Cross-Dataset Deep Fake Detection

José Pedro da Costa Marques

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática
(2º ciclo de estudos)

Orientador: Prof. Doutor João Carlos Raposo Neves

Outubro de 2024

Declaração de Integridade

Eu, José Pedro da Costa Marques, que abaixo assino, estudante com o número de inscrição M12829 de/o Engenharia Informática da Faculdade Universidade da Beira Interior, declaro ter desenvolvido o presente trabalho e elaborado o presente texto em total consonância com o Código de Integridades da Universidade da Beira Interior.

Mais concretamente afirmo não ter incorrido em qualquer das variedades de Fraude Académica, e que aqui declaro conhecer, que em particular atendi à exigida referenciação de frases, extratos, imagens e outras formas de trabalho intelectual, e assumindo assim na íntegra as responsabilidades da autoria.

Universidade da Beira Interior, Covilhã 05/10/2024

Assinado por: **JOSÉ PEDRO DA COSTA MARQUES**
Num. de Identificação: 15898750
Data: 2024.10.10 09:59:46+01'00'

Resumo

Esta dissertação apresenta uma investigação da tecnologia deepfake, nas técnicas generativas e no desafio crítico de detetar tais falsificações. O estudo aborda as principais técnicas da criação de deepfakes, destacando o uso de Redes Gerativas Adversariais (GANs), Modelos de Difusão e Autoencoders. Para além dos processos generativos, são examinados vários algoritmos de deteção, sublinhando a importância da análise de características específicas e dos detetores baseados em aprendizagem profunda para identificar falsificações sofisticadas. Nesta dissertação, propõe-se uma abordagem inovadora que integra a análise avançada de características faciais usando características espaciais, de frequência e de pontos faciais, com Gated Multimodal Units para fusão de características em conjunto com um mecanismo de atenção tripla e ainda utiliza transformadores para explorar dependências temporais. Estas metodologias são concebidas para melhorar a deteção de deepfakes, enfrentando a crescente sofisticação dessas tecnologias e a sua capacidade de escapar aos detetores tradicionais. As contribuições desta dissertação abrangem tanto o entendimento académico como as aplicações práticas, visando melhorar a robustez dos mecanismos de deteção de deepfakes e contribuir para um ambiente digital mais seguro.

Palavras-chave

Deteção de Deepfakes, Redes Adversárias Generativas (GANs), Modelos de Difusão, Autoencoders, Análise de Características Faciais, Algoritmos de Aprendizagem Automática, Estratégias de Deteção Multimodal, Deteção de Falsificação de Vídeo.

Resumo alargado

Esta dissertação apresenta uma análise detalhada da tecnologia deepfake, nas técnicas generativas utilizadas para criar falsificações digitais altamente convincentes e no desafio crescente de detetar estas manipulações. A tecnologia deepfake evoluiu rapidamente, impulsionada pelo desenvolvimento de modelos de inteligência artificial avançados, como as Redes Gerativas Adversariais (GANs), Modelos de Difusão e Autoencoders, que são capazes de gerar conteúdos visuais de alta qualidade, quase indistinguíveis dos originais. O trabalho explora profundamente estas tecnologias, oferecendo uma visão abrangente sobre o seu funcionamento e como são utilizados para fabricar vídeos e imagens falsificadas. A dissertação aborda a problemática da deteção, um desafio crítico face à crescente sofisticação destas falsificações. A pesquisa destaca a importância de técnicas de análise específicas, como a avaliação de artefactos visuais e discrepâncias em características faciais, e do uso de detetores baseados em aprendizagem profunda, como redes neurais convolucionais, para identificar as subtis inconsistências que caracterizam os deepfakes. A combinação de métodos tradicionais e técnicas de aprendizagem profunda procura melhorar a precisão e eficácia na identificação de falsificações. Nesta dissertação é proposto de um novo método de deteção, que combina a análise de características faciais avançadas com o uso de características espaciais, de frequência e de pontos faciais (landmarks). Esta abordagem utiliza Gated Multimodal Units para a fusão de características provenientes de diferentes domínios em conjunto com um mecanismo de atenção tripla para melhorar a seleção das características mais relevantes e o uso de transformadores para captar as dependências temporais ao longo dos frames dos vídeos analisados. Este método pretende aumentar a robustez e precisão dos sistemas de deteção de deepfakes, especialmente em cenários onde as técnicas tradicionais falham devido à elevada qualidade das falsificações modernas. A dissertação segue uma estrutura bem definida, começando com uma revisão exaustiva da literatura existente, passando pela recolha e análise de dados, desenvolvimento do modelo de deteção e, finalmente, testes rigorosos para validar a abordagem proposta. Esta abordagem metodológica garante uma análise completa, desde o contexto teórico até à aplicação prática dos mecanismos de deteção. As contribuições deste trabalho são significativas tanto no plano académico, ao aprofundar o entendimento sobre a criação e deteção de deepfakes, como no campo prático, oferecendo soluções para melhorar a segurança digital num mundo cada vez mais ameaçado pela manipulação de conteúdos visuais.

Abstract

This thesis presents an in-depth study of deepfake technology, focusing on generative techniques and the critical challenge of detecting such forgeries. The study delves into the technical foundations of deepfake creation, emphasizing the use of Generative Adversarial Networks (GANs), Diffusion Models, and Autoencoders. Alongside the generative processes, this research examines a variety of detection algorithms, stressing the importance of specific feature analysis and deep learning detectors in identifying sophisticated forgeries. A novel approach is proposed in this thesis, incorporating advanced facial feature analysis using spatial, frequency, and temporal landmark features, with Gated Multimodal Units for feature fusion, allied to a triplet attention mechanism, and transformers to leverage temporal dependencies. These methodologies are designed to improve the detection of deepfakes, addressing the growing sophistication of such technologies and their ability to bypass traditional detectors. The contributions of this thesis extend both to academic understanding and practical applications in enhancing the robustness of deepfake detection mechanisms, ultimately contributing to a safer digital environment.

Keywords

Deepfake Technology, Deepfake Detection, Generative Adversarial Networks (GANs), Diffusion Models, Autoencoders, Facial Feature Analysis, Machine Learning Algorithms, Multimodal Detection Strategies, Video Forgery Detection.

Contents

List of Figures

List of Tables

List of Acronyms

GANs	Generative Adversarial Networks
NCSN	Noise-Conditioned Score Networks
DDPM	Denoising Diffusion Probabilistic Models
ML	Machine Learning
CNN	Convolutional Neural Network
VAEs	Variational Autoencoders
TDNN	Time-Delay Neural Network
DNNs	Deep Neural Networks
LSTM	Long Short-Term Memory
SVM	Support Vector Machine
AI	Artificial Intelligence
RELU	rectified linear unit
DCT	Discrete Cosine Transform
I-DCT	Inverse Discrete Cosine Transform
MHA	Multi-Head Attention
CMT	Cross-Modal Transformer
SFF	Spatial-Frequency-Fusion Module
LGL	Landmark Graph Learning
GAT	Graph Attention Layer
MLP	Multi-Layer Perceptron
GNN	Graph Neural Network
FF++	FaceForensics++
DF-VAE	DeepFake Variational Auto-Encoder
DF	DeepFake
MTCNN	Multi-task Cascaded Convolutional Networks
FFT	Fast Fourier Transform
HOG	Histogram of Oriented Gradients
ROI	Region of interest
TCN	Temporal convolution networks
GMU	Gated Multimodal Units
GRU	Gated Recurrent Units

Chapter 1

Introduction

In this era of technology and connectivity online media has evolved to include various forms like communication entertainment and information distribution transforming the way we interact with digital content, however as we embrace these advancements we also confront a new danger in the form of deepfakes which seem realistic but are fake, generated using advanced artificial intelligence and machine learning technologies. These deceptions present substantial obstacles to verifying the truthfulness and reliability of digital information impacting critical areas such, as politics journalism and individual safety. As deepfake technology advances and becomes widely available to the public, the distinction between what is real and what is fake becomes increasingly indistinct, stressing the importance of developing effective detection methods. This research paper explores the realm of deepfake detection with a focus on overcoming the challenge of identifying these fabrications across various types of data. While current detection techniques can detect deepfakes within known datasets, they encounter difficulties, in generalizing their performance when encountering manipulated content created using methods or from diverse sources. This thesis seeks to address this gap by putting forward a stronger and widely applicable method, for identifying deepfakes.

1.1 Motivation and Scope

Deepfake technology's advanced capabilities allow for the alteration of videos and images that can have profound impacts on fields such as politics and media, while also threatening personal privacy. This demonstrates the requirement for reliable detection methods to combat deepfake threats and protect the integrity of digital materials, from deceptive alterations. The main goal of this research is to create a detection approach that is both precise and resilient, against different types of manipulations.

1.2 Novel Contributions

This thesis introduces two novel components aimed at enhancing the robustness and accuracy of deepfake detection methods:

- **Gated Multimodal Units (GMUs)**: Inspired by the concept of Gated Recurrent Units (GRUs), GMUs are designed to dynamically integrate information from multiple modalities (e.g., spatial, frequency, and landmark features). By adaptively weighing the importance of each modality, GMUs enable a more robust and context-aware fusion of features, improving the overall detection accuracy.
- **Triplet Attention Mechanism**: This mechanism extends traditional attention by simultaneously considering the height, width, and channel dimensions of feature maps. By

capturing interdependencies across these dimensions, Triplet Attention enhances the model's ability to detect subtle inconsistencies in deepfake content, thereby improving the precision and reliability of the detection process.

1.3 Problem Statement and Objectives

The main issue discussed in this research is the challenge of adapting current deepfake detection techniques to datasets effectively and reliably across various scenarios. With deepfake technology advancing quickly, it has become harder for existing models to keep up with methods of generating deepfake content or data from unfamiliar sources. This difficulty restricts their use in real life situations where deepfake content is created using diverse datasets and techniques. To address this issue, this research aims to achieve the following goals;

1. **Analyze Existing Detection Methods:** Conduct a thorough analysis of state-of-the-art deepfake detection methods, identifying their strengths and limitations, particularly in cross-dataset applications.

2. **Innovate Detection Strategies:** Design and develop detection strategies utilizing GMUs and Triplet Attention to enhance the accuracy and robustness of deepfake detection across diverse datasets and manipulation techniques.

3. **Evaluate and Validate the Proposed Methods:** Rigorously test and validate the proposed method using FF++[14] and CelebDF [15] datasets to ensure their effectiveness and reliability in various scenarios.

4. **Contribute to the Field of Digital Media Authenticity:** Provide valuable insights and advancements in deepfake detection, contributing to the effort of maintaining the integrity and authenticity of digital media.

Chapter 2

State of the Art

2.1 Generative Techniques for Deep Fake Creation

Innovative methods used to create deep fakes, represent a major step forward in technology. This segment explores the realm of algorithms and frameworks that make it possible to produce authentic and compelling digital material. Leading the way in these breakthroughs are GANs (Generative Adversarial Networks) which have become a tool, in generating visuals that push the limits of what we perceive as real. Besides, GANs and other techniques contribute to the world of deepfake creation, each, with its distinctive role in shaping the potential and direction of digital media alteration.

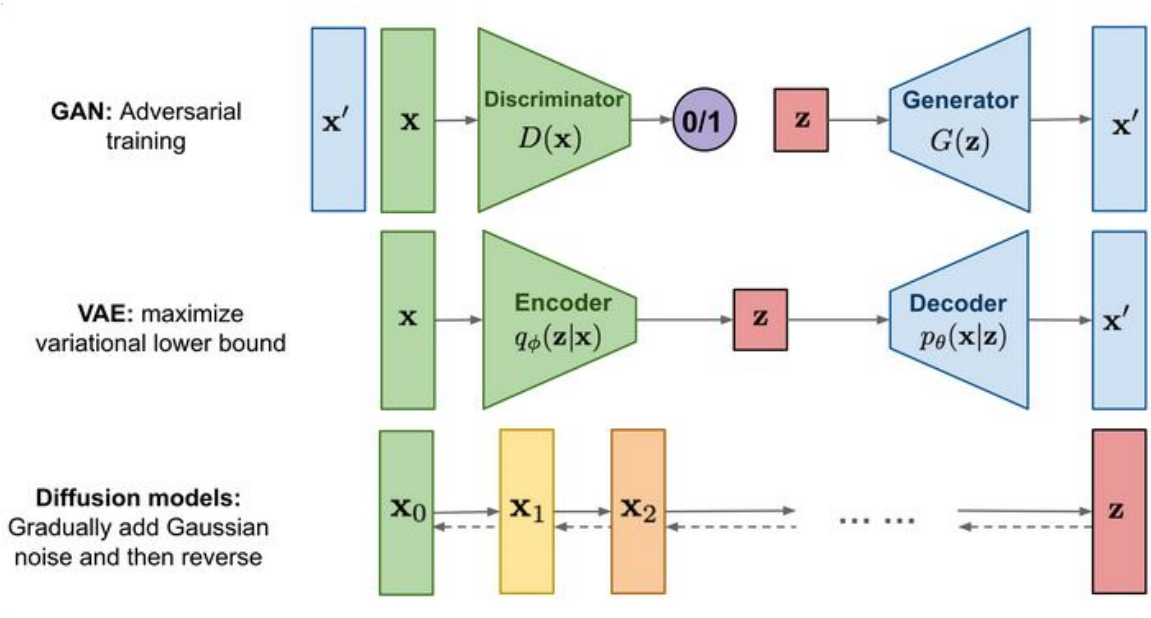


Figure 2.1: Overview of different types of generative models. Adapted from [1].

2.1.1 Generative Adversarial Networks

GAN technology has revolutionized the technology of artificial intelligence (AI) by introducing groundbreaking methods, for creating content effectively and creatively. It functions through a system that consists of two networks. The generator and the discriminator. Which work together in a competitive yet cooperative manner.

2.1.1.1 The Fundamentals of GAN Technology

GAN technology operates through a constant back and forth, between two networks engaged in an adversarial process. One network generates data while the other network evaluates its

authenticity to improve the overall creativity of the system by encouraging the generator to produce more realistic content. The success of a GAN model depends on how it challenges the discriminator to differentiate between generated and real. The special ability of GAN models is what makes it so powerful for producing deepfake content. They can create images that closely resemble real ones, with great accuracy. [2].

