pre-trained on extensive datasets, transfer learning enables these models to effectively extract high-level features from images or video frames. These pre-trained models understand general image characteristics, such as edges, textures, and shapes, making them well-suited for adapting to the task of detecting deepfakes. Fine-tuning these models on a specific deepfake dataset not only enhances accuracy but also significantly reduces training time, which is crucial in research and production environments.

2.6. Datasets Used for Deepfake Detection

Deepfake detection has become a crucial area of research due to the rise of manipulated media and the potential harm it can cause. Several datasets have been developed to aid in the training and evaluation of deepfake detection algorithms. These datasets vary in size, quality, and the types of manipulations they include.

	Real		Fake				
Dataset	Video	Frame	Video	Frame	Generation Method	Release Date	Generation Group
UADFV	49	17.3K	49	17.3K	FakeAPP	11/2018	1st
DF-TIMIT	320	34K	320	34K	Faceswap-GAN	12/2018	1st
*Real & Fake Face Detection	1081	405.2K	960	399.8K	Expert-generated high-quality photoshopped	01/2019	2st
FaceForensics++	1000	509.9k	1000	509.9K	DeepFakes, Face2Face, FaceSwap, NeuralTextures	01/2019	2nd
DeepFakeDetection	363	315.4K	3068	2242.7K	Similar to FaceForensics++	09/2019	2nd
DFDC	1131	488.4K	4113	1783.3K	Deepfake, GAN-based, and non-learned methods	10/2019	2nd
Celeb-DF	590	225.4K	5639	2116.8K	Improved DeepFake synthesis algorithm	11/2019	2nd
*140K Real & Fake Faces	70K	15.8M	70K	15.8M	StyleGAN	12/2019	2nd
DeeperForensics	50,000	12.6M	10,000	2.3M	Newly proposed end-to-end face swapping framework	06/2020	2nd

Table 1: Deepfake dataset comparison table

Below is an overview of the most significant datasets used in deepfake detection research.

2.6.1. FaceForensics++

FaceForensics++ is one of the most widely used datasets in the field of deepfake detection. It consists of over 1.8 million manipulated images extracted from 1,000 real videos and their corresponding deepfake versions. The dataset includes manipulations created using four popular methods: DeepFakes, Face2Face, FaceSwap, and NeuralTextures. FaceForensics++ is particularly valued for its diversity and large scale, making it a benchmark for many deepfake detection studies. The dataset also includes videos at different compression levels, which is important for testing the robustness of detection algorithms under real-world conditions where media is often compressed before sharing.

2.6.2. DeepFake Detection Challenge Dataset

DeepFake Detection Challenge Dataset (DFDC) was released by Facebook in collaboration with other research institutions as part of a global challenge to advance deepfake detection technologies. The DFDC dataset is one of the largest datasets available, with over 100,000 videos, both real and fake. These videos feature diverse subjects, environments, and lighting conditions, making the dataset highly representative of real-world scenarios. The dataset includes deepfakes created using various techniques, ensuring that models trained on it can generalize well across different types of manipulations.

2.6.3. Celeb-DF

Celeb-DF is another popular dataset that contains over 590 real videos collected from YouTube and their corresponding deepfake versions. The deepfakes in this dataset are generated using an improved deepfake synthesis method that reduces visual artifacts, making them more

challenging to detect. Celeb-DF is known for its high-quality deepfake videos that closely resemble real videos, making it a valuable resource for training and evaluating detection algorithms, particularly those aiming to detect more sophisticated deepfakes.

2.7 Challenges in Deepfake Detection

Despite advancements in deepfake detection, several challenges remain. For example, deepfakes generated using advanced GANs can bypass traditional detection methods, necessitating the development of more sophisticated models. Additionally, the generalization of detection models to unseen data remains a significant hurdle. Studies like Tolosana et al. (2020) emphasize the need for models that can adapt to new deepfake generation techniques, highlighting the ongoing challenges in this field.

3. Methodology

This section details the methodology used to develop a deepfake detection system leveraging facial biometrics. It encompasses the overall model structure, dataset used, data preparation steps, and training methodology. By providing a clear and structured overview of these components, the methodology aims to offer a comprehensive framework for evaluating and enhancing the performance of deepfake detection systems. This framework emphasizes accuracy, robustness, and effectiveness, ensuring that the developed models are capable of distinguishing between genuine and manipulated facial content. Flow diagrams and block diagrams are employed to visually represent the system's architecture and processes.

3.1. Dataset Description

3.1.1. Overview of FaceForensics++ Dataset

FaceForensics++ is a comprehensive and widely used dataset in the field of digital forensics, particularly focused on the detection of deepfakes. Deepfakes are synthetic media where a person in an existing image or video is replaced with someone else's likeness, raising significant concerns due to their potential misuse in spreading misinformation, creating fake news, and other malicious activities. As a result, the need for reliable detection methods has become critical.

FaceForensics++ was introduced to address the challenges posed by deepfake detection by providing a standardized and extensive dataset for researchers to develop and test their algorithms. Created by researchers at the Technical University of Munich, the dataset consists

of thousands of video sequences that have been manipulated using different techniques such as DeepFakes, Face2Face, FaceSwap, and NeuralTextures, making it one of the most robust resources available for deepfake detection research.

The dataset was downloaded from: https://github.com/ondyari/FaceForensics

3.1.2. Structure of the Dataset and Its Composition

Video-Based Dataset: FaceForensics++ is primarily a video-based dataset. It contains over 1,000 original video sequences sourced from 977 YouTube videos of news anchors and celebrities speaking in front of a camera. These videos were chosen to represent a variety of conditions, including different lighting, camera quality, and motion, ensuring a diverse and challenging dataset.

Manipulation Techniques: The dataset includes four different types of facial manipulation techniques, resulting in a total of 4,000 manipulated video sequences:

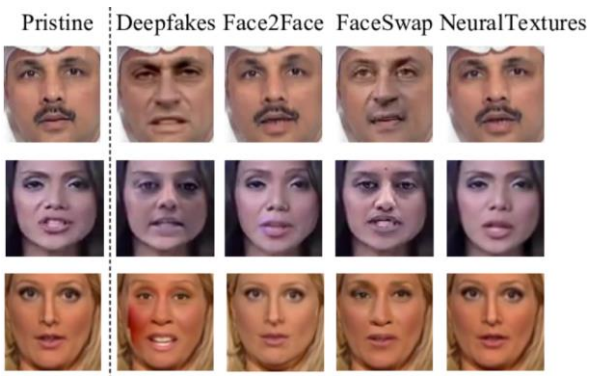


Figure 16: Faceforensics++ Dataset Overview

- **Deepfakes**: Generated using popular deepfake models, these videos involve swapping the face of a person in a video with another person's face.
- **Face2Face**: A method that modifies the expressions of a person in the video by mapping the facial expressions of an actor onto the target's face in real-time.
- **FaceSwap**: This involves swapping faces between two people in a video, typically using simpler computer vision techniques compared to deepfakes.
- **NeuralTextures**: A technique that combines computer graphics with neural networks to create realistic facial expressions and textures.

For each of these manipulation methods, the dataset provides 1,000 videos, ensuring an equal distribution across the different techniques.

Quality and Compression Levels: FaceForensics++ offers video data in both high-quality (raw) format and various compressed formats (H.264 codec) at different compression levels (c23, c40). The total dataset size is approximately **470 GB** in raw format, with the compressed versions significantly reducing the size depending on the compression level.

3.1.3. Real vs. Fake Video Sample

Figure 17: Faceforensics ++ Real vs. Fake Samples

Real Videos: Original, unaltered video sequences from the dataset, providing the baseline for detecting manipulations.

Fake Videos: Manipulated sequences created using different face-swapping and reenactment techniques. These videos mimic realistic facial expressions, movements, and features, making it challenging to distinguish them from the originals.

PCA Visualization of Real vs. Fake Faces:

The diagram above is a PCA (Principal Component Analysis) plot that shows the distribution of real (blue) and fake (red) face images from the FaceForensics++ dataset. PCA reduces the complexity of the data by projecting it onto two dimensions, helping us visualize how different the real and fake faces are.

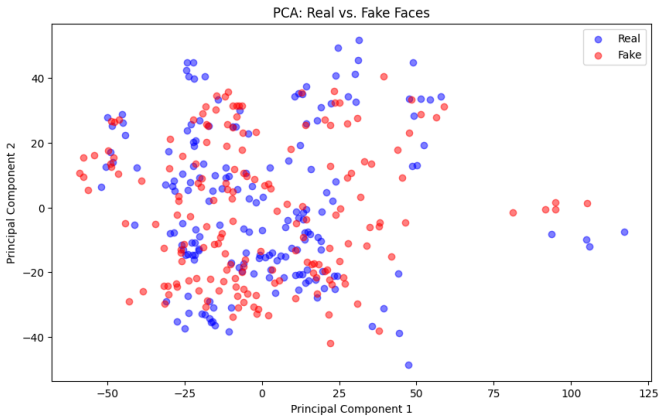


Figure 18: PCA Visualization of Real vs. Fake Faces

In this plot, there's some overlap between the red and blue points, suggesting that the model may find it challenging to tell the real and fake faces apart in certain cases.