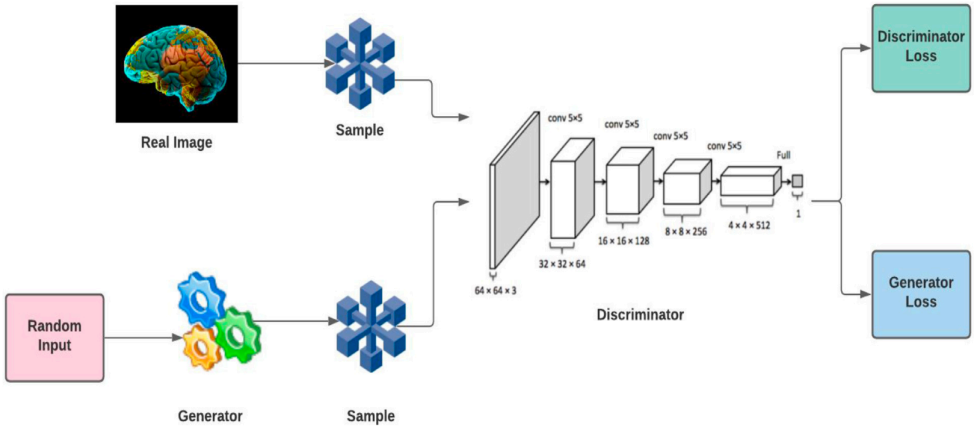


Figure 2.2: Block diagram of the GAN. Adapted from [2].

2.1.2 Diffusion Models

Diffusion models are a breakthrough in creating high quality digital content. These models work by transforming noise into organized data bit, by bit, to mimic the process of diffusion. This involves adding noise and then refining it iteratively to produce detailed and realistic images. The flexibility and effectiveness of diffusion models, in generating a variety of data, make them a significant advancement, in the fields of intelligence and digital media creation.

2.1.2.1 Theoretical Fundamentals of Diffusion Models

Diffusion models are a subclass of generative models characterized by their utilization of a Markov chain mechanism to incrementally infuse Gaussian noise into data. This process produces progressively noisier iterations, which the model subsequently learns to reverse, reconstructing the initial data set. These models encapsulate various iterations, including diffusion probabilistic models, noise-conditioned score networks (NCSN), and denoising diffusion probabilistic models (DDPM). The training regimen for these models predominantly concentrates on the optimization of the negative log-likelihood, employing variational lower bound strategies [1].

Application of Diffusion Models in Deepfake Generation

Distinct from GANs, which depend on the adversarial dynamics between a generator and a discriminator, diffusion models adopt a cyclical procedure of noise addition and subtraction from data, this method that has proven to be efficient in the creation of high-quality and life-like images and videos, thereby establishing these models as a formidable tool in

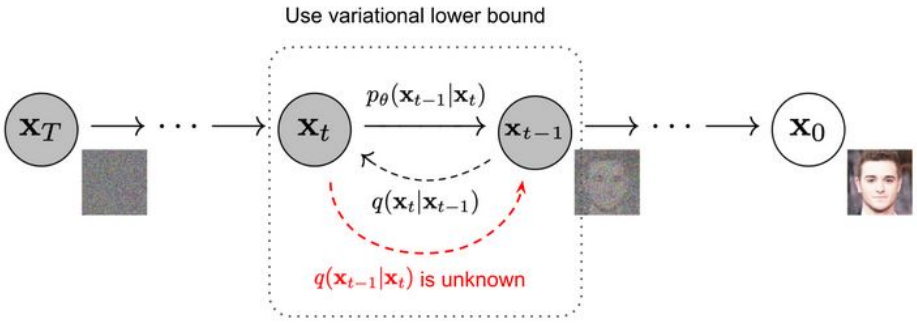


Figure 2.3: The Markov chain of forward (reverse) diffusion process of generating a sample by slowly adding (removing) noise. Adapted from [1].

deepfake creation. An advantage of diffusion models over GANs in the context of deepfake fabrication lies in their capacity to produce more stable and coherent outputs, GANs are occasionally prone to generating artefacts or inconsistencies due to their adversarial training paradigm. Diffusion models offer a more methodical and gradual data generation approach, which results in minimized artefacts and enhanced realism. Furthermore, these models are less susceptible to mode collapse, a common pitfall in GANs where the generator's output becomes monotonously uniform.

2.1.3 Autoencoders

Autoencoders have become indispensable to the domain of deepfake technology, because of their ability to accurately capture and reconstruct facial features. These neural network architectures are used for both compressing data (encoding) and reconstructing it (decoding), whether to its original form or a modified version. This duality makes autoencoders particularly suited for deepfake creation.

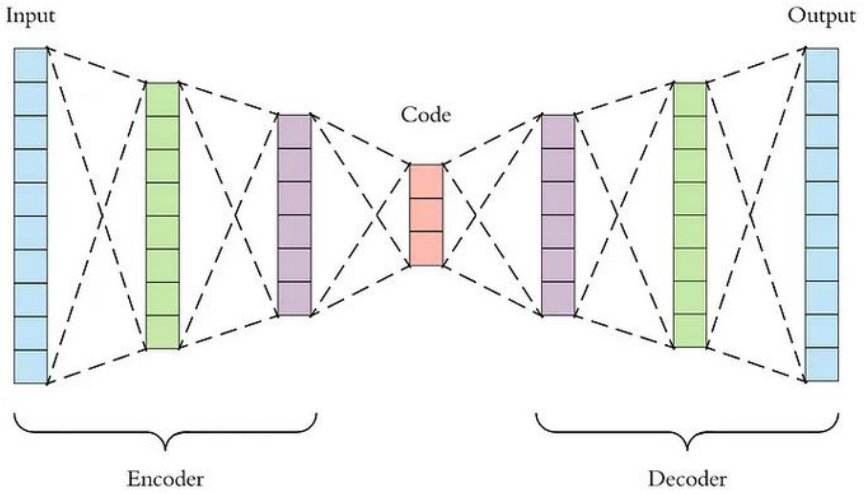


Figure 2.4: Encoder architecture. Adapted from [3].

Theoretical Foundations of Autoencoders

At the core of autoencoders are two main components: the encoder and the decoder. The

encoder is responsible for transforming the input data into a lower-dimensional space, known as the latent space or bottleneck. This process involves distilling the data to its most essential features, eliminating redundancies and noise. The decoder takes this compressed representation and reconstructs it back to its original dimensionality, aiming to approximate the original input as closely as possible. The efficiency of an autoencoder is measured by its ability to minimize the reconstruction error, which is evaluated using specific loss functions. The training of these networks involves fine-tuning the weights through backpropagation and optimization algorithms, often employing stochastic gradient descent. This approach, often characterized as self-supervised learning, allows autoencoders to adapt to specific data characteristics without needing labelled datasets. Autoencoders are designed to be adaptable, with variations like variational autoencoders (VAEs) and convolutional autoencoders customized for different data types and objectives, such as generating new, yet similar, instances of the training data. In the context of deepfake technology, autoencoders are valued for their ability to accurately capture and reconstruct facial imagery. They train on a dataset of facial images to encode significant facial characteristics and can decode this information to reconstruct the same face or superimpose it onto another. This capability supports the creation of hyperrealistic deepfakes, where facial features are altered or replaced flawlessly in video or image content.

2.1.3.1 The Role of Autoencoders in Deepfake Generation

Autoencoders have significantly influenced the way facial images are manipulated and transformed for synthetic media creation [4]. When training autoencoders, for deepfake projects, it's crucial to ensure that the autoencoders trained on datasets can work together. One way to achieve this is by using a shared encoder for datasets and having decoders for each specific dataset. The shared encoder is able to identify features, among different individuals, allowing these features to be applied in various reconstruction scenarios. Once the training is completed, the shared encoders representation of a person's face can be decoded using another person's decoder. This method enables the creation of deepfake videos by transferring the expressions and poses of one person onto another individual's face, successfully relies heavily upon the encoder's capacity to apply facial characteristics across various faces.

2.2 Deepfake Detection algorithms

2.2.1 Specific Feature Analysis in Deepfake Detection

The domain of deepfake detection comprises various methodologies, with specific feature analysis being a prominent category. These approaches entail the meticulous extraction and analysis of distinct features, including visual artefacts, temporal, and frequency information, to identify inconsistencies or abnormalities indicative of deepfake generation. The authors of this paper [16] conducted a detailed investigation into the discrepancies between visual lip movements and corresponding audio speeches in deepfakes, highlighting a crucial aspect of audiovisual incongruity. Matern et al. [17] undertook an in-depth exploration of the reflective properties within the eyes, teeth, and facial contours in deepfakes. Their work sheds light on

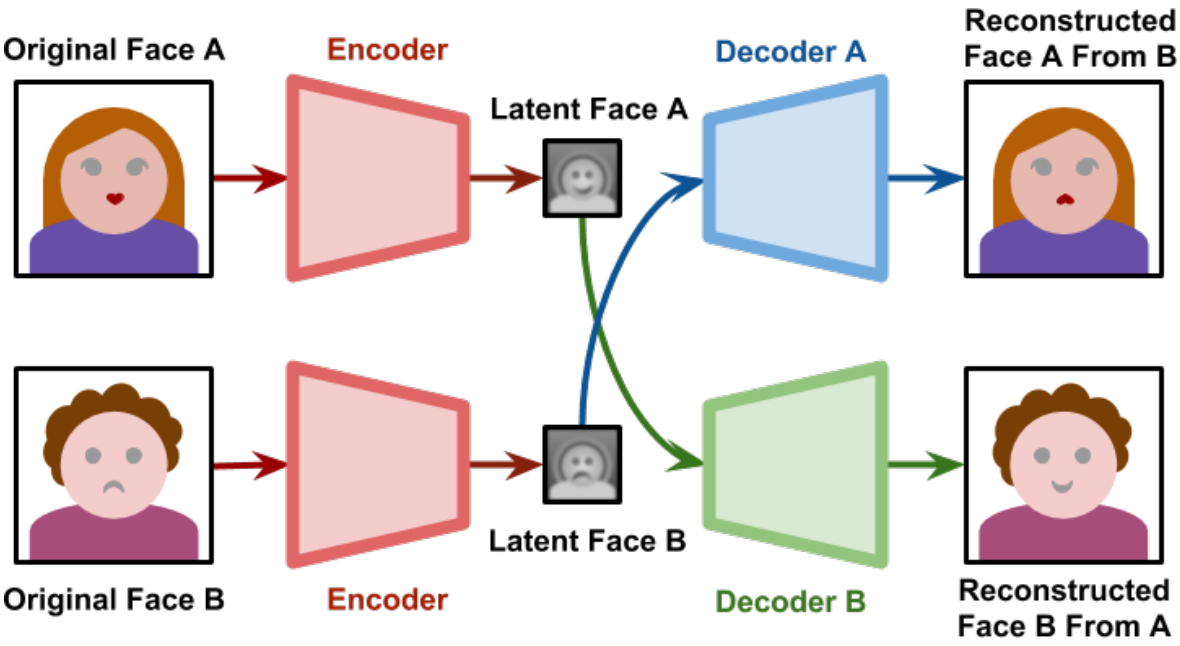


Figure 2.5: the Decoder B will try to reconstruct Subject B, from the information relative to Subject A. Adapted from [4].

the nuances of facial reflections as a potential indicator of falsified content. Further advancing this field, Yang et al. [5] focused on the comprehensive extraction of facial landmarks. Their work was instrumental in estimating 3D head poses and detecting anomalies in pose data, which could signify deepfake manipulation. In a similar frame of mind, Li et al. [18] observed the face warping processes prevalent in deepfake creation. They adeptly extracted and analysed visual artefacts present in facial regions and their boundaries, thereby contributing significantly to the classification of authentic and manipulated content. Agarwal et al. [19] focused in on the extraction of facial expressions and movements, with a specific focus on identity authentication in deepfake detection. Their methodology underscores the importance of individual-specific characteristics in ascertaining the authenticity of digital content. Complementing these approaches, Gu et al. [20] concentrated on the temporal aspect of video frames. Their research aimed to identify both spatial and temporal inconsistencies, which are telltale signs of deepfake videos. Liu et al. [21] integrated spatial-image and phase-spectrum features to detect anomalies, thereby capturing a broad spectrum of artefacts characteristic of deepfake content. However, it is crucial to acknowledge the inherent limitations of these detectors in practical applications. The reliance on specific features renders them susceptible to environmental variables such as lighting conditions or digital processes like image compression. Moreover, the emergence of advanced deepfake generation methods poses a continual challenge to the efficacy of these detection techniques [16].

2.2.2 Deep Learning Detectors

Deep learning detectors represent the second critical category in the realm of deepfake detection. These detectors primarily employ Deep Neural Networks (DNNs), such as CNNs, to differentiate between authentic and deepfake videos. This classification is based on microscopic-

level features, notably pixel-level differences, which are intricate and often imperceptible to the naked eye. Afchar et al. [5] introduced MesoNet, a highly efficient deepfake detection framework, noted for its lightweight network architecture. Building upon this, they further enhanced MesoNet by integrating a variant of the Inception module, resulting in the development of MesoInception [22]. This adaptation significantly augmented the network's capability to discern deepfakes. In another innovative approach, Tariq et al. [23] developed ShallowNet, a method specifically customized for detecting deepfakes generated by GANs. Among its three iterations, ShallowNet-V3 emerged as the most effective, showcasing superior performance in identifying deepfake content. Further contributing to this field, Rossler et al. [24], [25] employed advanced CNNs such as XceptionNet [22] and EfficientNet [18] for deepfake detection. Both of these neural networks demonstrated promising results, underlining their efficacy in the nuanced detection of deepfakes. Compared to specific-feature detectors, deep learning-based techniques offer a more practical and robust approach. By focusing on microscopic-level features, such as pixel differences, deep learning detectors can overcome some of the limitations inherent in visual artifact-based approaches. Their ability to examine content at a granular level allows for a more accurate and reliable classification of videos.

2.3 General Methodologies in Deepfake Detection

2.3.1 Deep Learning Methodologies in MesoNet for Video Forgery Detection

The work presented in the article [5] introduces groundbreaking methodologies utilizing the principles of deep learning to comprehensively address the issue of video forgery. This section provides a thorough and detailed exploration of the advanced approaches employed by the authors. Central to their approach is the development of compact yet efficient neural network architectures, which are carefully designed to capture mesoscopic image features. These features, existing at a crucial intermediate scale that bridges micro-level details (like pixel-based anomalies) and macro-level attributes (such as overall image composition), demonstrate exceptional efficacy in detecting subtle manipulations in compressed video data. The authors of the study have introduced two distinct network architectures, namely Meso-4 and MesoInception-4, each with its unique characteristics and strengths. Meso-4 commences its operations with four layers of convolutional processing, followed by pooling operations. Subsequently, it transitions into a dense network architecture comprising a single hidden layer. The convolutional layers in Meso-4 are equipped with advanced Rectified Linear Unit (RELU) activation functions and incorporate Batch Normalization [26], a critical component that introduces non-linearity and output regularization. This design choice is instrumental in mitigating the vanishing gradient problem, a common challenge in deep learning models. The fully-connected layers further incorporate the Dropout technique [27], a form of regularization that significantly enhances the model's robustness and generalization capabilities. Remarkably, despite its sophisticated functionalities, Meso-4 is characterized by its compactness, containing a total of 27,977 trainable parameters.