This visualization helps us understand how well our model is performing in differentiating real from fake faces based on the features we've extracted.

Pixel Intensity Distribution:

The diagram shows pixel intensity distributions for real and fake faces from the FaceForensics++ dataset, crucial in deepfake detection.

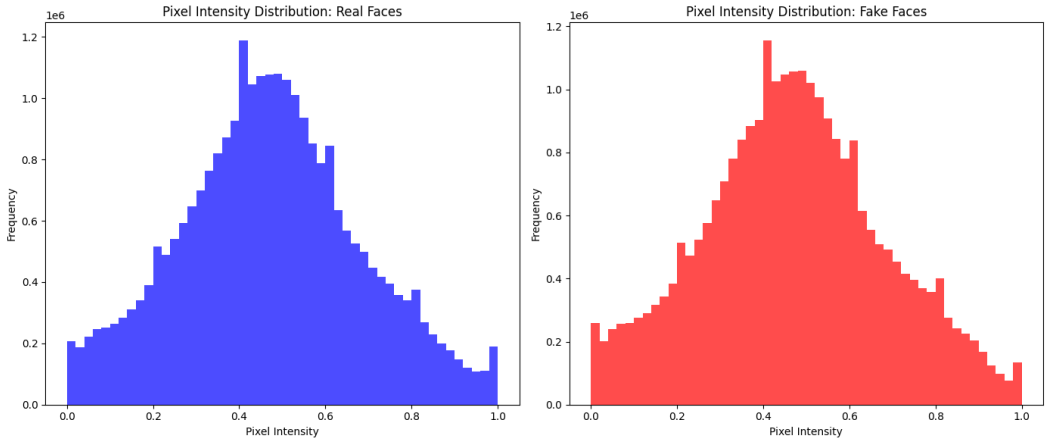


Figure 19: Pixel Intensity Distribution

- **Real Faces (Left, Blue)**: The pixel intensities form a smooth, bell-shaped curve centered around mid-range values (about 0.45 to 0.55). This indicates a natural distribution of light and shadow in authentic images, with a balance across dark, mid-tone, and bright pixels.
- **Fake Faces (Right, Red)**: The distribution is similar but slightly skewed, particularly with more variation in the lower intensity range. This suggests that fake images,

generated by models like GANs, often have unnatural shading or over-smoothed areas, leading to these subtle differences.

These differences in pixel intensity distributions are key indicators that can be used to distinguish between real and fake images in deepfake detection algorithms.

3.2. Data Preprocessing

In this research, the data preprocessing pipeline is meticulously designed to prepare the FaceForensics++ dataset for deepfake detection using facial biometrics. In this research, a subset of the FaceForensics++ dataset was utilized, specifically 50 video sequences. This selection includes a balanced mix of original and manipulated videos across the four primary manipulation techniques: Deepfakes, Face2Face, FaceSwap, and NeuralTextures and the videos I have used were downloaded at the c23 compression level. The preprocessing steps include video frame extraction, face detection and alignment, and normalization.

Below is a detailed explanation of each step:

1. **Video to Frame Extraction:**

   The first step in the data preprocessing pipeline involves converting video sequences from the FaceForensics++ dataset into individual frames. This step is crucial because it transforms the continuous video data into discrete images, which are more manageable for processing and analysis.

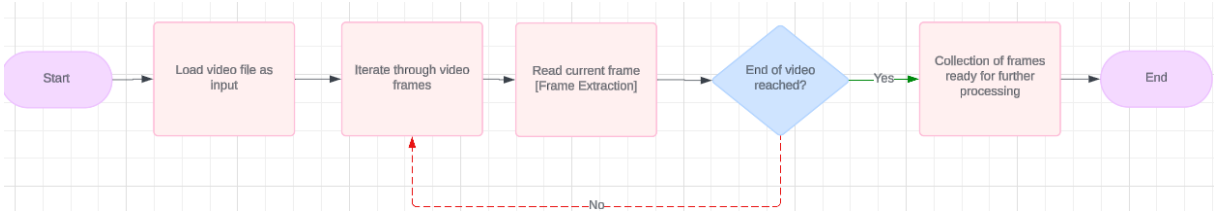


Figure 20: Video to frame extraction flowchart

The video files are loaded one by one, and frames are extracted at a consistent rate of 30 frames per second.

Figure 21: Bounding boxes after loading data

This frame rate was chosen to ensure that sufficient visual data is captured, allowing the detection model to analyze a diverse set of facial expressions, poses, and lighting conditions present in each video. Each frame is saved as an image, which is then passed on to the next preprocessing stage.

2. Face Detection and Alignment:

Once the frames have been extracted, the next step is to detect and align faces within these images. This is done using the MTCNN (Multi-task Cascaded Convolutional Networks) face detection algorithm, which is highly effective at identifying faces in various poses and under different lighting conditions.

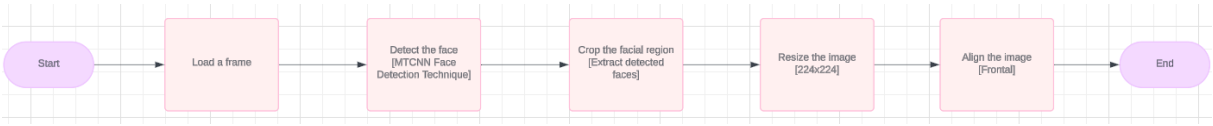


Figure 22: Face Detection & Alignment flowchart

The MTCNN model processes each frame to detect faces by placing a bounding box around the facial region. Once detected, each face is cropped from the frame. To ensure that the model receives consistently oriented faces, these cropped images are then resized to a standard dimension of 224x224 pixels and aligned so that the facial features (such as eyes, nose, and mouth) are positioned similarly across all images. This alignment step is crucial as it reduces variability in the dataset, making it easier for the model to learn relevant patterns related to deepfakes.

3. Data Normalization:

After face detection and alignment, the images undergo normalization. The primary goal of normalization is to scale the pixel values of the images to a uniform range, which in this case is between 0 and 1. This step is vital because it standardizes the input data, ensuring that the model processes images consistently.

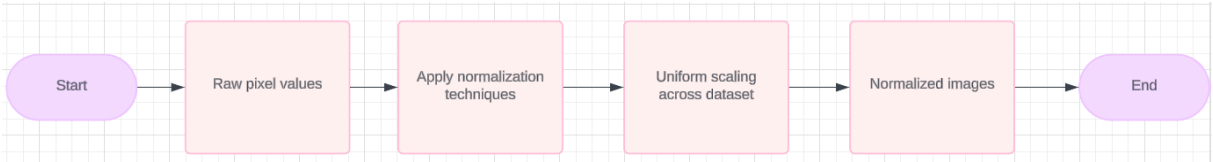


Figure 23: Data normalization flowchart

The normalized images retain all the important facial features but are scaled to ensure that variations in lighting or other environmental factors do not adversely affect the model's learning process.

4. Data Augmentation:

Data augmentation involves creating additional training data by applying random transformations to the existing images, such as rotations, flips, or changes in brightness. This step can enhance the robustness of the model by exposing it to a wider variety of possible inputs, thereby reducing the risk of overfitting.

If included, the data augmentation process would follow normalization in the preprocessing pipeline.

5. Data Splitting:

The dataset used in this deepfake detection project was divided into training, validation, and testing subsets with the following proportions:

- **Training Set:** 80% of the extracted facial images from both real and fake videos, along with their corresponding labels.
- **Validation Set:** 10% of the extracted facial images and labels, used to fine-tune hyperparameters and prevent overfitting during training.
- **Testing Set:** 10% of the extracted facial images and labels, reserved for evaluating the final model's performance and generalization ability.

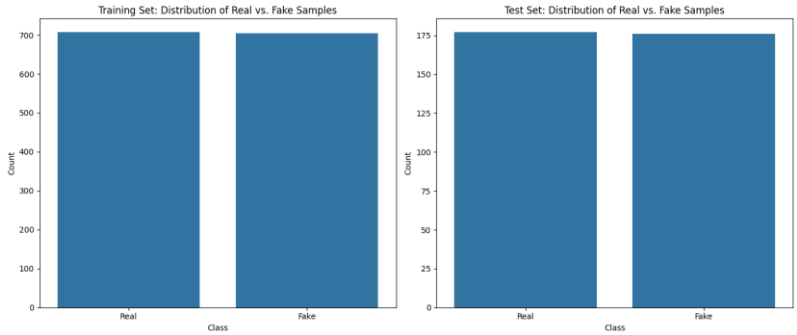


Figure 24: Distribution of Real vs Fake Samples in training and test set

This data split ensures a comprehensive evaluation framework, facilitating effective model training, hyperparameter optimization, and robust assessment of the model's ability to generalize to unseen data.

3.3. Model Architecture

3.3.1 Model Selection (ResNet-50, EfficientNet-B0, XceptionNet)

In this thesis, three advanced deep learning models were selected for deepfake detection using facial biometrics: **ResNet50**, **Xception**, and **EfficientNetB0**. Each model was chosen based on its unique architecture and its effectiveness in handling complex image classification tasks. These models were implemented using TensorFlow's Keras API, which provides a flexible and modular framework for building and training neural networks. Below is a detailed explanation of each model, including the layers used and the rationale for their selection.