The MesoInception-4 architecture emerges as an evolution of the Meso-4 model. It introduces

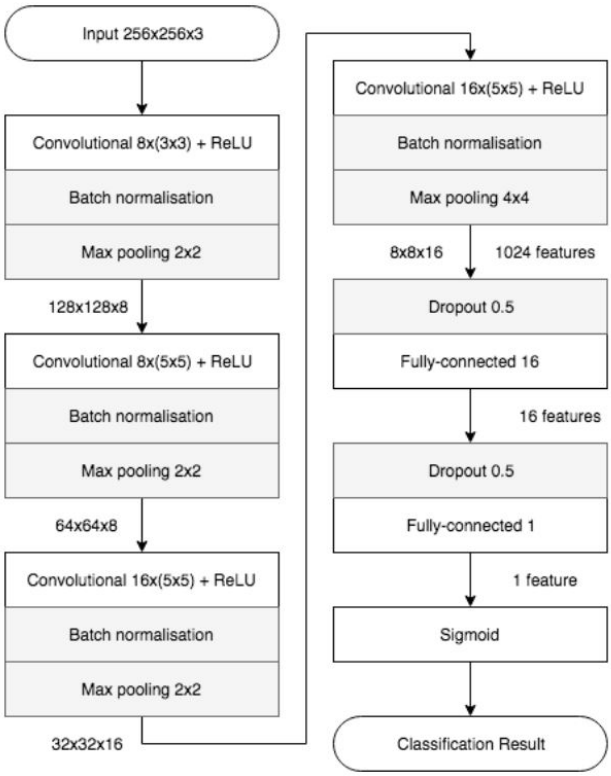


Figure 2.6: The intricate network architecture of Meso-4, detailing the layers, parameters, and output sizes. Adapted from [5].

an advanced variant of the inception module [28] into the initial convolutional layers. This inception module is designed to concatenate outputs from convolutional layers employing a diverse array of kernel shapes. Such an arrangement enriches the model's function space, thereby enhancing its optimization capabilities. Instead of conventional 5x5 convolutions, MesoInception-4 adopts the use of 3x3 dilated convolutions [29], a strategic choice that ensures effective handling of multiscale information while maintaining a focus on high semantic detail. Moreover, the inclusion of additional 1x1 convolutions in MesoInception-4 plays a dual role. Firstly, it aids in dimension reduction, a crucial aspect for managing computational efficiency. Secondly, it acts as a skip-connection [30], a feature that further enhances the model's performance. This specific model, characterized by its unique set of hyperparameters, encompasses a total of 28,615 trainable parameters, a testament to its advanced and refined architecture.

2.4 Methodologies in Deepfake Detection: Landmark-based Deepfake Detection

2.4.1 Advanced Facial Feature Analysis Combined with LSTM for Deepfake Detection

The groundbreaking study [7] explores the field of deepfake video detection through a novel integration of facial feature analysis and Long-Short Term Memory (LSTM) Deep Networks. This innovative approach exploits the spatial complexities of facial landmarks along with the temporal sequences of video data, aiming to improve the accuracy of deepfake identification.

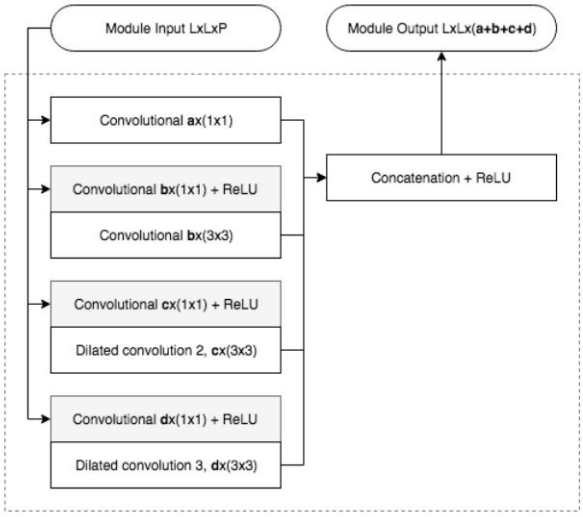


Figure 2.7: Detailed architecture of the inception modules employed in MesoInception-4, with parameterization using a, b, c, d N. The dilated convolutions are executed without stride, a critical design choice. Adapted from [5].

A significant step in deepfake detection technologies. Central to the study's methodology is the use of OpenFace2[6], an open-source facial landmark detection tool. This software meticulously identifies and records coordinates for 68 facial landmarks. Leveraging these landmarks, the study extracts key facial regions—namely, the mouth, eyes, and nose—and compares these with the whole face to determine the efficacy of specific region analysis in deepfake detection.

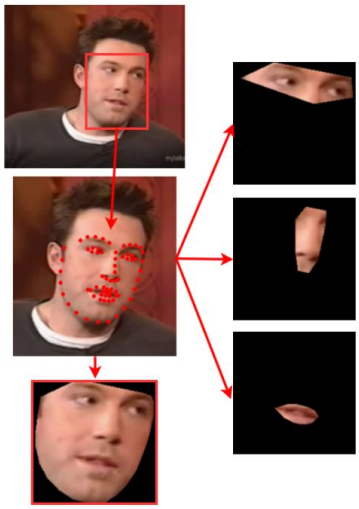


Figure 2.8: Preprocessing steps: (1) Face detection, (2) extraction of facial landmarks with OpenFace2[6], (3) extraction and alignment of face and (4) facial regions of interest (eyes, nose, mouth). Adapted from [7].

The model extracts these facial regions from every fifth frame. The extracted faces are then aligned and cropped, standardizing each to a resolution of 299x299 pixels for uniformity. This process encompasses several crucial stages: detecting the face, mapping facial landmarks, aligning and extracting the entire face, and isolating specific regions based on these landmarks.

Such a comprehensive analysis of each facial area significantly increases the model's ability to detect authentic content from manipulated videos. The study employs a hybrid model, integrating a CNN with an LSTM network. The CNN component is proficient at handling spatial data, extracting salient feature vectors, while the LSTM, known for its effectiveness in sequence processing, analyses these vectors for temporal patterns. This combination enables a robust analysis of both spatial and temporal elements in video sequences. The LSTM network operates following an "encoder" phase, which, in this case, is fulfilled by the Xception network[22]. Inspired by the Inception V3 architecture[31] but utilizing depthwise separable convolutions, Xception, pre-trained on the ImageNet dataset[32], has shown efficacy in both image classification and deepfake detection. The Xception model undergoes fine-tuning, with its last fully connected layer replaced by two additional layers, also fine-tuned. This results in a 256-length output feature vector, serving as the LSTM's input. The LSTM network comprises two layers, each with 256 hidden units, processing a feature vector from Xception for every fifth frame, totalling 60 frames for a 10-second sequence at a rate of 6 frames per second. This frame count is chosen based on research indicating a negligible accuracy difference beyond this number. The output from the LSTM passes through a fully connected layer with a dropout probability of 0.7, and a final Sigmoid activation layer outputs the probability of the video being either deepfake or genuine.

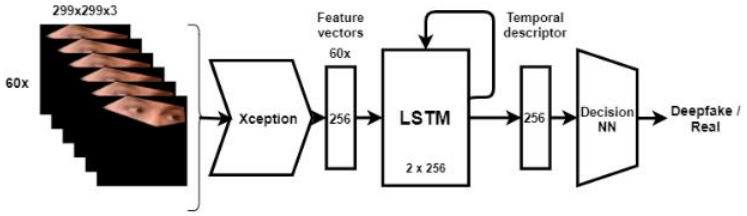


Figure 2.9: CNN-LSTM architecture. Xception network will output 60 feature vectors for a temporal sequence of 60 images. The 2 layer LSTM will output a temporal descriptor which is fed to the decision fully connected layers to obtain the deepfake probability. Adapted from [8].

The model's final outputs are preserved and utilized to train a late fusion algorithm. This algorithm combines deepfake probabilities from all analysed facial regions using a simple deep neural network, culminating in a deepfake probability score.

2.4.2 Innovative Approach in Deepfake Detection: Utilizing Spatial and Temporal Landmark Features

The study by Li et al. [10] presents a novel approach to address the challenges associated with detecting deepfake videos. In response to common processing transformations such as compression, the research introduces a unique methodology that leverages spatial and temporal features extracted from facial landmarks. This approach significantly enhances the accuracy of deepfake detection. The proposed methodology combines spatial and temporal data from facial landmarks, drawing inspiration from the work of Yang et al. [9]. It involves the selection of stable facial landmarks and the subsequent construction of facial vectors based on these landmarks. Spatial angle features are derived from the angles between these

facial vectors, while temporal rotation angle features are computed from the rotation angles of these vectors across consecutive frames. These spatial and temporal features are then used to create a feature vector for support vector machine (SVM) classification, with the aim of improving detection accuracy by capturing spatial inconsistencies and temporal irregularities in deepfake videos.

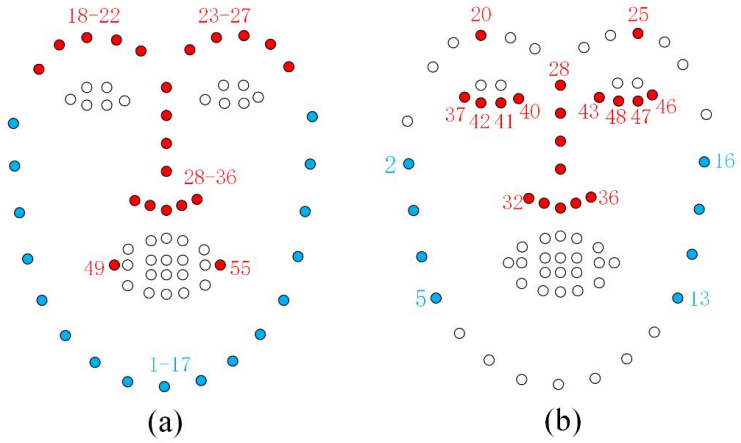


Figure 2.10: Comparison of two landmark selection schemes. (a) Landmark selection in [9]. (b) Landmark selection in [10] method. Adapted from [10].

The research emphasizes the critical role of facial landmark selection and vector construction in deepfake detection. It acknowledges that not all facial landmarks exhibit the same level of activity or error in deepfake videos. Specifically, landmarks located on the eyes and mouth—crucial regions for conveying facial expressions—are more challenging to simulate accurately in deepfakes. Moreover, highly active landmarks can introduce classification errors due to compression noise in videos. To address this issue, the study assesses the activity of each landmark across all frames and selects those with activity levels below a predetermined threshold. The selected landmarks are then used to construct facial vectors, incorporating both spatial and temporal features, to enhance deepfake detection. This approach underscores the significance of a thoughtful landmark selection process to strike a balance between detection accuracy and resilience against video compression artefacts.

In the spatial domain, the study defines the relationship between facial vectors using the included angle between two vectors. This angle, denoted as θ_a^i for a given landmark a on the frame i, is computed as follows:

$$\theta_a^i = \arccos\left(\frac{\vec{L}_{a,b1}^i \cdot \vec{L}_{a,b2}^i}{\|\vec{L}_{a,b1}^i\|\|\vec{L}_{a,b2}^i\|}\right)$$

Here, $\vec{L}_{a,b1}^i$ and $\vec{L}_{a,b2}^i$ represent two neighbouring facial vectors with a as their common vertex. The spatial angle feature for frame i is defined as $\{\theta_a^i\}$, where a varies over all selected landmarks. For a video clip comprising N frames, the set of spatial angle features is represented as $\theta_a = \{\theta_a^i, \forall a, i \in [1, N]\}$.

In the realm of temporal analysis, the study calculates the rotation angle of the same facial

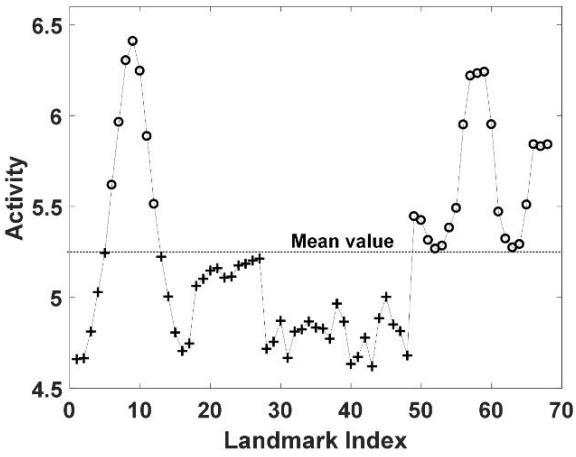


Figure 2.11: Activity of the 68 landmarks. "+" refers to the value lower than the mean. "∘" refers to the value greater than or equal to the mean. Activity of the 68 landmarks in Deepfake-TIMIT[11]. Adapted from [10]

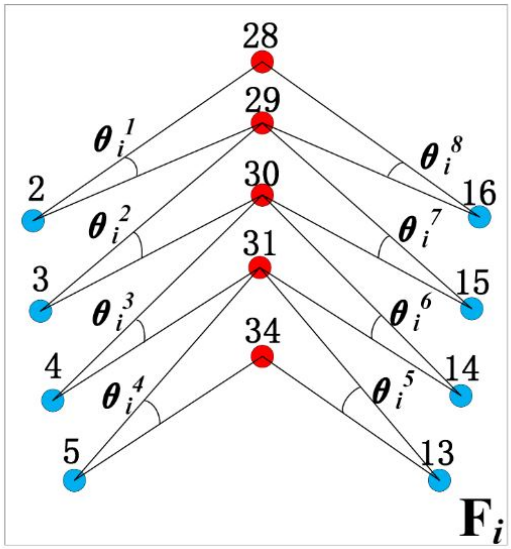


Figure 2.12: Construction of 8 spatial angles. Adapted from [10].

vector across two consecutive frames using the inner product formula:

$$\phi_{a,b}^{i} = \arccos\left(\frac{\vec{L}_{a,b}^{i} \cdot \vec{L}_{a,b}^{i+1}}{\|\vec{L}_{a,b}^{i}\|\|\vec{L}_{a,b}^{i+1}\|}\right)$$

Here, $\vec{L}_{a,b}^{i}$ and $\vec{L}_{a,b}^{i+1}$ represent the same facial vector on frames i and, $i+1$ respectively. For a video clip consisting of N frames, the set of temporal rotation angle features $\phi_{a,b}$ is defined as $\{\phi_{a,b}^{i}, \forall a, \forall b, i \in [1, N]\}$. To construct the SVM feature vector, the study utilizes the spatial and temporal relationships among facial vectors. Real video clips, captured naturally through various devices, exhibit consistent connections between frames in terms of content and context, a consistency that is often disrupted in deepfake videos due to manual interventions. The research proposes an SVM-based classifier that employs spatial angles θ_a^i and temporal rotation angles $\phi_{a,b}^{i}$ to uncover these spatial and temporal artefacts in deepfake

content. The combined information from these angles is used to compute statistics along the temporal axis, including their first four-order moments and first-order autocorrelation coefficients. The SVM feature vector is then constructed as follows:

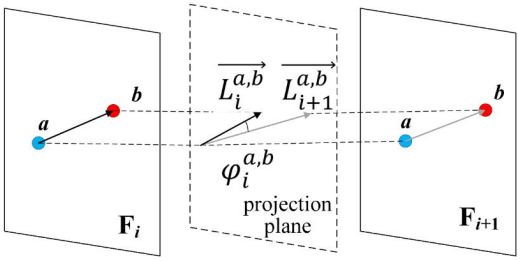


Figure 2.13: Temporal rotation angle. Adapted from [10].

$$
\begin{aligned}
\theta_a^i &= \frac{\sum_{i=1}^{N} \theta_a^i}{N} \\
\mathrm{Var}(\theta_a^i) &= \frac{\sum_{i=1}^{N} (\theta_a^i - \theta_a^i)^2}{N} \\
\mathrm{Sk}(\theta_a^i) &= E\left[\left(\frac{\theta_a^i - \theta_a^i}{\sqrt{\mathrm{Var}(\theta_a^i)}}\right)^3\right] \\
\mathrm{Kur}(\theta_a^i) &= E\left[\left(\frac{\theta_a^i - \theta_a^i}{\sqrt{\mathrm{Var}(\theta_a^i)}}\right)^4\right] \\
\mathrm{AR1}(\theta_a^i) &= \frac{\sum_{i=1}^{N-1} (\theta_a^i - \bar{\theta}_a^i)(\theta_a^{i+1} - \bar{\theta}_a^{i+1})}{\sqrt{\sum_{i=1}^{N-1} (\theta_a^i - \bar{\theta}_a^i)^2} \sqrt{\sum_{i=2}^{N} (\theta_a^i - \bar{\theta}_a^i)^2}}
\end{aligned}
$$

In these equations, θ_a^i, $\mathrm{Var}(\theta_a^i)$, $\mathrm{Sk}(\theta_a^i)$, $\mathrm{Kur}(\theta_a^i)$, and $\mathrm{AR1}(\theta_a^i)$ represent the mean, variance, skewness, kurtosis, and first-order autocorrelation coefficient of θ_a^i respectively. The length of the video clip is denoted as N. Similar calculations are performed for $\phi_{a,b}^i$. Consequently, the SVM feature vector comprises 800 dimensions, facilitating the classification process. This innovative approach, inspired by previous research, focuses on selecting stable landmarks, constructing facial vectors, and computing spatial and temporal angles to identify discrepancies in deepfake videos. The methodology emphasizes precise landmark selection to mitigate compression noise effects in video analysis. The angles between facial vectors and their temporal rotation across frames form the basis for an SVM classifier, enhancing detection accuracy.

2.4.3 Advancing Deepfake Detection with Multimodal Graph Learning: A Novel Approach in Digital Content Verification

Among the recent significant improvements made in the domain of deepfake detection is the work presented in [8]. This paper introduces a pioneering approach, diverging notably from traditional methods in the field. It proposes a novel framework that integrates di-

verse data modalities, including spatial, frequency, temporal, and facial landmark features, into a cohesive graph learning mechanism. This multimodal integration is essential, offering a more comprehensive representation of video data, thereby enhancing the robustness and accuracy of deepfake detection. Building upon foundational research such as [33], this multimodal graph approach contrasts methodologies discussed in works like [34], highlighting the rapidly evolving landscape of deepfake detection technologies. The subsequent discussion will delve into a detailed examination of the "Multimodal Graph Learning for Deepfake Detection" paper, underscoring the innovation of the proposed method and contextualizing it within the broader spectrum of ongoing research against deepfake technology. In this research, the authors employed a modified Xception architecture to extract spatial domain features, utilizing parameters pre-trained on the ImageNet dataset [32]. They operated on pre-processed fragments represented as $x \in \mathbb{R}^{C\times H\times W}$, where C, H, and W correspond to $3, 320$, and 320, respectively. The original Xception architecture outputs feature maps with dimensions $\mathbb{R}^{2048\times 7\times 7}$. Recognizing the importance of low-level features in deepfake detection, the authors redesigned the Xception architecture and developed an enhanced version incorporating a multiscale fusion module, as shown in Figure 2.14.

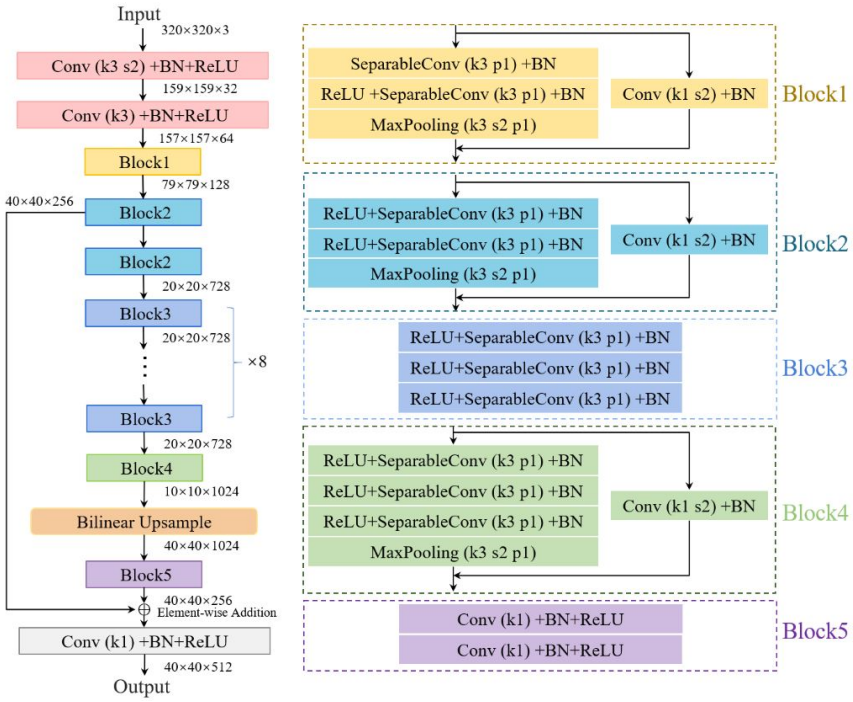


Figure 2.14: Overview of the improved Xception model architecture. Adapted from [8]

The improved Xception model combines feature maps from `block2` and `block5`, subsequently processed through a convolution layer with a kernel size of 1 to obtain the multiscale features X_s, where $X_s \in \mathbb{R}^{512\times 40\times 40}$, featuring a channel size of 512, four times smaller than the original model. The authors then describe a method for extracting frequency features, transforming input from the spatial domain to the frequency domain using the Discrete Cosine

Transform (DCT). A binary mask $M \in \mathbb{R}^{320\times320}$ is employed, defined by

$$M_{ij} = \begin{cases} 1, & \text{if low} < i + j < \text{up} \\ 0, & \text{otherwise} \end{cases}$$

with 'low' and 'up' as the lower and upper cut-off frequencies, respectively. In [35], [36], they highlight the efficiency of using filters in different bands for spectrum extraction. A residual structure with a learnable mask $M_s \in \mathbb{R}^{320\times320}$ for each band is used for adaptive forgery detection. The resulting features, y_f, are calculated as:

$$y_f = D(x) \odot \begin{pmatrix} M_{\text{low}} + \sigma(M_s^{\text{l}}) \\ M_{\text{mid}} + \sigma(M_s^{\text{m}}) \\ M_{\text{hig}} + \sigma(M_s^{\text{h}}) \\ M_{\text{all}} + \sigma(M_s^{\text{a}}) \end{pmatrix}$$

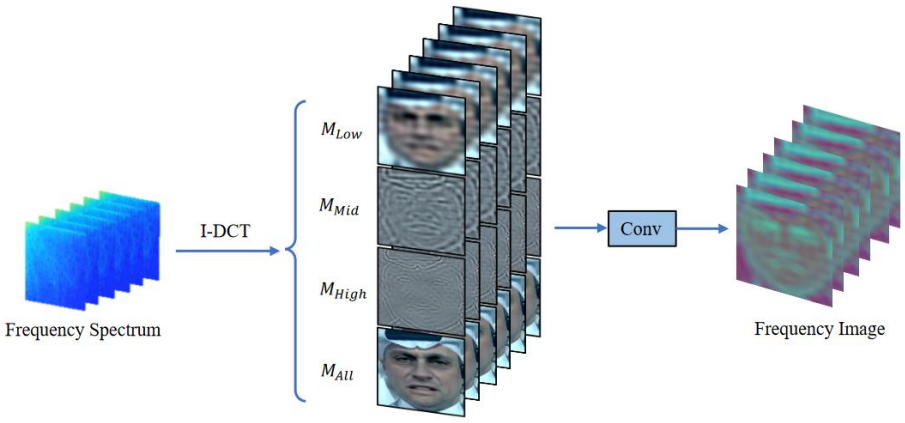


Figure 2.15: Overview of the Frequency Module (FM) utilized in the proposed method, which involves applying an inverse Discrete Cosine Transform (I-DCT) to the input image. We then apply four different filters (M_{Low}, M_{Mid}, M_{High}, M_{All}) to obtain separate frequency components. The outputs from these filters are then combined using a convolutional layer to synthesize the frequency image with 3 channels. Adapted from [8]

The Spatial-Frequency-Fusion (SFF) Module processes spatial and frequency input feature maps through convolutional layers. The features are then directed to a Cross-Modal Transformer (CMT) that employs self-attention to learn cross-modal dependencies. The CMT consists of a multi-head self-attention (MHA) layer and a feedforward neural network with residual connections and layer normalization. The output from MHA is given by:

$$Z_s = \text{MHA}(X_s, X_f, X_f) + X_s,$$
$$Z_f = \text{MHA}(X_f, X_s, X_s) + X_f,$$

Gating coefficients G_s and G_f are used to fuse the spatial and frequency features, denoted as $X_{sf}f$, obtained through element-wise multiplication of these coefficients with their respective

feature maps:

$$X_{sf} = G_s \odot Z_s + G_f \odot Z_f,$$

The Landmark Graph Learning (LGL) Module incorporates facial landmark information for robustness enhancement. Landmarks are represented as $L \in \mathbb{R}^{B\times 68\times 2}$ for batch size B, and an adjacency matrix $A \in \mathbb{R}^{B\times 68\times 68}$ is computed based on Euclidean distances between landmarks. Two graph attention layers (GATs) update the graph-level representation, each consisting of a linear projection and attention mechanism. The hidden features at layer i are $H^{(i)} \in \mathbb{R}^{B\times 68\times d_i}$, and the GAT updates them as:

$$H^{(i)} = \text{ReLU}\left(\sum_{j=1}^{68}\sum_{k=1}^{68} \alpha_{j,k}^{(i)} \cdot \text{Linear}(H_j^{(i-1)})\right),$$

The graph-level representation is averaged over landmarks:

$$X_{\text{lmk}} = \frac{1}{68}\sum_{j=1}^{68} H_j^{(N)},$$

Finally, the Multimodal Feature Fusion method fuses the landmark modality X_{lmk} and the spatial-frequency-domain modality X_{sff} by concatenating them and applying a learnable weight matrix M_l:

$$X_u = (X_{\text{sff}} \,\|\, X_{\text{lmk}}) \cdot M_l,$$

yielding $X_u \in \mathbb{R}^{512\times 40\times 40}$, the joint representation of spatial, frequency, and landmark modalities, which is then passed through a max pooling layer. After the multimodal feature fusion process, the proposed method progresses to extract temporal features from video frames. This unique approach involves treating video frames as nodes in a graph to model their temporal relationships, using a Graph Neural Network (GNN) equipped with multiple Graph Layers. This technique effectively learns the temporal dependencies between frames. The construction of the graph is a vital aspect, with video frames as nodes connected by edges that represent their pairwise similarities. The application of a Multi-Head Graph Attention mechanism is a significant innovation, enabling the detailed capture of temporal dependencies, crucial in the analysis of video frame sequences. In this graph, each node (a video frame) is initialized with the combined representation obtained from the previous feature fusion stages, enriching the graph with spatial, frequency, and landmark data. This rich initialization enhances the analysis of temporal features.

The GNN processes these interconnected nodes, and the Multi-Head Graph Attention mechanism's dynamic focusing ability allows for an in-depth evaluation of temporal features. This is particularly important for deepfake detection, where subtle temporal inconsistencies can be indicative of manipulation. The GNN architecture is composed of multiple Graph Layers, each employing a Multi-Head Graph Attention mechanism. The architecture's adaptability in terms of layers and heads efficiently models the temporal dependencies between frames. An Attention Module is applied post-GNN layers to derive final video-level features. Edge features, representing the relationships between video frames, are formulated based on pairwise similarity. For an input feature $F \in \mathbb{R}^{B\times T\times D}$, cosine similarity is computed between

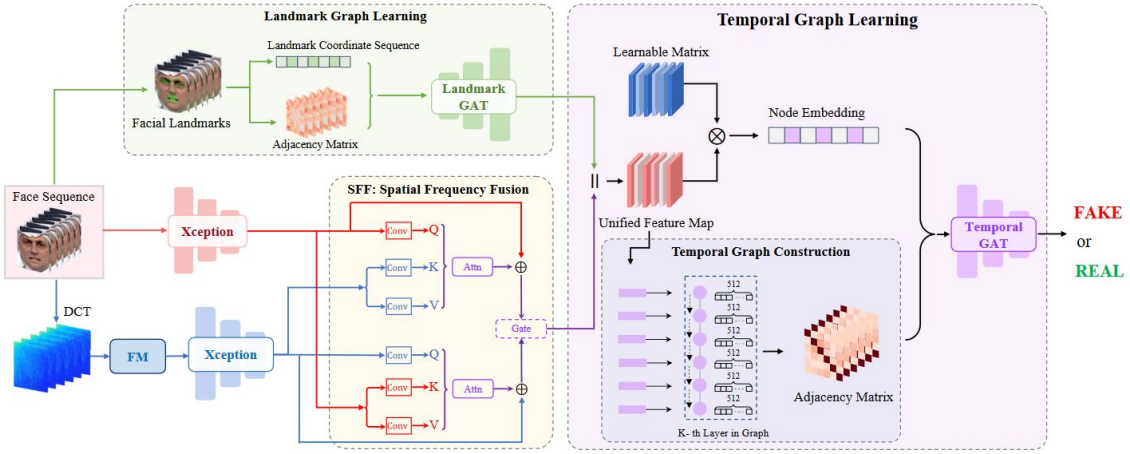


Figure 2.16: **Overview of the pipeline of the proposed framework:** where $\otimes$ denotes matrix multiplication, $\oplus$ denotes the element-wise addition, and $\|$ denotes the concatenate operation. For convenience, it only demonstrates the pipeline using six frames in this figure. Adapted from [8]

all pairs of normalized feature vectors, forming an edge feature matrix $E \in \mathbb{R}^{B\times N\times N}$, where $N = T$. Each matrix entry E_{ij} encapsulates the similarity between frames i and j. The Graph Layer includes a Multi-Head Graph Attention mechanism, Layer Normalization, and a feed-forward network. It processes node features X, the adjacency matrix A, and edge features E, producing a transformed feature $H \in \mathbb{R}^{B\times N\times D}$. The Multi-Head Graph Attention is defined by:

$$H_i = \left(\prod_{k=1}^{m} \text{head}_k(X, A, E) \right) W^O,$$

with attention coefficients calculated as:

$$\alpha_{i,j}^k = \text{SoftMax}\left(\frac{1}{\sqrt{D}} (W_k^Q X_i)^\top \left(\frac{1}{\sqrt{D}} W_k^K X_j + E_{i,j} W_k^V \right) \right).$$

The output H effectively captures temporal dependencies between frames, further refined by Layer Normalization and a feed-forward network. Finally, the Attention Module aggregates node features into a single vector representing the entire video, X_{gat}. This architecture captures local and global temporal dependencies in video frames for a comprehensive temporal analysis. In the model's final stage, an MLP layer extracts deeper semantic information from the features, using several 3×3 convolutional and normalization layers. The predicted label is calculated using:

$$\hat{Y} = \text{SoftMax}(\text{MLP}(X_{\text{gat}})).$$

The model is optimized using the cross-entropy loss function L_c, with parameters updated via back-propagation. This completes the discussion on the Multimodal Graph Learning approach for deepfake detection, highlighting its advanced graph-based temporal feature extraction and optimization techniques to boost detection performance.