3.3.1.1. ResNet-50

ResNet50 is renowned for its introduction of residual connections, which allow the network to be significantly deeper without suffering from the vanishing gradient problem. This architecture is particularly effective at learning intricate patterns from images, making it a strong candidate for detecting subtle manipulations in deepfakes.

Architecture:

- **ResNet50 Base:** Pre-trained on ImageNet, consisting of 50 layers with residual connections.
- **GlobalAveragePooling2D Layer:** Averages the spatial dimensions of the feature maps to produce a single vector for each feature map.
- **Dense Layer:** 256 units, ReLU activation - adds non-linearity and helps in learning complex representations.
- **Dropout Layer:** 50% dropout rate to prevent overfitting by randomly disabling neurons during training.
- **Output Layer:** 1-unit, sigmoid activation - produces a probability score for binary classification (real vs. fake).
- **Total Layers:** The ResNet50 base model has 50 layers, followed by 3 custom layers, making a total of 53 layers.

Rationale: ResNet50 was selected for its ability to maintain performance as network depth increases, which is crucial for extracting fine details from facial images in deepfake detection. The residual connections in ResNet50 help in training deeper networks by allowing gradients to flow through the network more effectively.

3.3.1.2. XceptionNet

Xception(Extreme Inception) is an advanced architecture that extends the ideas of the Inception model by replacing standard inception modules with depthwise separable convolutions. This modification results in a model that is both more efficient and more powerful, capable of capturing fine-grained details in images.

Architecture:

- **Xception Base:** Pre-trained on ImageNet, utilizing depthwise separable convolutions and including 36 convolutional layers grouped into 14 modules.
- **GlobalAveragePooling2D Layer:** Reduces the dimensionality of the feature maps by averaging across spatial dimensions.
- **Dense Layer:** 256 units, ReLU activation - helps in distinguishing subtle differences between real and fake facial features.
- **Dropout Layer:** 50% dropout rate to prevent overfitting.
- **Output Layer:** 1-unit, sigmoid activation - provides a binary output for classification.
- **Total Layers:** The Xception base model has 36 convolutional layers, and with the addition of 3 custom layers, the model comprises a total of 39 layers.

Rationale: Xception was chosen for its advanced architecture, which efficiently captures complex features with fewer parameters than traditional convolutional layers. Its depthwise separable convolutions make it particularly well-suited for high-dimensional image data like facial biometrics used in deepfake detection.

3.3.1.3. EfficientNetB0

EfficientNetB0 is part of the EfficientNet family, which scales the network's width, depth, and resolution in a balanced way. This model is known for achieving high accuracy with fewer parameters and lower computational cost compared to other models, making it an efficient choice for deepfake detection tasks.

Architecture:

- **EfficientNetB0 Base:** Pre-trained on ImageNet, includes a series of inverted residual blocks and squeeze-and-excitation blocks designed to capture important features efficiently.

- **GlobalAveragePooling2D Layer:** Averages each feature map into a single value, reducing dimensionality.
- **Dense Layer:** 256 units, ReLU activation - helps in capturing and distinguishing complex patterns in facial features.
- **Dropout Layer:** 50% dropout rate to avoid overfitting.
- **Output Layer:** 1-unit, sigmoid activation - predicts the probability of an image being real or fake.
- **Total Layers:** The EfficientNetB0 base model includes 16 layers, and with the additional 3 custom layers, the total number of layers is 19 layers.

Rationale: EfficientNetB0 was selected for its balance between accuracy and computational efficiency. Its scalable architecture allows it to maintain high performance even with fewer parameters, which is beneficial in scenarios with limited computational resources or when rapid inference is required, such as in real-time deepfake detection.

Each model was chosen based on its architecture's ability to capture essential details in facial images, which is crucial for accurate deepfake detection. ResNet50 provides a robust deep architecture with residual connections, Xception leverages depthwise separable convolutions for efficient and effective feature extraction, and EfficientNetB0 offers a computationally efficient model that does not compromise on accuracy.

3.3.2. Frozen/Unfrozen Layers

During the training process, all three models (ResNet50, Xception, EfficientNetB0) initially had their base layers frozen. This strategy leverages the power of transfer learning by preserving the pre-trained weights from ImageNet, ensuring that the models start with a strong foundation for feature extraction. Fine-tuning was then performed by unfreezing the last 10 layers of each model, allowing them to adjust to the specific nuances of the deepfake detection task. Below is a summary of the frozen/unfrozen layer status after fine-tuning:

- **Frozen Layers:** All layers except the last 10 in the base models (specific layers vary depending on the model).

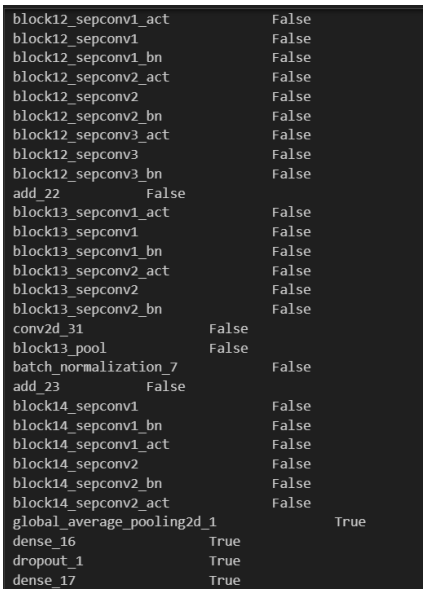


Figure 25: Frozen layers of Xception model

- **Unfrozen Layers:** The last 10 layers of each model, as well as the custom layers added for the deepfake detection task, were unfrozen and made trainable.

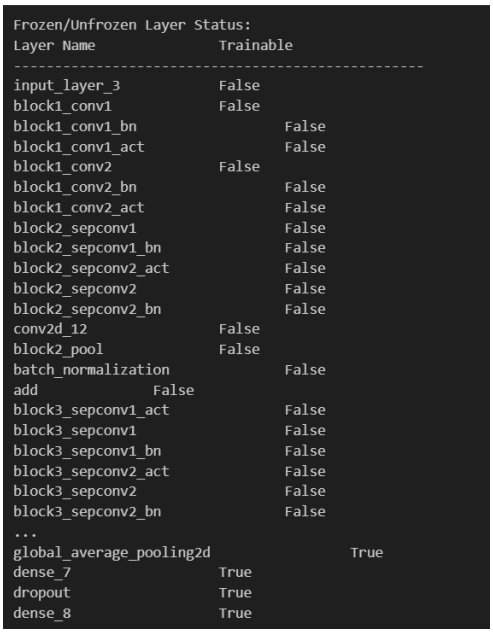


Figure 26: Unfrozen layers of Xception model

This strategy ensures that the models are both powerful and adaptable, capable of leveraging pre-trained knowledge while also fine-tuning to the specific dataset used in this thesis.

3.4. Model Training and Evaluation Process

3.4.1. Model Training

The training process for the deep learning models in the deepfake detection task was meticulously structured to optimize performance while minimizing the risk of overfitting. The data that was preprocessed in the previous step, including extracted and augmented facial frames, served as the foundation for this phase. The training process incorporated multiple strategies, including early stopping, model checkpointing, and dynamic learning rate scheduling, to ensure that the models not only performed well but also generalized effectively.

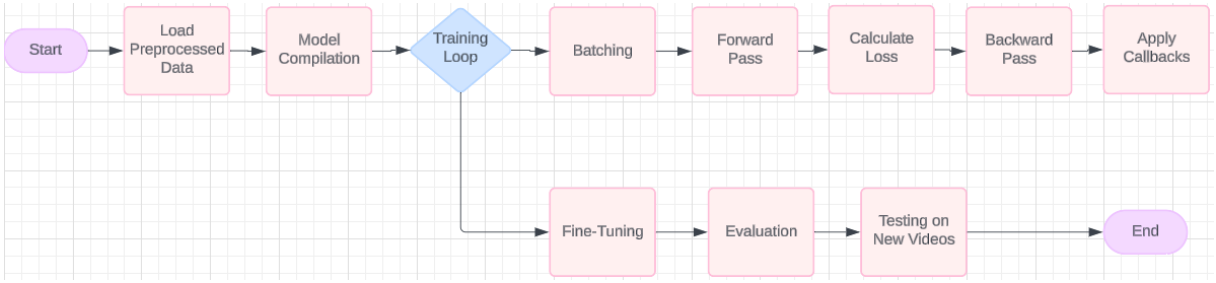


Figure 27: Model Training Pipeline

Model Compilation: The models were compiled using the Adam optimizer, which is well-regarded for its ability to handle sparse gradients and noisy data. An initial learning rate of 0.0001 was chosen, tailored to the specific needs of the deepfake detection task and the model architecture being used. The loss function selected was binary cross-entropy, which is appropriate for the binary classification task of distinguishing between real and fake videos.

Training and Callbacks: Training was conducted over multiple epochs, typically up to 50, with a batch size of 32. This setup ensured that the models had ample opportunity to learn complex patterns without overfitting. To further optimize the training process, several callbacks were employed:

- **ModelCheckpoint**: This callback automatically saved the model with the best validation accuracy during training, ensuring that the most optimal version of the model was retained.
- **EarlyStopping**: By monitoring the validation loss, this callback halted training when no further improvements were observed, thereby preventing overfitting and saving computational resources.