2.5 Potential Methodologies for Enhancing Deepfake Detection

2.5.1 Detailed Explanation of Triplet Attention with Equations

Triplet Attention[12] is an attention mechanism created to understand interconnections across different dimensions in convolutional neural networks instead of just focusing on a single axis or spatial dimension, like conventional attention mechanisms do. By extending its attention across the height and width dimensions and also the channel dimension through rotating the input tensor effectively broadens the network's capability to capture intricate feature interactions and improve feature portrayal. Given an input feature map $\mathbf{X} \in \mathbb{R}^{C \times H \times W}$, where C, H, and W represent the number of channels, height, and width of the feature map, respectively, Triplet Attention operates through three key branches: attention across the height, width, and channel dimensions.

2.5.1.1 Attention Across Height

To compute attention along the height dimension, the input tensor $\mathbf{X}$ is rotated such that the height axis becomes the focus. We transpose the input tensor as follows:

$$\mathbf{X}_H = \mathbf{X}^\top \in \mathbb{R}^{H \times C \times W}$$

After rotating the feature map to place the height dimension at the beginning and keeping the width and channel dimensions, a convolution process is then used to generate an attention map focusing specifically over the height aspect.

$$\mathbf{A}_H = \sigma\left(\text{Conv}_{1\times1}\left(\mathbf{X}_H\right)\right)$$

where σ represents the sigmoid activation function, and $\text{Conv}_{1\times1}$ denotes a 1x1 convolution that compresses the features into a single attention map for the height dimension. The attention map $\mathbf{A}_H \in \mathbb{R}^{H \times 1 \times W}$ is then multiplied with the original input tensor to emphasize important features:

$$\mathbf{X}_H^{\text{attended}} = \mathbf{A}_H \odot \mathbf{X}_H$$

where $\odot$ denotes element-wise multiplication.

2.5.1.2 Attention Across Width

Similarly, attention is computed along the width dimension by first rotating the input tensor $\mathbf{X}$ so that the width axis becomes the focus:

$$\mathbf{X}_W = \mathbf{X}^\top \in \mathbb{R}^{W \times C \times H}$$

An attention map is generated along the width axis using a 1x1 convolution:

$$\mathbf{A}_W = \sigma\left(\text{Conv}_{1\times1}\left(\mathbf{X}_W\right)\right)$$

where $\mathbf{A}_W \in \mathbb{R}^{W \times 1 \times H}$ is the attention map across the width. The original feature map is then enhanced by multiplying with the attention map:

$$\mathbf{X}_W^{\text{attended}} = \mathbf{A}_W \odot \mathbf{X}_W$$

2.5.1.3 Attention Across Channels

Finally, the computation of attention extends to the channel dimension. Channel attention differs from dimensions as it doesn't involve any transposition. It primarily concentrates on the interplay, among feature channels.

$$\mathbf{A}_C = \sigma\left(\text{Conv}_{1\times1}\left(\mathbf{X}\right)\right)$$

where $\mathbf{A}_C \in \mathbb{R}^{C \times 1 \times 1}$ represents the attention weights for each channel. This attention map is then multiplied element-wise with the original input tensor to emphasize the most important channels:

$$\mathbf{X}_C^{\text{attended}} = \mathbf{A}_C \odot \mathbf{X}$$

2.5.1.4 Combining the Three Branches

After calculating the attention maps, for the height and width as well as the channel dimensions and merging the results from all three attention branches together to obtain the final attended feature map; they are combined through element wise addition.

$$\mathbf{X}^{\text{final}} = \mathbf{X}_H^{\text{attended}} + \mathbf{X}_W^{\text{attended}} + \mathbf{X}_C^{\text{attended}}$$

This combination ensures that the model has captured feature interactions across all three dimensions.

2.5.1.5 Advantages of Triplet Attention

- **Cross-Dimensional Feature Learning**: Using Triplet Attention enables the model to understand relationships among height and width dimensions as well channel variations, effectively enhancing its capacity to analyse intricate connections, within the data.
- **Efficiency**: Triplet Attention proficient at capturing feature relationships, and it doesn't add much computational burden, so it can be easily incorporated into current frameworks without a substantial rise, in model intricacy.
- **Improved Focus on Relevant Features**: By utilizing attention across aspects simultaneously, the system can concentrate on critical sections of the feature map improving its effectiveness in activities, like detecting deepfake content accurately.

Triplet Attention presents a powerful approach to capture interconnections between different dimensions in CNNs efficiently. By directing attention to the height, width and channel dimensions, the network can improve its ability to comprehend complex feature interactions,

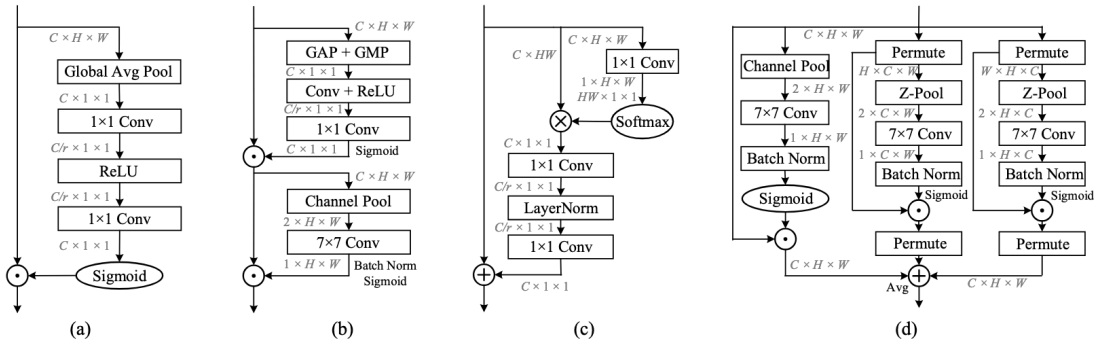


Figure 2.17: Comparisons with different attention modules: (a) Squeeze Excitation (SE) Module; (b) Convolutional Block Attention Module; (c) Global Context Module; (d) Triplet Attention. The feature maps are denoted by their feature dimensions, e.g. $C \times H \times W$ denotes a feature map with channel number C, height H, and width W. $\otimes$ represents matrix multiplication, $\odot$ denotes broadcast element-wise multiplication, and $\oplus$ denotes broadcast element-wise addition. Adapted from [12]

leading to stronger feature representations. This attention method proves beneficial, in detecting deepfakes as it can effectively identify subtle spatial and channel based alterations by recognizing feature interdependencies across various dimensions.

2.5.2 Gated Multimodal Units for Information Fusion

This paper [13] presents a network design that aims to blend various types of information (such as text and images or audio) by adapting the weightage of each type dynamically through multiplicative gates in GMUs(Gated Multimodal Units for Information Fusion) to determine their impact, on the final prediction result.

Key Contributions:

- **Gated Multimodal Units (GMUs):** Taking inspiration from Gated Recurrent Units (GRUs) the GMUs utilize gating mechanisms to regulate the influence of each modality on the result, where the gates gather inputs, from various modalities and make real time decisions on how to mixture them for improving the comprehensive representation.
- **Multimodal Fusion:** Instead of just putting together feature representations from various sources in a straightforward manner, the GMU is trained to carefully merge these representations based on the situation. This becomes quite handy for situations in which one source of information may hold more significance than others.
- **Evaluation:** The GMUs were assessed in a task involving predicting movie genres using both the movie's plot (text data) and poster (images). The outcomes demonstrate that GMUs surpass typical fusion methods such, as concatenation and mixture of experts.

2.5.2.1 GMU Architecture

The GMU consists of three components for each modality:

- A feature extractor for each modality (e.g., a CNN for images, a word embedding model for text).

- A gate that dynamically controls how much the representation from each modality should contribute to the final output.
- A combination of the gated outputs to form a multimodal representation.

For two modalities x_v (visual) and x_t (textual), the GMU can be described by the following equations:

$$h_v = \tanh(W_v \cdot x_v)$$

$$h_t = \tanh(W_t \cdot x_t)$$

$$z = \sigma(W_z \cdot [x_v, x_t])$$

$$h = z \odot h_v + (1 - z) \odot h_t$$

Where:

- h_v and h_t are the representations for the visual and textual modalities.
- W_v, W_t, and W_z are learnable weight matrices.
- z is a gating variable that decides how much each modality contributes to the final output.
- $\odot$ represents element-wise multiplication.
- $[x_v, x_t]$ is the concatenation of the two modalities.

The gating mechanism z is a sigmoid function that outputs a value between 0 and 1, controlling the balance between the two modalities. If z is close to 1, the model favours the visual modality; if z is close to 0, it favours the textual modality.

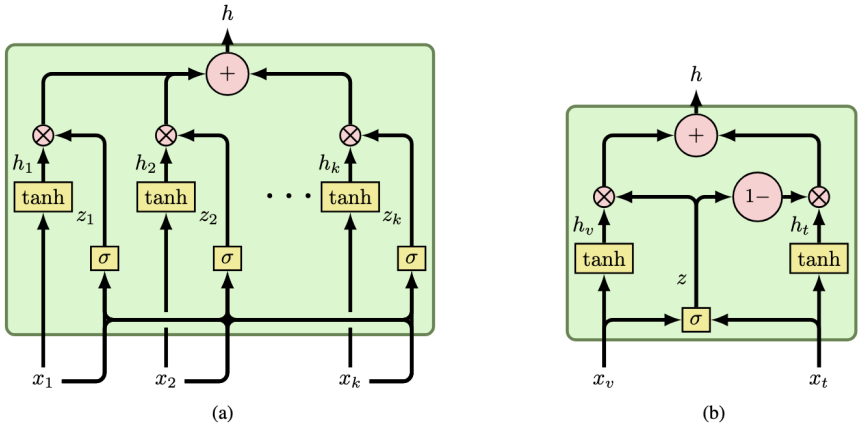


Figure 2.18: Illustration of gated units. a) The proposed model to use with more than two modalities. b) A simplification for the bimodal approach. Adapted from [13]

2.5.2.2 Training and Results

The GMU is trained using backpropagation and gradient-based optimization, with stochastic gradient descent (SGD) and the Adam optimizer to learn the parameters of the GMU. The GMU was tested on the MM-IMDb dataset, which contains multimodal data for movies, including plot summaries (text) and posters (images). GMUs outperformed several baseline models, including simple concatenation and mixture of experts (MoE) models, achieving higher macro F1-scores for genre classification. The success of GMUs, in combining modalities through gating mechanisms, resulted in better outcomes than alternative fusion approaches.

2.5.2.3 Relevance to Deepfake Detection

GMUs are experts at adjusting the significance of input types dynamically, which proves valuable in spotting deepfakes that rely on a mix of data from different sources. The complexity of detecting deepfakes often demands models to assign varying importance to different features based on the task at hand. By utilizing Generalized Mutual Understanding Units (GMUs) in the identification of deepfake content, the system can dynamically concentrate on analysing expressions inconsistencies or any anomalies present in the data provided.

2.6 Datasets

2.6.1 FaceForensics++

The FaceForensics++[37] dataset, a cornerstone in deepfake detection, offers over 1,000 original videos, intricately altered using DeepFakes, Face2Face, FaceSwap, and NeuralTextures, culminating in a rich collection of over 1.8 million images(frames).

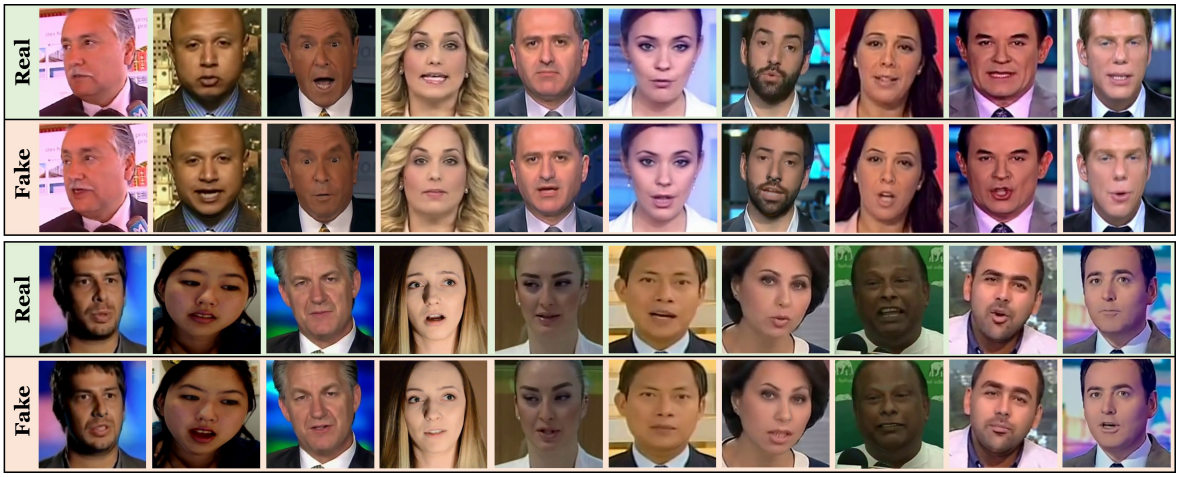


Figure 2.19: FaceForensics++ samples. Adapted from [14]

The dataset's diversity extends to various compression levels and image sizes, ranging from pristine, uncompressed videos to those compressed with the H.264 codec at varying bitrates. This spectrum represents the diverse quality of videos encountered online, from high-quality footage to those heavily compressed, often found on social media platforms, enhancing the dataset's practical relevance.

2.6.2 Celeb-DF

The Celeb-DF[16] dataset marks a groundbreaking leap in DeepFake technology, providing an extensive and high-quality video collection. It's designed to offer a more rigorous benchmark for DeepFake detection, responding to the urgent need for enhanced tools in this rapidly evolving domain. This dataset uniquely features a rich diversity of realistic DeepFake videos, elevating the complexity and authenticity required in detection tasks. With a comprehensive array of videos portraying various celebrities, it showcases a wide spectrum of appearances and expressions. The high quality of these DeepFakes, crafted using cutting-edge techniques, makes them indistinguishably realistic, thus posing a significant challenge for detection algorithms. The dataset's creation involved a detailed process, starting with the selection of genuine celebrity videos, primarily from YouTube. These videos spanned a broad range of celebrities, ensuring diversity in appearances and expressions. The DeepFake videos were then generated using advanced methods, significantly improved to enhance their realism. Special focus was placed on perfecting facial expressions and movements, overcoming common flaws observed in earlier datasets. High resolution and quality were key priorities in these crafted videos, augmenting the challenge for detection algorithms. This meticulous preparation phase was crucial in making the dataset a valuable tool for developing and evaluating DeepFake detection technologies, aiming to mimic real-life scenarios and appearances closely.

Figure 2.20: Celeb-DF samples. Adapted from [15]

Celeb-DF's distinction lies in its unique characteristics, making it a vital resource in DeepFake detection research. Its high-quality DeepFake videos, remarkably lifelike due to advanced synthesis techniques, address issues like facial alignment inaccuracies and unnatural expressions found in previous datasets. The dataset encompasses a diverse range of subjects, covering various ages, ethnicities, and genders, providing a broad spectrum for testing detection algorithms. The videos are consistently crafted, maintaining uniformity in lighting, resolution,

and facial features, adding complexity to detection tasks. This combination of realism, diversity, and quality positions Celeb-DF as a challenging and invaluable resource for researchers striving to develop more effective detection methods.

Chapter 3

Proposed Method

3.1 Introduction

Identifying deepfake videos accurately requires models that can analyse a wide range of complex features effectively. The new deepfake detection model, presented in this study, incorporates sophisticated techniques for spatial analysis as well as frequency and geometric examination.

- **Advanced Landmark Feature Extraction:** One key element of the study involves analysing features by measuring distances using Euclidean distances and angles using cosine similarity calculations. This technique in extracting features enables the model to understand not where facial landmarks are located, but also how they align geometrically with other important facial characteristics. By depicting the face in this manner the model improves its ability to detect changes, in deepfakes that might go unnoticed during regular visual checks.
- **Application of Triplet Attention and GMUs:** Triplet Attention enhances feature maps by focusing on the height, width dimensions and the channel dimension to capture complex feature interactions often missed by traditional attention methods. GMUs offer a flexible approach to merging multimodal data by dynamically adjusting the importance of each feature. This unique blend of attention and fusion techniques boosts the model's capacity to detect irregularities.
- **Comprehensive Multimodal Integration:** The model employs a sophisticated multimodal architecture that processes spatial, frequency, and geometric (landmark) features. The fusion of these diverse modalities is made by GMUs, followed by temporal modelling throug transformer encoders, ensuring that both visual appearance and temporal changes in facial geometry are analysed in detail. This comprehensive integration of multiple feature types is rarely present in the current deepfake detection literature.

3.2 Preprocessing Pipeline

In this model is important to focus solely on the faces of the possible victims of deepfake, face recognition and cropping ensures that the focus of the input remains on the face area, within video content as it holds the utmost importance in such scenarios when dealing with deepfake videos where facial attributes are often manipulated significantly to mislead viewers through digital alterations, hence highlighting the necessity to centre the analysis around facial elements specifically. The model capability to detect and isolate the face aids in minimizing any impact from surrounding background details which subsequently enhances its

effectiveness, in recognizing anomalies introduced by deepfake technology during video creation. Furthermore, aligning faces ensures that important facial landmarks, like the eyes and mouth, stay in the place in each frame, so the model can accurately detect changes in shape and expressions crucial for distinguishing genuine from fabricated videos. This preliminary step makes the model focus solely on alterations impacting facial features. Detecting faces in videos present a number of challenges when faced with issues, like lighting or unusual angles that can mislead algorithms. Differences in face dimensions and placement from one frame to another could result in imprecise bounding boxes that could either crop information or capture background elements. Small mistakes, in detecting faces, can have an impact on how the model performs. Having robustness in this processing stage is key, to getting trustworthy results.

3.2.1 Face Detection Using MTCNN

The MTCNN(Multi-task Cascaded Convolutional Networks)[38] algorithm is a strong method, for detecting faces that work in three steps; proposing regions of interest for faces first, then refining these areas to create accurate bounding boxes, and finally providing important facial features like eyes and mouth locations. The efficiency and reliability of MTCNN make it ideal for real time face detection, as it can handle face angles and harsh lighting situations effectively, with partial obstruction. For every frame, in the video stream, MTCNN identifies the facial bounding boxes and key points accurately. It then adjusts the bounding boxes to encompass a bit of the surrounding area of the face while excluding background details. This is managed by an expansion factor that slightly increases the size of the bounding box by a factor of 1.2 to create a margin, around the face. The new bounding box dimensions are calculated by extending both width and height from the box while ensuring it fits within the frame limits. After making adjustments, to the bounding box to fit the face region accurately in the frame's image capture area, the cropped face is resized to a size of 256 by 256 pixels. Before feeding the frames to the model, the cropped faces go through a normalization process that consists of dividing each pixel by 255, this procedure guarantees uniformity across frames and video while minimizing fluctuations.

3.2.2 Frequency Domain Transformation

Studying videos with domain conversions is crucial in spotting deepfake content, as it uncovers characteristics that are not easily visible in the spatial domain of images or videos. By converting video frames into the domain using techniques such as the FFT, it becomes easier to identify anomalies [39]. These conversions reveal patterns and uncommon high frequencies that are often overlooked by spatial analysis methods. Numerous research works have emphasized the importance of utilizing frequency based features to effectively detect deepfake content. It has been shown in studies that the Fourier Transform's capable of detecting irregularities in images that are absent in authentic ones—especially in the high frequency spectrum, where artificially created images frequently show anomalies [40]. During this step of the process, each frame is transformed into the domain using a two-dimensional FFT technique. This transformation converts the intensity levels into a range of elements. The FFT

Figure 3.1: Original frame and cropped face

operation works on the greyscale image by changing it from a grid of pixel values $I(x, y)$ to a format where each value signifies a frequency element $F(uv)$. The numeric values are then modified to centre the frequency elements, which enables a thorough examination of the distribution of frequencies. This modification results in a range where lower frequencies indicate gradual shifts and higher frequencies point out boundaries and irregularities that could imply changes in the information. To handle the array of values effectively, a logarithmic change is implemented to condense and steady the numerical information. Afterwards the values are adjusted to fit into a range (like from zero, to one) which aids in keeping consistency across various segments.

- **Greyscale Conversion:** The image is converted to greyscale. This step simplifies the analysis by focusing on intensity changes rather than colour details.
- **FFT and Frequency Centring:** A two-dimensional Fast Fourier Transform is applied to the image:

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} I(x, y)\, e^{-2\pi i \left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

  The frequency components are then shifted to centre the low frequencies.
- **Magnitude Spectrum Calculation:** The magnitude (or absolute value) of the complex FFT output is calculated:

$$S(u, v) = |F(u, v)|$$

  This reveals the intensity of each frequency component in the image.
- **Logarithmic Transformation:** To stabilize the data and manage large variations, a logarithmic transformation is applied:

$$S_{\log}(u, v) = \log\left(1 + S(u, v)\right)$$

This ensures that the data can be handled more efficiently by the model.

- **Normalization:** The adjusted magnitude values are standardized within a scale of zero to one by dividing each value by the maximum value:

$$S_{\text{norm}}(u, v) = \frac{S_{\log}(u, v)}{\max\left(S_{\log}(u, v)\right)}$$

This procedure guarantees uniformity across frames and video clips while minimizing fluctuations.

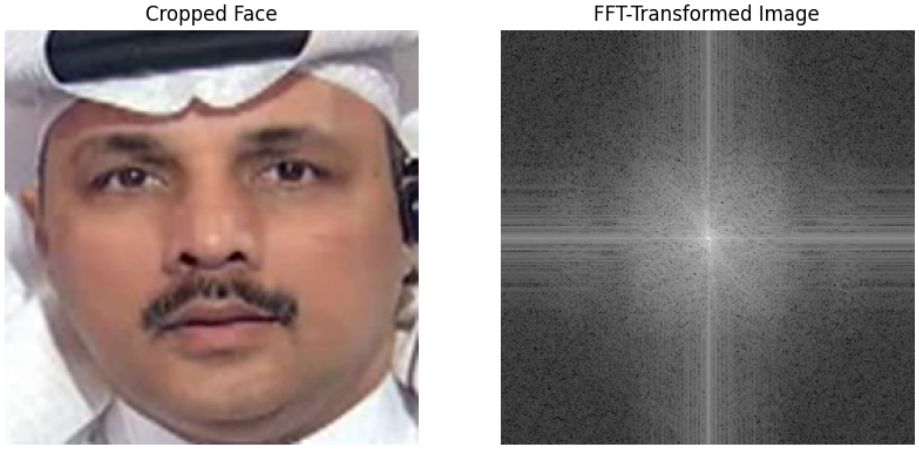


Figure 3.2: Cropped face and FFT Transformation of cropped face

3.2.3 Landmark Feature Extraction

Facial landmarks are points found throughout a face that indicate anatomical features such, as the eyes and mouth, among others. These points serve to map out the shape and structure of a face in detail for studying expressions and movements. Facial landmark detection algorithms are typically created to identify areas throughout the face that aid, in portraying facial structure aspects like proportions and the arrangement of features in relation to each other. These distinct markers provide insights for tasks involving analysis by observing how these landmarks shift and align in frames to uncover irregularities indicative of video modifications.

3.2.3.1 Landmark Extraction

Facial landmarks are extracted using dlib's state-of-the-art face detection and shape prediction algorithms. This process begins with detecting the face in a frame using a pre-trained detector. Dlib's frontal face detector, based on Histogram of Oriented Gradients (HOG), identifying regions of interest (ROIs) where a face is likely to be present. Once the face is localized, the frame is passed through a shape predictor trained to map 68 key points or landmarks, which represent critical facial features such as the eyes, nose, mouth, and jawline.

Figure 3.3: Cropped face with landmarks pinpointed as green dots

3.2.3.2 Converting landmarks into useful features

In this model, the facial landmarks are processed to extract 2 features. These include:

- **Euclidean Distances:** The distances between each set of points are calculated, in order to understand how they are positioned relative to one another. Such as the eyes distance from the nose and mouth or the jawlines position in relation, to other facial features.
- **Angular Relationships:** Angles between normalized coordinates of landmark points are calculated. This helps capture geometric consistency across frames.

3.2.3.3 Euclidean Distances

The model calculates distances, between landmarks, to understand the spatial connections. Given 68 landmarks, each represented by 2D coordinates (x, y), the model first calculates the differences between every pair of points using a difference matrix. This matrix contains the differences along both the x and y axes between each landmark pair. The Euclidean distance for each pair is then computed using the formula

$$d = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

where x_i and y_i represent the coordinates of two distinct points. Given that the distance matrix displays symmetry characteristics; solely the distinct distances (values found in the

upper triangle region of the matrix)'re taken out to prevent repetition issues. These unique distances are subsequently saved in a one array, as a component of the feature set utilized by the model. By prioritizing the relationships, between distances over the locations of landmarks in the model, this ensures adaptability to changes in how the face is rotated or moved around without compromising performance reliability even during imperfect alignment of the face. Furthermore, using distances offers a way to represent complex coordinate data while maintaining key geometric details intact. This functionality aids in identifying any distortions, like elongation or shrinking, that may be found in edited videos. This strategy is backed by studies such, as [8].

3.2.3.4 Angular Features

Angular features between facial landmarks are computed to capture the relative orientation between key facial points, aiding in the detection of manipulations. The process begins with 68 landmarks, each represented by 2D coordinates (x_i, y_i). These coordinates are first normalized by dividing each vector by its Euclidean norm, producing unit vectors for each landmark:

$$\text{norm\_coords}[i] = \frac{[x_i, y_i]}{\sqrt{x_i^2 + y_i^2}}$$

This normalization ensures that the angular measurements are not influenced by the size and position of the face. Next, the cosine similarity between every pair of normalized landmark vectors $\mathbf{a}_i$ and $\mathbf{b}_j$ is computed, giving the cosine of the angle θ between them:

$$\cos(\theta_{ij}) = \frac{\mathbf{a}_i \cdot \mathbf{b}_j}{\|\mathbf{a}_i\|\|\mathbf{b}_j\|}$$

where $\mathbf{a}_i$ and $\mathbf{b}_j$ represent the normalized coordinates of landmarks i and j, respectively. This calculation results in a cosine similarity matrix. To ensure numerical stability, the values are clipped to the valid range of $[-1, 1]$. The actual angles between landmarks are then computed by applying the inverse cosine (arccos) to the cosine similarity values, converting them to angles in degrees:

$$\theta_{ij} = \arccos(\cos(\theta_{ij})) \times \left(\frac{180}{\pi}\right)$$

Since the cosine similarity matrix is symmetric (i.e., $\cos(\theta_{ij}) = \cos(\theta_{ji})$), only the upper triangular part of the angle matrix is extracted to avoid redundancy. These angles are included in the model's feature vector to capture the relative positioning of facial landmarks. When used alongside features such as Euclidean distances, these angular measurements help the model identify subtle changes in facial geometry caused by manipulations. Cosine similarity is an effective method for angle calculations as it precisely quantifies the alignment between two vectors (in this case, landmark coordinates) regardless of their size or scale, making it suitable for analysing facial features [41]. When measuring dimensions using key points, distances and angles to assess symmetry and alignment of features is crucial for detection accuracy in the model's dual approach method. Combining these two types of features enhances the model's

resilience against manipulations by addressing structural deformations from distances and rotational distortions from angles.