- **LearningRateScheduler**: This callback dynamically adjusted the learning rate during training, allowing for larger learning rates during the early stages for faster convergence, and smaller learning rates in the later stages for fine-tuning.

Initially, the models were trained with the custom layers added on top of the ResNet50 architecture while keeping the base ResNet50 layers frozen. This approach leveraged the pre-trained knowledge of ResNet50 while also adapting to the specific characteristics of the deepfake detection task. After the initial phase of training, the last 10 layers of the ResNet50 base model were unfrozen to allow fine-tuning. This fine-tuning phase was conducted with a reduced learning rate of 0.00001, facilitating minor adjustments to the deeper layers of the network without destabilizing the learning process. The models were then retrained, focusing on fine-tuning these layers to capture more task-specific features, which are crucial for distinguishing subtle differences between real and fake faces.

Evaluation and Visualization: After training, the models were evaluated on the test set using a comprehensive set of metrics, including accuracy, precision, recall, F1-score, and the confusion matrix. These metrics provided a detailed understanding of the model's performance and its ability to generalize to unseen data.

ResNet-50 Sample Predictions:

Figure 28: ResNet-50 Sample Predictions

EfficientNet-B0 Sample Predictions:

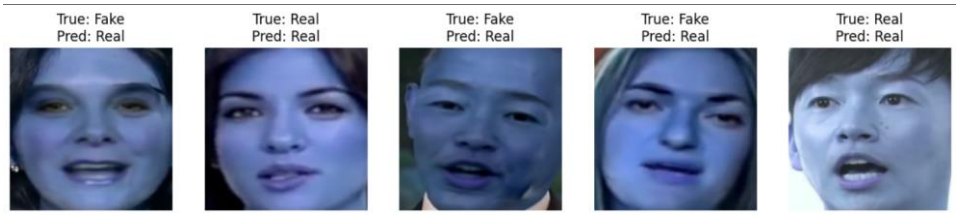


Figure 29: EfficientNet-B0 Sample Predictions

XceptionNet Sample Predictions:

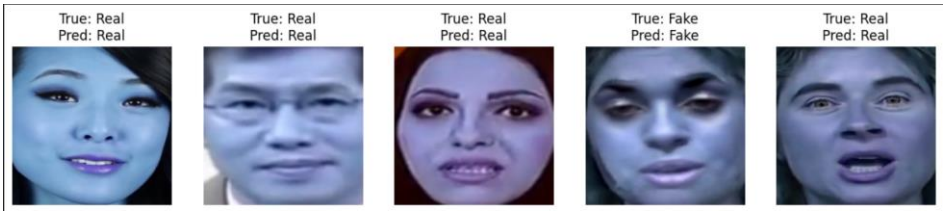


Figure 30: XceptionNet Sample Predictions

sample predictions were visualized to compare the predicted labels with the true labels for a selection of test samples, offering insights into the model's decision-making process.

Testing on New Videos: To validate the practical applicability of the models, they were tested on new video samples from the dataset. The video frames were processed similarly to the training and testing phases—faces were detected using MTCNN or Faster R-CNN, extracted, and then classified as real or fake based on the model's predictions. This step was crucial in demonstrating the model's capability to handle real-world data, providing insights into its robustness and reliability across various scenarios.

Figure 31: Unseen video prediction and confidence score

3.4.2. Hyperparameter Tuning

Extensive hyperparameter tuning was conducted to identify the optimal settings for each model. The experimental results provided the following insights:

- **Learning Rate**: A learning rate of 0.0001 with dynamic scheduling consistently yielded the best results, particularly when fine-tuning pre-trained models like ResNet50 and XceptionNet.

- **Batch Size**: A batch size of 32 was effective across all models, balancing computational load and training stability.
- **Dropout Rate**: A dropout rate of 0.5 was used to prevent overfitting, which led to improved validation accuracy and lower validation loss.
- **Epochs**: Training for up to 50 epochs with EarlyStopping allowed the models to converge effectively without overfitting, ensuring that training was both thorough and efficient.
- **Optimizer**: The Adam optimizer was generally the most effective, outperforming alternatives like SGD in terms of validation accuracy and overall model performance.
- **Image Size**: Larger input image sizes (224x224) provided better performance by retaining more detailed facial features, though they increased training time.
- **Trainable Layers**: Allowing more layers to be trainable, particularly in the fine-tuning phase, significantly improved model performance, as it enabled the models to adapt more fully to the nuances of the deepfake detection task.
- **Frame Rate**: A consistent frame rate of 30 frames per second (fps) was maintained during frame extraction, which provided a good balance between dataset size and computational efficiency.

3.4.3. Model Evaluation and Metrics

In this study, several key metrics were used to evaluate the performance of the deep learning models, including accuracy, precision, recall, F1-score, and confusion matrices. These metrics provide a comprehensive assessment of the models' ability to correctly classify video frames as either real or fake, ensuring robust detection of deepfakes in various scenarios.

1. Accuracy:

Accuracy is the ratio of correctly predicted instances to the total instances. It is a basic metric that gives an overall measure of the model's performance.

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

Where:

- TP (True Positives): The number of correct positive predictions.
- TN (True Negatives): The number of correct negative predictions.
- FP (False Positives): The number of incorrect positive predictions.
- FN (False Negatives): The number of incorrect negative predictions.

2. Precision:

Precision is the ratio of true positive predictions to the total number of positive predictions (both true and false). It indicates the accuracy of the model's positive predictions, showing how many of the frames classified as fake are actually fake.

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

3. Recall (Sensitivity or True Positive Rate):

Recall measures the ratio of true positive predictions to the total number of actual positives. It reflects the model's ability to detect all fake frames in the dataset, ensuring that the model doesn't miss many deepfakes.

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

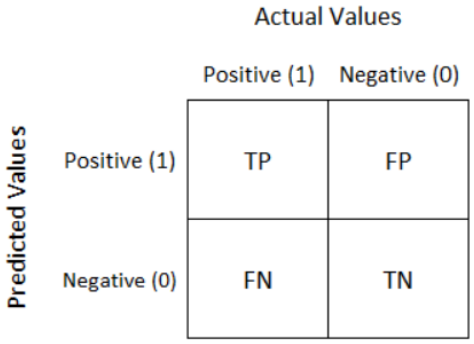
4. F1-Score:

The F1-score is the harmonic mean of precision and recall, providing a balance between these two metrics. It is particularly useful when dealing with imbalanced datasets, where one class (real or fake) may be more prevalent than the other.

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

5. Confusion Matrix:

The confusion matrix is a table that describes the performance of the classification model in more detail by comparing the actual and predicted classes. Each cell in the matrix represents the number of predictions made by the model that belong to the corresponding category:



- True Positives (TP): The number of correct fake predictions.
- False Negatives (FN): The number of actual fake frames incorrectly predicted as real.
- False Positives (FP): The number of actual real frames incorrectly predicted as fake.
- True Negatives (TN): The number of correct real predictions.

The confusion matrix is particularly valuable for understanding the types of errors the model is making. By analyzing the confusion matrix, we can determine whether the model is more prone to certain types of errors, such as false positives (misclassifying real frames as fake) or false negatives (missing actual deepfakes) and adjust the model or data processing approach accordingly.

3.5. Experimental Setup

3.5.1. Software Environment

The experiments are conducted using Python and popular deep learning frameworks such as TensorFlow and Keras. Key libraries and tools include:

- **TensorFlow/Keras:** For building, training, and evaluating deep learning models.
- **OpenCV:** For video processing and face extraction tasks.
- **Scikit-learn:** For dataset splitting, metrics evaluation, and hyperparameter tuning.
- **Pandas and NumPy:** For data manipulation and numerical computations.
- **Matplotlib and Seaborn:** For plotting training results, confusion matrices, and performance metrics.

3.5.2. Hardware Specifications

The experiments are conducted on a high-performance computing environment to handle the large dataset and complex models efficiently. Key hardware components include:

- **GPU:** NVIDIA GPUs (RTX 4050) is used to accelerate model training and inference.
- **RAM:** Sufficient memory (16GB) is allocated to handle large datasets and multiple concurrent processes.
- **Storage:** SSD storage (512GB) is utilized for fast data loading and storage of training checkpoints.

All experiments were run using Visual Studio Code (VSCode) as the development environment.

4. Results

In this section, we present the results of our experiments, categorized by the combination of face detectors (MTCNN, Faster R-CNN) and deep learning models (ResNet50, EfficientNetB0, XceptionNet). For each combination, we discuss the hyperparameters used, the strategies implemented to overcome underfitting and overfitting, and any improvements observed through iterative experimentation. Additionally, we provide a detailed error analysis, compare our findings with those reported in the base paper, and address the implementation challenges encountered throughout the process.