3.2.3.5 Final Feature

When the distances and angle characteristics are merged together in a feature vector by concatenating them, this ensures that both absolute and relative geometric information are captured effectively. To maintain consistency, in scale for these features and ensure comparison, between them, they undergo normalization through max normalization process. The resulting combined feature vector is then presented as follows;

$$\text{combined\_features} = \left[\frac{\text{Euclidean} - \min_{\text{Euclidean}}}{\max_{\text{Euclidean}} - \min_{\text{Euclidean}}}, \frac{\text{Angle} - \min_{\text{Angle}}}{\max_{\text{Angle}} - \min_{\text{Angle}}} \right]$$

3.3 Model Architecture

The method suggested here, for identifying deepfake content, utilizes different types of data input to gather detailed video representations through sophisticated fusion of features and focusing techniques to enhance overall effectiveness.

3.3.1 Model Steps

- EfficientNet helps analyse both advanced characteristics, in spatial and frequency domain frames.
- Utilize Triplet Attention to boost areas, in spatial and frequency characteristics.
- Fuse low- and high-level spatial and frequency features using GMUs.
- Compute landmark features and project them to the required dimension.
- Fuse spatial, frequency, and landmark features using a Triple GMU.
- Apply positional encoding to fused features for temporal information.
- Use a Transformer encoder for temporal modelling across video frames.
- Perform final classification using a fully connected layer.

3.3.2 Feature Extraction with EfficientNet

EfficientNet stands out at recognizing visual patterns. It is essential because it is able to capture low level and high level features crucial, for spotting abnormalities in deepfake content [42] [43]. To allow the frequency domain data to flow through Efficientnet, the information is transformed into three channels before being inputted into EfficientNet. The reason, behind this step, is that the model is primarily structured to handle RGB images that consist of three-colour channels. EfficientNet expects input tensors of shape $(\text{batch\_size}, 3, \text{height}, \text{width})$, corresponding to three-channel images. This allows EfficientNet to process the frequency

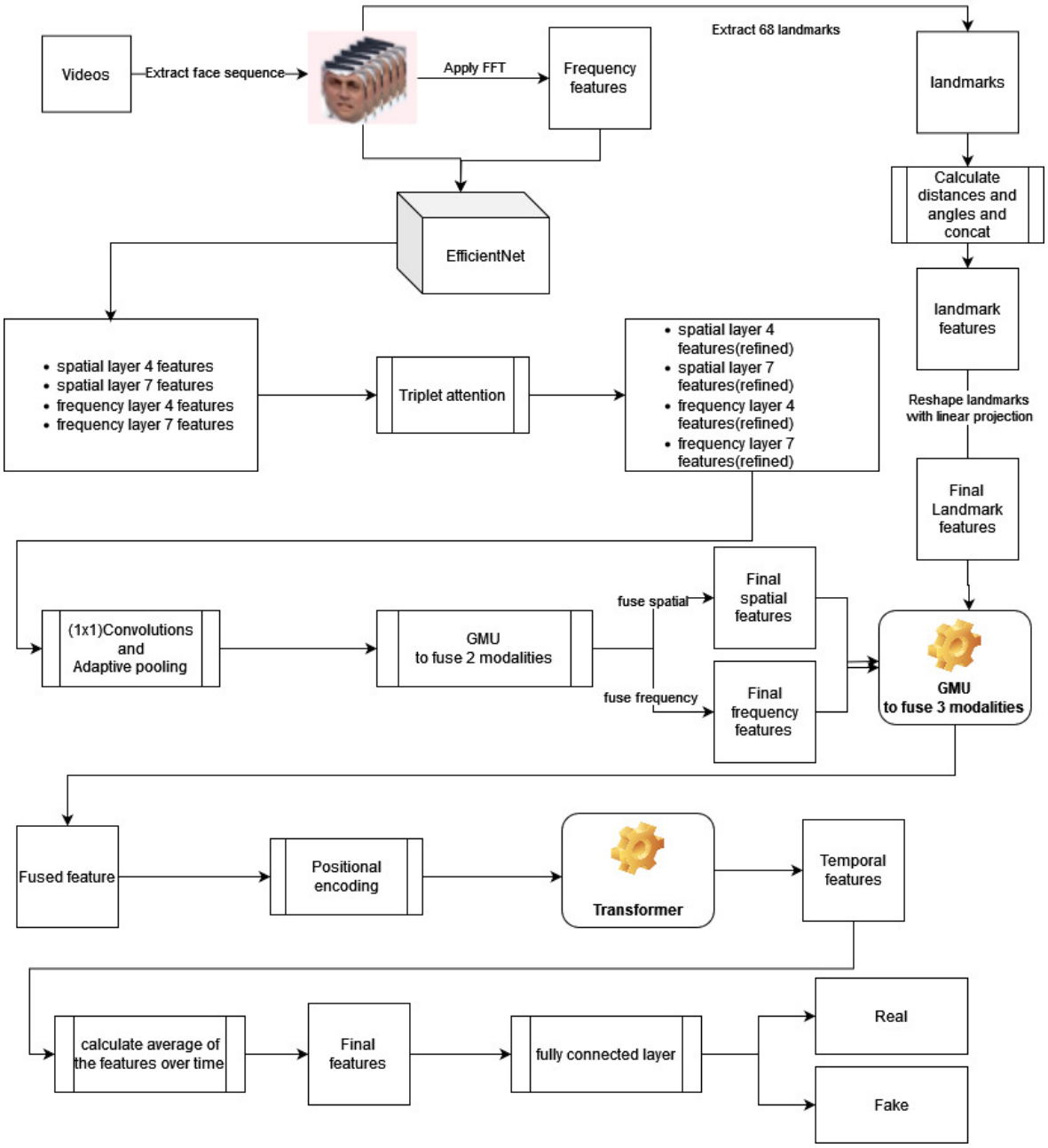


Figure 3.4: Diagram of the proposed Model Architecture

data similarly to RGB images, leveraging its capabilities on 3-channel inputs, while retaining the structure and patterns of the frequency domain. The model extracts feature maps from different layers. In EfficientNet architectures Layers 4 and 7 play a role, in deepfake detection by capturing features at varying levels of complexity and detail required for analysis. Layer 4 is dedicated to detecting low level features like edges and textures that help identify minute inconsistencies or subtle alterations in the manipulated images. While, Layer 7 focuses broadly by extracting high level features, such as structure and component relationships, across the image. Combining these two layers of data allows the system to gain a better view and enhance its capability to identify changes.

3.3.3 Triplet Attention

The Triplet Attention technique is applied individually to every feature map extracted from EfficientNetused(self attention) to enhance the model's capability, in capturing connections within the features from both higher and low levels and derived from both frequency and spatial domains successfully by focusing on the height, width and channel dimensions of the feature maps[12]. The attention process involves rotating the input feature map along three axes—height, width and channel, and calculating attention separately for each axis. This approach enables the model to concentrate on patterns, in each dimension. When the feature map is rotated horizontally and vertically to focus on connections and rotated along the channel axis to capture relationships, between channels; attention is computed for each axis before combining the attended maps to help the model grasp intricate cross dimensional connections effectively enhancing its capability to identify slight inconsistencies, across spatial and channel dimensions. The rationale for using Triplet Attention, is that this mechanism distinguishes itself from other attention mechanisms through its unique combination of components designed to capture richer and more comprehensive feature representations. It models cross-dimensional dependencies between spatial (height and width) and channel dimensions at once, unlike other mechanisms that treat these dimensions separately or focus on only one. The three-branch architecture ensures that all possible interactions are considered, with each branch capturing dependencies between channel-height, channel-width, and height-width dimensions. A key innovation is its rotation-based design, which rearranges tensor dimensions to effectively connect different dimensions, enabling the capture of elaborate patterns without significant computational complexity. Additionally, the Z-Pool layer combines global max and average pooling to retain essential information while reducing dimensionality, preserving the richness of feature maps more effectively than other methods. Triplet Attention also avoids dimensionality reduction when computing attention weights, maintaining the full dimensional integrity of feature maps, which is crucial for highlighting subtle details often lost in other attention mechanisms. Despite its ability to capture these complex relationships.

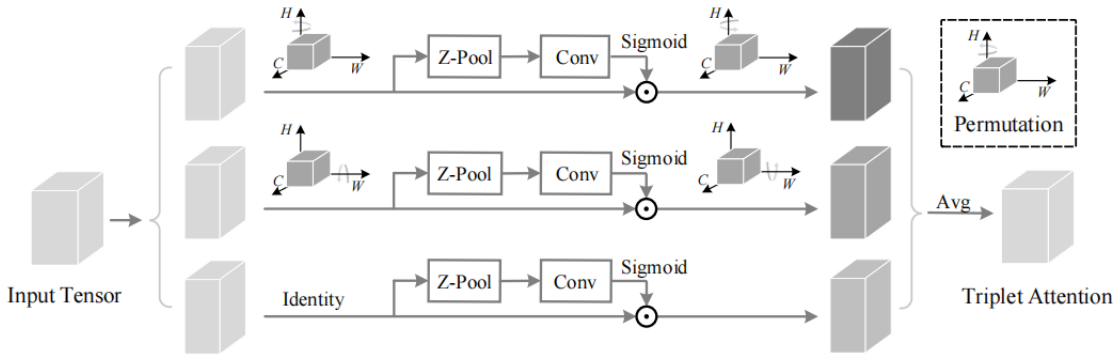


Figure 3.5: Illustration of the triplet attention, which has three branches. image extracted from [12]

3.3.4 Fusion of Spatial and Frequency Domain Features

Upon implementing Triplet Attention on the high level features in both spatial and frequency domains, the next step consists of refining and condensing the features through a sequence of steps before merging them using binary GMUs[13]. To start with, after applying Triplet Attention on the features extracted from both spatial and frequency domains in the model incorporates reducing feature dimensionality using 1x1 convolutional layers. These convolutional layers decrease the dimensionality of the extracted frequency features while retaining their crucial information intact to enhance computational efficiency, for subsequent operations. Low and high level spatial and frequency characteristics are decreased individually. Then comes the step where the model employs average pooling to the reduced feature maps by resizing them to an identical shape throughout all frames. This process condenses the details into a singular feature vector for each frame, which simplifies the merging of features across time intervals. Pooling is carried out for both high level spatial as well, as frequency characteristics to ensure uniformity in their dimensions across various input videos. In the end the low level and high level characteristics, in both areas (spatial and frequency) are merged using binary GMUs. These GMUs adapt to adjust the mix of low level and high level characteristics to decide the impact each should have in shaping the ultimate depiction. This gating system makes the model to concentrate selectively on abstract information depending on the circumstances of each instance. This step makes sure that the model has a well-rounded understanding of both spatial and frequency features.

3.3.5 Landmark Processing

Landmark characteristics are transformed using a linear layer to align with the necessary feature dimensions before merging them with spatial and frequency features ensuring integration with other modalities, in the model.

3.3.6 Fusion of Spatial, Frequency, and Landmark Features Using Triple GMU

Once the spatial and frequency features have been combined using binary GMUs, the model fuses these features with the landmark features through a Triple Gated Multimodal Unit (Triple GMU). The Triple GMU processes the three feature streams—spatial, frequency, and landmark—by adaptively balancing their contributions. The Triple GMU allows the model to dynamically accentuate the most relevant features from each modality based on the input data, resulting in a comprehensive fused representation. After the fusion, positional encoding is applied to the fused features. Positional encoding introduces temporal information into the feature sequence by encoding the position of each video frame, enabling the model to capture temporal dependencies across frames. This prepares the fused features for subsequent temporal modelling, allowing the network to analyse both spatial and temporal patterns in the video data. The rationale for using GMUs to combine frequency, spatial and landmark features is because they can mix data types dynamically and adjustably. GMUs use gates that figure out how each type of data matters, in a given situation, helping the model highlight important features while diminishing less useful ones. This flexible fusion boosts the

models' ability to gather information from various feature categories, resulting in enhanced performance. In contrast to methods like simple concatenation or cross attention, GMUs achieve a good balance between computational effectiveness and the capacity to depict intricate inter-modal relationships, which positions them as a reliable option, for fusing features across multiple features.

3.3.7 Triple GMU and Temporal Modelling with Transformers

After combining the spatial and frequency domain features using GMUs, the next step involves integrating these features with landmark features via a Triple Gated Multimodal Unit known as Triple GMU. The Triple GMU[13] is vital, for managing the input where spatial aspects capture appearance details frequency features contain information like compression artefacts and landmark attributes encode geometric connections among facial points. The Triple GMU system uses gates to selectively manage how much each input contributes to the overall output. The gates are trained to let the model decide dynamically which aspects are most important, in each situation, and adjust its focus accordingly. This adaptive approach allows the model to respond flexibly to content and changes seen in every video frame. Once the spatial, frequency and landmark characteristics have been merged the model incorporates encoding, into these combined features. Positional encoding[44] plays a role, in providing details about the sequence of frames to the model when using Transformer architectures for temporal modelling, since these architectures don't inherently grasp the sequence order by default. This encoding supplements information that helps the model comprehend how frames are interconnected and identify irregularities characteristic of deepfake videos, like awkward movements or inconsistencies between frames. Once positional encoding is incorporated, the combined features undergo processing, by a Transformer Encoder. The Transformer Encoder[44] is skilled, at understanding connections that exist between frames in a video, which helps it to figure out relationships over time in the video as a whole. This is very important when it comes to spotting deepfake videos because changes, in expressions, movements and lighting can happen across frames and just looking at each frame individually might not catch these subtle differences. By using head attention mechanisms to analyse the sequence of frames, the Transformer Encoder picks up trends over time that suggest any kind of tampering with the video content. After the Transformer completes processing the sequence of data inputted to it, the model gathers information by taking the mean over time [45]. Taking the mean over time involves combining information from all frames within a video into a unified vector representation. This approach is beneficial in video-related tasks, as it captures overall tendencies throughout the entire sequence, rather than focusing on individual frames. By averaging features across all time periods, the model creates a robust representation of the video content, effectively reducing the complexity of the temporal dimension while integrating the information from each frame. This enables the model to understand the coherence of the frames and detect changes that persist or evolve over time, such as subtle inconsistencies or gradual transitions between consecutive frames—key characteristics of deepfake videos.