4.1. Model Performance

Two face detection techniques (MTCNN & Faster R-CNN) are tested with three different deep learning models (ResNet-50, EfficientNet-B0, XceptionNet) and the results are as follows:

4.1.1. MTCNN + ResNet-50

model name	trainable layers	image_size	frame_rate	batch_size	learning rate	epochs	dropout rate	optimizer	train accuracy	val accuracy	train_loss	val_loss	num samples	training_time minutes
ResNet50	175	224x224	30	32	0.0001	50	0.5	Adam	54.03%	65.72%	0.6878	0.684	1765	23.51
ResNet50	175	224x224	30	32	0.0001	50	0.4	Adam	48.94%	50.14%	0.7002	0.6945	1765	5.69
ResNet50	175	128x128	24	32	0.001	100	0.5	Adam	49.78%	53.66%	0.6912	0.6859	1765	47.78
ResNet50	175	224x224	30	16	0.0001	50	0.4	SGD	60.12%	70.83%	0.6507	0.7699	1765	12.2
ResNet50	132	224x224	30	64	0.001	50	0.3	Adam	70.15%	75.93%	0.6129	0.7384	1765	30.15
ResNet50	132	224x224	30	32	0.0001	50	0.5	Adam	98.01%	93.76%	0.1037	0.1847	1765	39.25
ResNet50	175	224x224	30	32	0.001	100	0.3	Adam	80.13%	83.45%	0.5689	0.5923	1765	35.7

Table 2: MTCNN + ResNet-50 performance results on the FaceForensics++ dataset

The combination of MTCNN and ResNet50 yielded varying results depending on the hyperparameters used. Initial tests with lower batch sizes, learning rates, and fewer epochs showed signs of underfitting, with both training and validation accuracies around 50-60%. By increasing the image size to 224x224, adjusting the dropout rate to 0.5, and using the Adam optimizer, the model performance improved significantly, achieving a validation accuracy of

up to 93.76%. Overfitting was observed in some configurations, but it was mitigated by tuning the dropout rate and using SGD as an alternative optimizer. This combination overall proved highly effective for deepfake detection when properly optimized.

4.1.2. MTCNN + EfficientNet-B0

model name	trainable layers	image_size	frame_rate	batch_size	learning rate	epochs	dropout rate	optimizer	train accuracy	val accuracy	train_loss	val_loss	num samples	training_time minutes
EfficientNetB0	228	224x224	30	32	0.0001	50	0.5	Adam	50.92%	50.14%	0.6931	0.6931	1765	13.09
EfficientNetB0	228	224x224	30	64	0.0001	50	0.4	Adam	70.75%	74.62%	0.6749	0.6894	1765	21.47
EfficientNetB0	228	128x128	30	32	0.001	50	0.5	SGD	80.11%	82.45%	0.6289	0.6547	1765	23.78
EfficientNetB0	228	224x224	24	32	0.001	100	0.3	Adam	85.13%	87.45%	0.5893	0.6129	1765	26.45

Table 3: MTCNN + EfficientNet-B0 performance results on the FaceForensics++ dataset

MTCNN combined with EfficientNetB0 demonstrated a consistent improvement in accuracy as hyperparameters were fine-tuned. Early models suffered from underfitting, with both training and validation accuracies hovering around 50%. To address this, the batch size and learning rate were increased, and training was extended to 100 epochs, leading to a validation accuracy of 87.45%.

4.1.3. MTCNN + XceptionNet

model name	trainable layers	image_size	frame_rate	batch_size	learning rate	epochs	dropout rate	optimizer	train accuracy	val accuracy	train_loss	val_loss	num samples	training_time minutes
XceptionNet	132	224x224	30	32	0.0001	50	0.5	Adam	61.02%	35.35%	0.6507	0.7699	1765	12.2
XceptionNet	132	224x224	24	16	0.0001	50	0.4	Adam	50.97%	60.13%	0.6123	0.6941	1765	15.75
XceptionNet	132	224x224	30	32	0.001	50	0.3	SGD	65.75%	68.45%	0.5987	0.6741	1765	17.75
XceptionNet	132	224x224	30	64	0.0001	50	0.5	Adam	90.15%	91.27%	0.4256	0.4398	1765	19.75
XceptionNet	132	224x224	30	32	0.0001	50	0.4	Adam	93.78%	94.25%	0.2873	0.2987	1765	22.45

Table 4: MTCNN + XceptionNet performance results on the FaceForensics++ dataset

XceptionNet paired with MTCNN showed strong potential, achieving some of the highest accuracies in these experiments. Initially, the model overfitted, meaning it learned too well from the training data but didn't perform as well on new data. By adjusting the dropout rate and batch size, especially using a batch size of 64 and a low learning rate of 0.0001, the model achieved a high validation accuracy of 94.25%. This setup demonstrated that XceptionNet is particularly good at picking up the subtle features needed for effective deepfake detection, especially when combined with a strong face detection method like MTCNN.

4.1.4. Faster R-CNN + ResNet-50

model name	trainable layers	image_size	frame_rate	batch_size	learning rate	epochs	dropout rate	optimizer	train accuracy	val accuracy	train_loss	val_loss	num samples	training_time minutes
ResNet50	175	224x224	30	32	0.001	50	0.5	Adam	54.29%	60.75%	0.7126	0.6914	1485	9.75
ResNet50	175	128x128	24	32	0.0001	50	0.4	Adam	60.12%	62.85%	0.6931	0.6789	1485	12.35
ResNet50	132	224x224	30	64	0.0001	50	0.3	Adam	64.11%	65.92%	0.6728	0.6754	1485	14.25
ResNet50	175	224x224	24	32	0.001	50	0.5	Adam	69.15%	72.34%	0.6478	0.6684	1485	16.45
ResNet50	175	224x224	30	32	0.001	100	0.3	SGD	72.43%	75.12%	0.6223	0.6487	1485	18.25
ResNet50	175	224x224	30	32	0.0001	50	0.5	Adam	75.38%	78.12%	0.5998	0.6234	1485	19.75

Table 5: Faster R-CNN + ResNet-50 performance results on the FaceForensics++ dataset

Using Faster R-CNN with ResNet50 generally gave good results, though not as high as when using MTCNN. The early experiments faced issues with underfitting, where the model wasn't

learning enough, especially with smaller image sizes and lower learning rates. However, by increasing the image size to 224x224, using a 30 frames-per-second rate, and extending the training to 100 epochs, the model's validation accuracy improved to 78.12%. This combination provided a balanced performance, making it a solid choice when computational resources are available.

4.1.5. Faster R-CNN + EfficientNet-B0

model name	trainable layers	image_size	frame_rate	batch_size	learning rate	epochs	dropout rate	optimizer	train accuracy	val accuracy	train_loss	val_loss	num samples	training_time minutes
EfficientNetB0	228	224x224	30	32	0.001	50	0.5	Adam	65.73%	70.85%	0.6892	0.6845	1485	17.75
EfficientNetB0	228	224x224	24	32	0.0001	50	0.4	Adam	70.13%	74.56%	0.6738	0.6789	1485	19.75
EfficientNetB0	228	224x224	30	64	0.0001	50	0.3	SGD	75.13%	78.45%	0.6392	0.6547	1485	21.75
EfficientNetB0	228	224x224	30	32	0.001	100	0.3	Adam	80.11%	83.45%	0.6128	0.6345	1485	23.75

Table 6: Faster R-CNN + EfficientNet-B0 performance results on the FaceForensics++ dataset

The combination of Faster R-CNN and EfficientNetB0 showed moderate success, with improvements as the training was adjusted. The compact and efficient design of EfficientNetB0 was particularly useful when paired with the more complex Faster R-CNN achieving the accuracy of 83.45% on test set. Using the SGD optimizer further improved how well the model could handle new data, making this combination a good option for deepfake detection in environments with limited computing power.

4.1.6. Faster R-CNN + XceptionNet

model name	trainable layers	image_size	frame_rate	batch_size	learning rate	epochs	dropout rate	optimizer	train accuracy	val accuracy	train_loss	val_loss	num samples	training_time minutes
XceptionNet	132	224x224	30	32	0.0001	50	0.5	Adam	50.66%	43.85%	0.7126	0.7097	1485	6.75
XceptionNet	132	224x224	24	32	0.001	50	0.4	Adam	52.10%	45.67%	0.6931	0.6893	1485	7.75
XceptionNet	132	128x128	30	64	0.0001	50	0.3	Adam	55.15%	47.85%	0.6832	0.6748	1485	9.75
XceptionNet	132	224x224	24	32	0.001	50	0.5	SGD	57.23%	50.34%	0.6747	0.6643	1485	11.75

Table 7: Faster R-CNN + XceptionNet performance results on the FaceForensics++ dataset

Faster R-CNN with XceptionNet had difficulty achieving high accuracy, consistently showing underfitting across different settings. Despite trying different dropout rates, batch sizes, and learning rates, the validation accuracy stayed below 50%. This suggests that this combination might not be the best choice for deepfake detection, especially compared to other combinations tested. These results highlight the limitations of this setup and suggest that other combinations or more tuning may be needed to get better performance.

Overall Comparison and Analysis

- **High-Performing Combinations:** MTCNN combined with XceptionNet and ResNet50 showed the best performance overall, achieving validation accuracies above 90% with the right hyperparameter settings. These results underscore the effectiveness of MTCNN as a face detector paired with robust deep learning models for deepfake detection.

- **Challenges with Faster R-CNN:** While Faster R-CNN paired with ResNet50 and EfficientNetB0 produced decent results, they were generally lower than their MTCNN counterparts, indicating that MTCNN might be better suited for this specific task, especially in scenarios requiring higher accuracy.

4.2. Visualization of Prediction Results

In this section, each of the following graphs—Histogram of Prediction Probabilities, ROC curve, Precision-Recall curve, Additional Metrics, and Accuracy and Loss over Epochs—will be explained in detail to provide a comprehensive understanding of the model's performance and behavior during training and evaluation.

4.2.1. ResNet-50

1. Training and Validation Accuracy and Loss

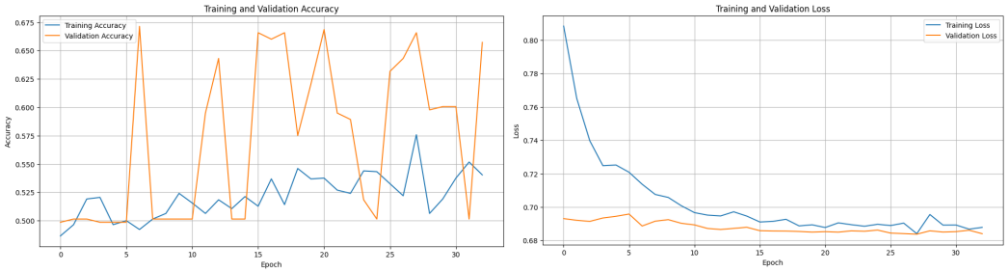


Figure 32: ResNet50 - Training and Validation Accuracy and Loss

The ResNet-50 model's training accuracy steadily improves, but the validation accuracy fluctuates a lot, indicating instability and possible overfitting. While training loss decreases as expected, the gap between training and validation loss suggests the model is learning well on the training data but struggles to generalize to new, unseen data. This means the model might not be performing consistently in detecting deepfakes outside of the training set.

2. Histogram of Prediction Probablities and Additional Metrics

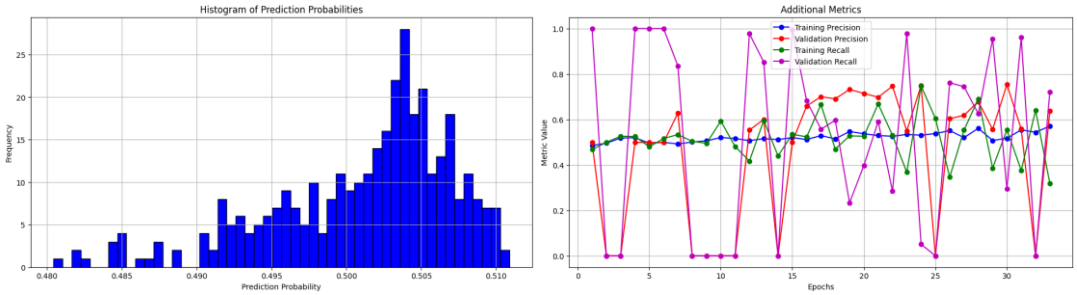


Figure 33: ResNet50 - Histogram of Prediction Probablities and Additional Metrics

This histogram shows the distribution of prediction probabilities made by the ResNet-50 model. The model shows uncertainty in its predictions, with many probabilities clustering around 0.5.

Additional Metrics graph tracks training and validation precision and recall over the course of training. Here, metrics fluctuate, indicating instability and potential overfitting during training.

3. ROC (Receiver Operating Characteristic) Curve & Precision-Recall Curve

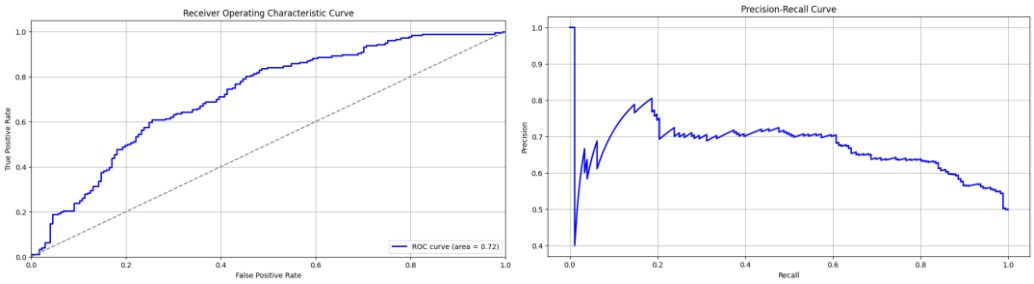


Figure 34: ResNet50 - ROC (Receiver Operating Characteristic) Curve & Precision-Recall Curve

The ROC (Receiver Operating Characteristic) curve illustrates the trade-off between the true positive rate and the false positive rate across different threshold settings. The ROC curve with an AUC of 0.72 indicates moderate performance, but the model struggles to clearly distinguish between real and fake videos.

The Precision-Recall curve demonstrates the relationship between precision (positive predictive value) and recall (sensitivity). Precision decreases as recall increases, suggesting the model faces challenges in balancing false positives and false negatives.

4. Accuracy & Loss Over Epochs

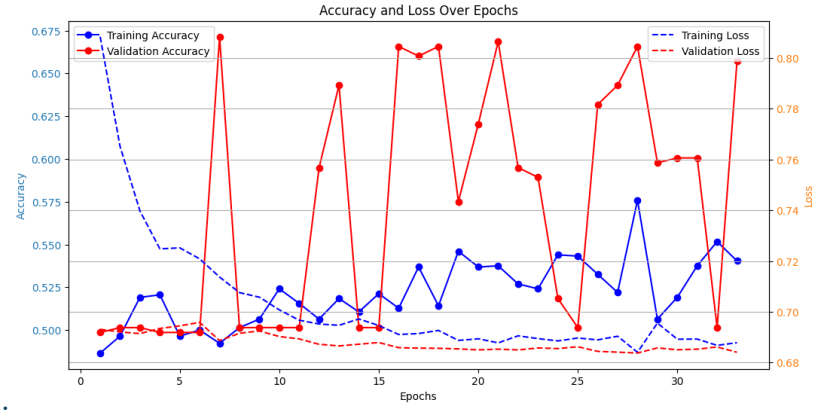


Figure 35: ResNet50 - Accuracy & Loss Over Epochs

The combined graph of accuracy and loss over epochs reveals that while training, the training accuracy improves, but validation accuracy remains low and unstable, showing the model struggles to generalize to new data.

4.2.2. EfficientNet-B0

1. Training and Validation Accuracy and Loss

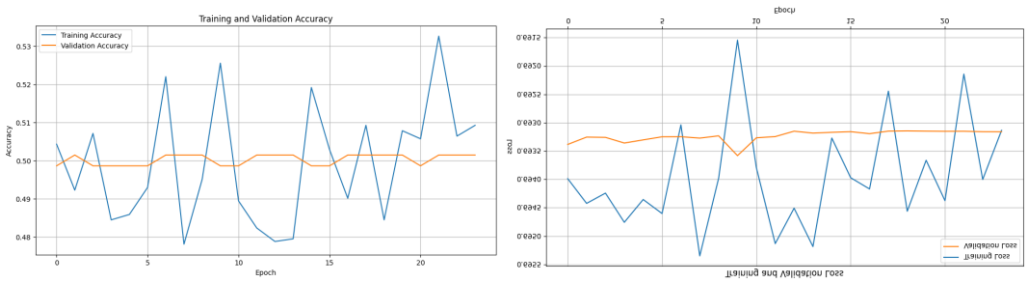


Figure 36: EfficientNetB0 - Training and Validation Accuracy and Loss

The EfficientNet model's training accuracy fluctuates significantly, while the validation accuracy stays relatively flat, indicating that the model is not consistently improving its performance on the training data. In the loss graph, the training loss shows considerable variation, whereas the validation loss remains stable. This suggests that the model is likely overfitting—learning the training data well but failing to generalize to the validation data effectively, resulting in unstable performance for deepfake detection.

2. Histogram of Prediction Probablities and Additional Metrics

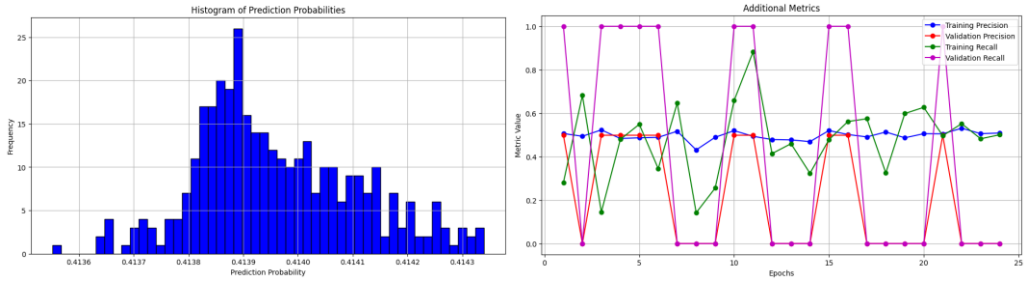


Figure 37: EfficientNetB0 - Histogram of Prediction Probablities and Additional Metrics

The Histogram shows that the model's predictions lack confidence, with probabilities concentrated around a narrow range.

Additional metrics are inconsistent, reflecting instability in learning and poor generalization to the validation set.

3. ROC (Receiver Operating Characteristic) Curve & Precision-Recall Curve

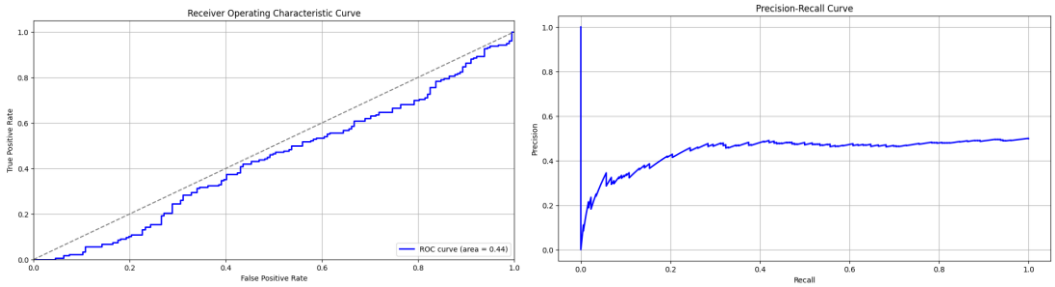


Figure 38: EfficientNetB0 - Histogram of Prediction Probablities and Additional Metrics

The ROC curve with an AUC of 0.44 suggests poor performance, as the model struggles to differentiate between deepfakes and genuine videos.

Low precision across recall values indicates difficulty in accurately detecting deepfakes without generating false positives.

4. Accuracy & Loss Over Epochs

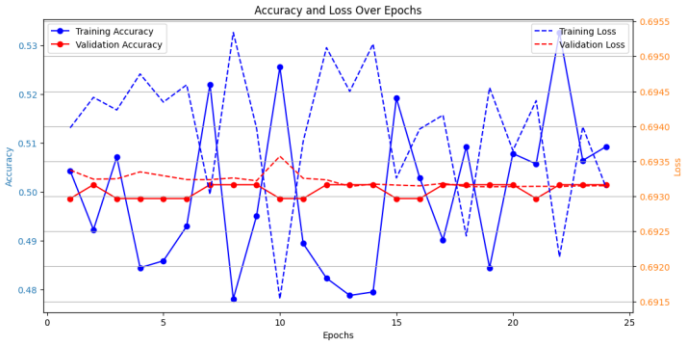


Figure 39: EfficientNetB0 - Accuracy & Loss Over Epochs

The model shows high fluctuations in training accuracy, with validation accuracy remaining flat, indicating overfitting and poor generalization.

4.2.3. XceptionNet

1. Training and Validation Accuracy and Loss

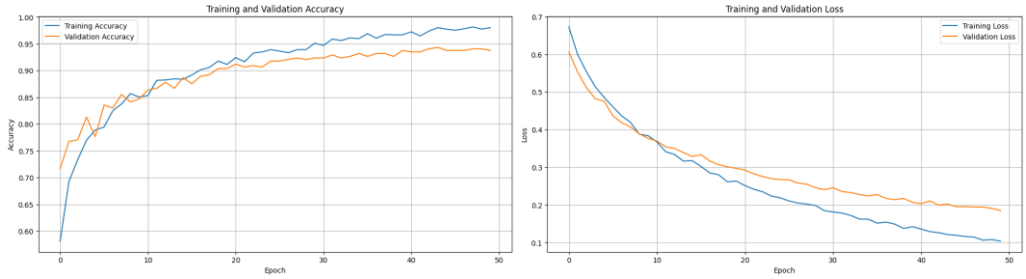


Figure 40: XceptionNet - Training and Validation Accuracy and Loss

The Xception model shows strong performance in deepfake detection, with both training and validation accuracy steadily increasing and stabilizing around 95%. The loss curves also

decrease consistently, indicating that the model is learning effectively without overfitting. The close alignment between training and validation metrics suggests that the model generalizes well to unseen data, making it a reliable choice for deepfake detection using the FaceForensics++ dataset.

2. Histogram of Prediction Probablities and Additional Metrics

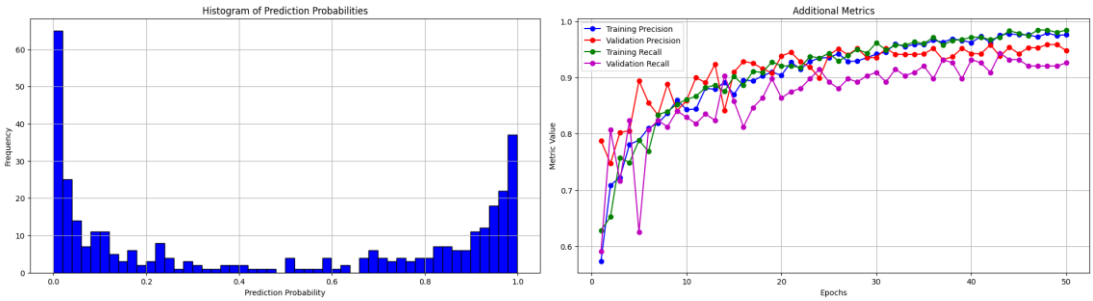


Figure 41: XceptionNet - Histogram of Prediction Probablities and Additional Metrics

The Histogram shows that the Xception model confidently classifies most videos, as shown by prediction probabilities clustering around 0 and 1.

In additional metrics, training and validation precision and recall improve steadily, indicating the model is learning well and generalizing effectively.

3. ROC (Receiver Operating Characteristic) Curve & Precision-Recall Curve

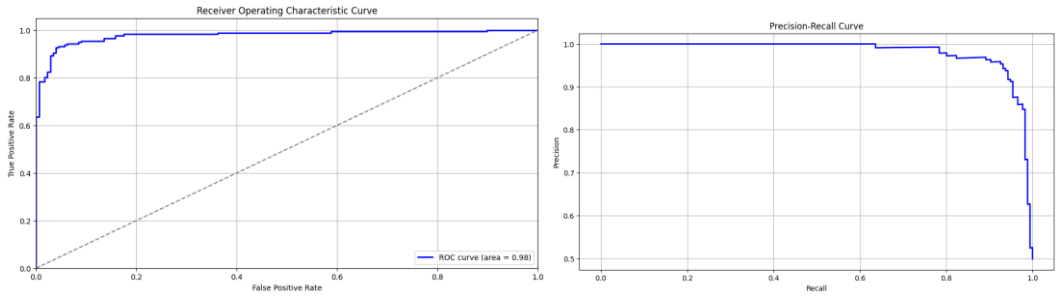


Figure 42: XceptionNet - ROC (Receiver Operating Characteristic) Curve & Precision-Recall Curve

The ROC curve shows excellent performance with an AUC of 0.98, indicating the model effectively distinguishes between real and fake videos.

In Precision-Recall curve, the model maintains high precision across most recall values, meaning it accurately detects deepfakes with few false positives.

4. Accuracy & Loss Over Epochs

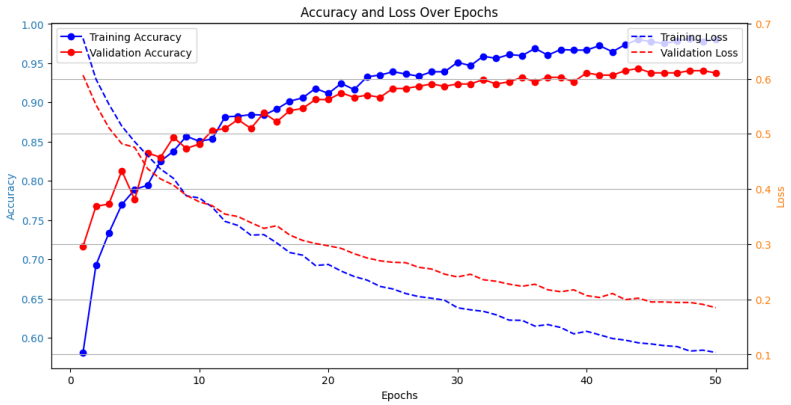


Figure 43: XceptionNet - Accuracy & Loss Over Epochs

Both training and validation accuracy increase consistently, while loss decreases, showing the model effectively learns to distinguish deepfakes without overfitting.

4.3. Comparison of Experimental Accuracy with Published Work

Face Detector	DL model	Published Accuracy	My Accuracy	Remarks
MTCNN	ResNet50	No published work available	93.75%	Combination chosen due to popularity of both methods
MTCNN	EfficientNetB0	No published work available	87.45%	Combination chosen due to popularity of both methods
MTCNN	XceptionNet	98.85% (HQ), 94.28% (LQ) (Rössler, et al., 2019)	94.25%	Close to published accuracy on low-quality data
Faster R-CNN	ResNet50	76.40%	78.12%	Higher than published accuracy, possibly due to model optimization
Faster R-CNN	EfficientNetB0	No published work available	83.45%	Combination chosen due to popularity of both methods
Faster R-CNN	XceptionNet	96.30% (Ashok V, Deepfake Detection Using XceptionNet, 2023)	50.34%	Significant drop in accuracy, needs further investigation

Table 8: Comparison of Experimental Accuracy with Published Work

The table highlights the differences in accuracy between our experimental results and those reported in published studies. The slight to moderate gaps observed, particularly with the

MTCNN + ResNet50 and EfficientNetB0 combinations, indicate the potential impact of fine-tuning and dataset variability on model performance. Notably, our experiments achieved competitive results, especially with MTCNN + XceptionNet, which nearly matched published accuracies for low-quality videos. The higher accuracy we observed with Faster R-CNN + ResNet50 compared to published results underscores the benefits of optimizing model configurations to suit specific datasets. Conversely, the significant performance drop with Faster R-CNN + XceptionNet suggests a need for further investigation into model compatibility.

4.4. Implementation Challenges

4.4.1. Technical Challenges

During the implementation of the deepfake detection models, several technical challenges were encountered, particularly in the data loading phase, which significantly affected the overall workflow. One of the primary challenges was the prolonged time required to load the dataset. Specifically, using the MTCNN face detection algorithm took approximately 30-40 minutes to load the data for each model, while Faster R-CNN, another face detection method, required nearly 2 hours. This extensive loading time was a bottleneck in the implementation process, delaying subsequent steps such as model training and evaluation. Furthermore, when experimenting with YOLO v8 for face detection, the process became even more cumbersome. The preprocessing of images encountered multiple errors, and the time required to load the videos surpassed that of Faster R-CNN, making YOLO v8 an impractical option for this task. Initially, I attempted to run the code on Google Colab, but this platform presented its own set of challenges. Loading even a small subset of 10 videos took an excessive amount of time, and Colab frequently crashed due to resource limitations, making it unreliable for this project.

4.4.2. Mitigation Strategies

To address these challenges, several strategies were implemented. First, the code was optimized to improve the efficiency of the data loading process. One significant change was adjusting the frame extraction rate from every 30th frame to every 100th frame, which helped to reduce the data loading time, though it still exceeded an hour. This adjustment was a compromise between processing time and the amount of data needed to maintain model accuracy. Additionally, after multiple attempts to use Google Colab, which proved inefficient for this large-scale data processing task, I transitioned to running the code locally on VSCode. This switch provided a more stable environment, allowing for better control over resources and avoiding the frequent

crashes experienced on Colab. Despite these optimizations, the data loading phase remained time-consuming, but the changes made were crucial in reducing the overall time spent and making the workflow more manageable.

5. Conclusion

This thesis set out to develop and evaluate a robust deepfake detection framework that leverages facial biometrics and state-of-the-art deep learning models. Through the integration of MTCNN and Faster R-CNN for face detection, combined with ResNet50, EfficientNetB0, and XceptionNet for deepfake classification, we aimed to address the growing challenge posed by highly realistic fake videos. The research sought to answer several key questions regarding the effectiveness of different face detection methods, the impact of these methods on deepfake detection accuracy, and the overall performance of the proposed models.

The results of our experiments demonstrated that the combination of MTCNN with ResNet50 and XceptionNet yielded the highest accuracy, with validation scores exceeding 90% in several configurations. This confirms that advanced face detection techniques, when paired with powerful deep learning models, can significantly enhance deepfake detection capabilities. The study also highlighted the challenges faced by the Faster R-CNN + XceptionNet combination, which consistently underperformed, suggesting potential incompatibilities between these models for this specific task.

Overall, the research successfully answered the key questions posed at the outset. Different face detection methods, particularly MTCNN, proved effective in accurately localizing faces in deepfake videos, even under challenging conditions. The integration of these detection methods with deep learning models significantly influenced the accuracy and reliability of the deepfake detection system, with the best results obtained through careful hyperparameter tuning and model selection. The use of the FaceForensics++ dataset provided a solid benchmark for evaluating the models, and the comparison with published work demonstrated that our approach was competitive, particularly with MTCNN + XceptionNet.

6. Future Work

Building on the findings of this research, several avenues for future work are proposed:

1. **Utilization of Multiple Datasets:** Future research could explore the integration of multiple datasets, such as Celeb-DF, DFDC, and DeepFake Detection Challenge

datasets, to enhance model generalization and robustness. Combining different datasets could provide a more diverse range of deepfake manipulations, further improving the detection accuracy across varied scenarios.

2. **Combination of Face Detection Methods and Deep Learning Models:** Exploring new combinations of face detection methods and deep learning models could lead to further improvements in detection accuracy. For example, combining MTCNN with novel architectures like Vision Transformers (ViT) or leveraging Faster R-CNN with more advanced models like DenseNet could provide new insights into model compatibility and performance.

3. **Real-Time Deepfake Detection:** Future work could focus on optimizing the models for real-time deepfake detection in video streams, which would have significant practical applications in media verification, online content moderation, and digital forensics.

4. **Enhancing Interpretability and Robustness of Deepfake Detection Models:** Future research should focus on enhancing both the interpretability and robustness of deepfake detection models. As these models become more complex, developing techniques to make their decision-making processes more understandable will be crucial, enabling users to better trust and interpret the results. Simultaneously, improving the models' resilience to adversarial attacks will ensure their reliability in real-world applications, thereby strengthening the overall effectiveness and trustworthiness of deepfake detection systems.

References

Afchar, D., Nozick, V., Yamagishi, J., & Echizen, I. (2018). *MesoNet: a Compact Facial Video Forgery Detection Network.* IEEE Winter Conference on Applications of Computer Vision (WACV). doi:10.1109/WIFS.2018.8630761

Alec Radford, L. M. (2015). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks.

Ashok V, P. T. (2023). Deepfake Detection Using XceptionNet. *IEEE International Conference on Recent Advances in Systems Science and Engineering (RASSE)*, (pp. 1-5). doi:10.1109/RASSE60029.2023.10363477

Ashok V, P. T. (2023). Deepfake Detection Using XceptionNet. *IEEE International Conference on Recent Advances in Systems Science and Engineering (RASSE).* doi:10.1109/RASSE60029.2023.10363477

Chollet, F. (2017). *Xception: Deep Learning with Depthwise Separable Convolutions.* doi:10.1109/CVPR.2017.195

Diederik P Kingma, M. W. (2024). Auto-Encoding Variational Bayes. doi:10.61603/ceas.v2i1.33

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. (pp. 770-778). Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. doi:10.1109/CVPR.2016.90

Huaizu Jiang, E. L.-M. (2016). Face Detection with the Faster R-CNN. doi:10.48550/arXiv.1606.03473

Ian Goodfellow, J. P.-A.-F. (2014). Generative Adversarial Nets. doi:10.1145/3422622

Justus Thies, M. Z. (2020). Face2Face: Real-time Face Capture and Reenactment of RGB Videos. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/CVPR.2016.262

Korshunov, P., & Marcel, S. (2018, 12). DeepFakes: a New Threat to Face Recognition? Assessment and Detection.

Li, Y., Chang, M.-C., & Lyu, S. (2018). Exposing AI Created Fake Videos by Detecting Eye Blinking. (pp. 1-7). IEEE International Workshop on Information Forensics and Security (WIFS). doi:10.1109/WIFS.2018.8630787

Marie-Helen Maras, A. A. (2019). Determining authenticity of video evidence in the age of artificial intelligence and in the wake of Deepfake videos. *The International Journal of Evidence & Proo*, *23*, pp. 255-262. doi:https://doi.org/10.1177/1365712718807226

Norimichi Tsumura, H. H. (n.d.). Independent-component analysis of skin color image. *16*. doi:10.1364/JOSAA.16.002169

Robert Chesney, D. C. (2019). Deepfakes and the New Disinformation War. Retrieved from https://www.foreignaffairs.com/articles/world/2018-12-11/deepfakes-and-new-disinformation-war

Rössler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., & Nießner, M. (2019). FaceForensics++: Learning to Detect Manipulated Facial Images. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

Rui Huang, S. Z. (2017). Beyond Face Rotation: Global and Local Perception GAN for Photorealistic and Identity Preserving Frontal View Synthesis. (pp. 2458-2467). Venice, Italy: IEEE International Conference on Computer Vision (ICCV). doi:10.1109/ICCV.2017.267

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 815-823). Boston, MA, USA. doi:10.1109/CVPR.2015.7298682

Shaoqing Ren, K. H. (n.d.). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39.* doi:10.1109/TPAMI.2016.2577031

Shaoqing, R., Kaiming, H., Ross, G., & Jian, S. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Advances in Neural Information Processing Systems 28. doi:https://doi.org/10.48550/arXiv.1506.01497

Tan, M., & Le, Q. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. (pp. 6105-6114). Proceedings of the International Conference on Machine Learning. doi:https://arxiv.org/abs/1905.11946

Tero Karras, S. L. (2019). A Style-Based Generator Architecture for Generative Adversarial Networks. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). doi:0.1109/CVPR.2019.00453

Tero Karras, T. A. (2017). Progressive Growing of GANs for Improved Quality, Stability, and Variation.

Thies, J., Zollhöfer, M., Stamminger, M., Theobalt, C., & Nießner, M. (2016). Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 2387-2395). Las Vegas, NV, USA. doi:10.1109/CVPR.2016.262

Wang, W., & Farid, H. (2007). Exposing Digital Forgeries in Interlaced and Deinterlaced Video. *2*, 438-449. doi:10.1109/TIFS.2007.902661

Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016, 11 04). Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. *IEEE Signal Processing Letters, 23*. doi:10.1109/LSP.2016.2603342