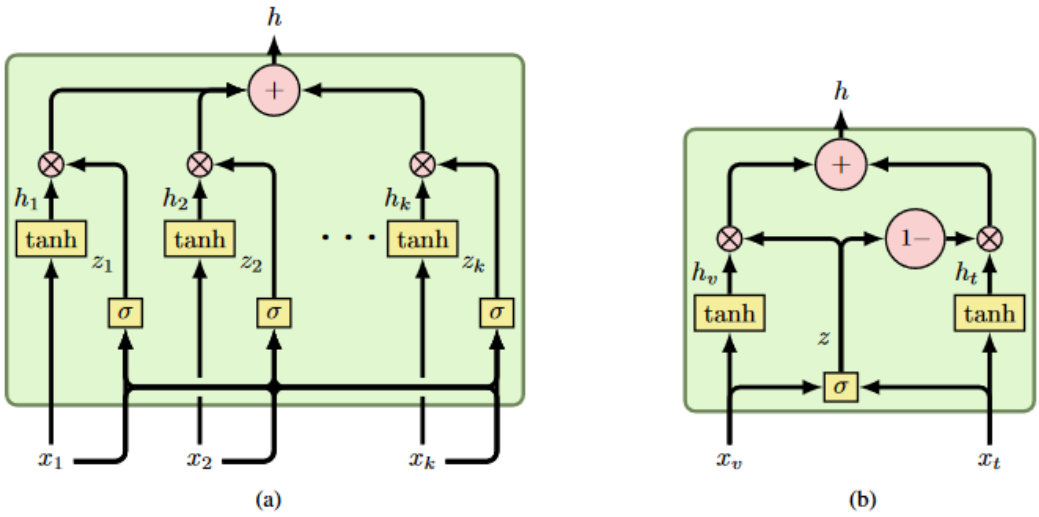


Figure 3.6: Illustration of the GMUs, image extracted from [13]

3.3.8 Final Classification

After the temporal features have been processed and aggregated through the temporal modelling layers, the final step involves passing these features through a fully connected layer for classification. The aggregated feature vector, which represents the entire video, is of the shape $(batch\_size, feature\_dim)$, where each element corresponds to the compressed representation of a video in the batch. The fully connected layer acts as the final classifier, taking the feature vector as input and mapping it to the desired number of output classes (in this case, two classes: real or fake, for deepfake detection).

3.3.8.1 Optimization Algorithm: AdamW

The AdamW (Adaptive Moment Estimation with Weight Decay) optimizer is employed as the primary optimization algorithm for training the model. AdamW is an improvement over the Adam optimizer, incorporating decoupled weight decay regularization to improve generalization and prevent overfitting. It maintains per-parameter learning rates that are adapted based on the first and second moments of the gradients, while the weight decay is applied directly to the weights instead of through the gradients, addressing one of Adam's limitations. The update rules for AdamW are as follows:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (3.1)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (3.2)$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad (3.3)$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad (3.4)$$

$$\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon} - \lambda \theta_t \quad (3.5)$$

Where:

- g_t is the gradient at time step t.
- m_t and v_t are the first and second moment estimates, respectively.
- β_1 and β_2 are hyperparameters controlling the decay rates of these estimates.
- η is the learning rate.
- ε is a small constant to prevent division by zero.
- θ_t represents the model parameters at time step t.
- λ is the weight decay coefficient.

AdamW offers a better balance between effective gradient-based optimization and weight decay, which is critical for preventing overfitting in models that handle data from multiple modalities, such as spatial, frequency, and geometric characteristics. This optimizer enables the model to converge faster while maintaining strong regularization effects on the model weights.

3.3.8.2 Weight Update Mechanism: Gradient Clipping

Gradient clipping is employed to prevent the issue of exploding gradients, which can impair the training process, especially in deep networks with numerous layers. Before updating the weights, the gradients $\|\mathbf{g}\|$ are clipped as follows:

$$\mathbf{g}_{\text{clipped}} = \begin{cases} \frac{\mathbf{g}}{\|\mathbf{g}\|} \times \text{max\_norm}, & \text{if } \|\mathbf{g}\| > \text{max\_norm}, \\ \mathbf{g}, & \text{otherwise.} \end{cases}$$

Where $\|\mathbf{g}\|$ denotes the norm of the gradient vector, and `max_norm` is a predefined threshold. Keeping the model parameters in check by applying gradient clipping helps maintain control over the adjustments and ensures they stay within a range to prevent abrupt shifts that could disrupt the training process of the model.

3.3.8.3 Regularization Techniques: Weight Decay and Dropout

To mitigate overfitting and enhance the generalization capabilities of the model, regularization strategies such as weight decay and dropout are incorporated. Weight Decay (L2 Regularization), the loss function $\mathcal{L}$ is augmented with an L2 penalty on the weights $\boldsymbol{\theta}$:

$$\mathcal{L}_{\text{total}} = \mathcal{L} + \lambda \|\boldsymbol{\theta}\|_2^2$$

Where λ is the weight decay coefficient. Using weight decay aids in ensuring that the model does not overly prioritize any feature by promoting simpler models that are adept at adjusting to new data that was not part of the training set This method of regularization proves particularly beneficial in complex models with numerous parameters as it prevents overfitting to the training data. During training, dropout randomly sets a fraction p of the input units to zero:

$$h_{\text{dropout}} = h \odot \text{Bernoulli}(1 - p)$$

Where h is the input activation, and $\odot$ denotes element-wise multiplication. Dropout serves as a type of learning by preventing neurons from adapting too closely together which helps improve the models resilience and capacity to make generalizations effectively in our model dropout ensures that the network doesn't solely rely on particular features, from the spatial or landmark inputs but rather promotes a well-rounded and adaptable feature representation.

Chapter 4

Implementation

4.1 Datasets

During this investigation, the FaceForensics++[14] and Celeb-DF[15] datasets were used. The FF++ dataset is a collection of both videos and altered versions, with varying compression levels to simulate real world video scenarios. Specifically focusing on the c23 subset, which offers a compressed video representation. This particular group consists of videos that have been altered using the Face Swap method known as Face2Face (F2F) which involves transferring expressions from person to persons, for creating misleading content. A total of 1800 videos were utilized for purposes, during the training phase of this approach while an additional 200 videos were kept aside to validate, confirm its accuracy and effectiveness. To assess how well the model can adapt to scenarios and data sets beyond its training scope and limitations, the Celeb DF dataset was chosen for evaluation purposes, this dataset is well known for its visual fidelity and authentic deepfake alterations. It consists of a mix of 5639 videos and 590 genuine videos making it a demanding standard, for testing generalization abilities. Contrary to the FF++ dataset, the Celeb FD dataset was not utilized in the models training phase. It was instead employed post training to evaluate its performance. In studies that follow a setup as this [8], models trained using FF++ are tested in Celeb DF to assess their ability to handle new deepfake techniques effectively. The experiments of this thesis exclusively utilized the FF++ c23 subset, for training and validation purposes, while the Celeb DF dataset was specifically used for testing generalization.

4.2 Implementation Details

4.2.1 Face Detection Using MTCNN

The Multi-task Cascaded Convolutional Networks (MTCNN) algorithm [38] was utilized for face detection. MTCNN operates in three stages:

1. **Proposal of Candidate Windows**: Generating candidate facial regions through a shallow convolutional network.
2. **Refinement of Bounding Boxes**: Refining the candidate regions to produce accurate bounding boxes using a more complex network.
3. **Facial Landmark Localization**: Predicting facial landmarks such as eyes and mouth positions using a dedicated network.

For each frame in the video, MTCNN detects faces and provides bounding boxes and key points. The bounding box is then adjusted to include a margin around the face, capturing

some surrounding context while excluding unnecessary background. This is achieved by expanding the bounding box by an *expand factor* 1.2 times the original size, The adjusted bounding box dimensions are calculated as follows:

$$\text{new\_width} = \text{width} \times \text{expand\_factor}$$

$$\text{new\_height} = \text{height} \times \text{expand\_factor}$$

$$\text{new\_left} = \max\left(\text{center\_x} - \frac{\text{new\_width}}{2}, 0\right)$$

$$\text{new\_top} = \max\left(\text{center\_y} - \frac{\text{new\_height}}{2}, 0\right)$$

The adjusted bounding box ensures that the face region is accurately captured within the frame's image boundaries. The cropped face is then resized to a fixed size of 256×256 pixels to maintain consistency in input dimensions when feeding into the model. This preprocessing step guarantees uniformity across frames and video segments, enhancing the model's effectiveness through standardized input provision.

4.2.2 Frequency Domain Transformation

Analysing videos in the frequency domain is essential for detecting deepfakes because it reveals subtle alterations that may not be visible in the spatial domain. The Fast Fourier Transform (FFT) [39] was employed to convert video frames from the spatial domain to the frequency domain. The process involves the following steps:

1. **Greyscale Conversion**: Each RGB frame is converted to greyscale to simplify the analysis and focus on intensity variations.

2. **FFT Computation**: The 2D FFT is applied to the greyscale image to obtain the frequency representation:

$$\mathbf{F}_{\text{FFT}}(u, v) = \sum_{x=0}^{H-1} \sum_{y=0}^{W-1} \mathbf{I}(x, y) e^{-j2\pi\left(\frac{ux}{H} + \frac{vy}{W}\right)}$$

   where $\mathbf{I}(x, y)$ is the pixel intensity at coordinates (x, y), and (u, v) are the frequency domain coordinates.

3. **Frequency Shift**: The zero-frequency component is shifted to the centre of the spectrum using `fftshift` for better visualization and analysis.

4. **Magnitude Spectrum Calculation**: The magnitude spectrum is computed to obtain the amplitude of each frequency component:

$$\mathbf{M}(u, v) = |\mathbf{F}_{\text{FFT}}(u, v)|$$

5. **Logarithmic Transformation**: A logarithmic transformation is applied to stabilize the

variations and compress the dynamic range:

$$\mathbf{M}_{\log}(u, v) = \log\left(\mathbf{M}(u, v) + \varepsilon\right)$$

where ε is a small constant to prevent taking the logarithm of zero.

6. **Normalization**: The transformed magnitude spectrum is normalized to the range $[0, 1]$ to ensure uniformity across frames:

$$\mathbf{M}_{\text{norm}}(u, v) = \frac{\mathbf{M}_{\log}(u, v)}{\max_{u,v} \mathbf{M}_{\log}(u, v)}$$

Combining domain frequency characteristics expands the model viewpoint significantly and allows it to detect subtle irregularities, in the frequency spectrum, that are commonly missed in spatial features.

4.2.3 Landmark Feature Extraction

Facial landmarks are specific points on the face that correspond to facial features such as the eyes, nose, mouth, and jawline. These landmarks provide detailed geometric information about the face's structure, which can be indicative of manipulations in deepfake videos. In this implementation, facial landmarks were extracted using `dlib`'s face detection and shape prediction algorithms. The process involves:

1. **Face Detection**: Detecting faces in the frame using a Histogram of Oriented Gradients (HOG)-based detector.

2. **Landmark Detection**: For each detected face, predicting 68 facial landmarks using a pre-trained shape predictor model.

3. **Bounding Box Adjustment**: Similar to the MTCNN process, the bounding box is adjusted, and the face is cropped for consistency.

4. **Feature Calculation**: Computing pairwise Euclidean distances and angles between landmarks to form a feature vector $\mathbf{v}_{\text{landmark}}$.

The Euclidean distance between landmarks i and j is calculated as:

$$d_{ij} = \|\mathbf{L}_i - \mathbf{L}_j\|_2$$

and the angle θ_{ij} between vectors $\mathbf{L}_i - \mathbf{c}$ and $\mathbf{L}_j - \mathbf{c}$ is:

$$\theta_{ij} = \arccos\left(\frac{(\mathbf{L}_i - \mathbf{c})^\top (\mathbf{L}_j - \mathbf{c})}{\|\mathbf{L}_i - \mathbf{c}\|_2 \, \|\mathbf{L}_j - \mathbf{c}\|_2}\right)$$

where $\mathbf{c}$ is the centroid of the landmarks.

4.2.4 Model Architecture

The model processes video data through several stages to capture spatial, frequency, and temporal information, combining these modalities for deepfake detection. Below is a step-by-step breakdown of how data is passed through the model.

4.2.4.1 Step 1: Feature Extraction using EfficientNet Backbones

The model employs two separate EfficientNet-B0 backbones for feature extraction from both the spatial domain (RGB video frames) and frequency domain (FFT-transformed frames).

- Layer 4: Extracts low-level features, which have a shape of $[80, 32, 32]$, representing early patterns such as edges and textures.
- Layer 7: Extracts high-level features, with a shape of $[320, 8, 8]$, representing abstract and complex patterns.

These extracted features from both spatial and frequency domains form the foundation for further processing in the model.

4.2.4.2 Step 2: Triplet Attention Mechanism

The extracted spatial and frequency domain features are passed through Triplet Attention modules, which apply attention sequentially across the height, width, and channel dimensions. This mechanism enhances the model's focus on important regions and patterns in the data.

$$
\begin{aligned}
A_H(F) &= \text{softmax}(F_H) \odot F \\
A_W(F) &= \text{softmax}(F_W) \odot F \\
A_C(F) &= \text{sigmoid}(F_C) \odot F
\end{aligned}
$$

where A_H, A_W, and A_C are the attention mechanisms applied along height, width, and channel dimensions, respectively. The final output is:

$$F' = A_H(F) + A_W(F) + A_C(F)$$

4.2.4.3 Step 3: Feature Reduction

To reduce the dimensionality of the feature maps and maintain computational efficiency, the model applies 1x1 convolutions followed by adaptive pooling.

- 1x1 Convolutions: These reduce the number of channels in the feature maps, compressing the information while preserving spatial resolution. The result is a reduced feature map F':

$$F' = \text{Conv1x1}(F)$$

- Adaptive Pooling: The spatial dimensions of the feature maps are reduced to 1×1 using adaptive average pooling:

$$F'' = \text{AdaptiveAvgPool2D}(F') \in \mathbb{R}^{C \times 1 \times 1}$$

This ensures that the feature maps are prepared for the next stage of multimodal fusion.

4.2.4.4 Step 4: Landmark Feature Processing

In addition to spatial and frequency domain features, the model processes facial landmark features extracted from the video frames. These landmark features are fed into the model as a fixed-size feature vector. To ensure that the landmark features match the dimensionality of other features, they are passed through a linear layer for projection:

$$\mathbf{h}_{\text{landmark}} = \text{Linear}(\mathbf{h}_{\text{landmark}})$$

This projection ensures that the landmark features are aligned in size with spatial and frequency features.

4.2.4.5 Step 5: Multimodal Fusion with GMUs and Triple GMUs

After processing spatial, frequency, and landmark features, the model combines these modalities using Gated Multimodal Units (GMUs). The GMU is designed to integrate two input feature vectors $\mathbf{x}$ and $\mathbf{y}$ (for example, low-level and high-level spatial features). It uses gating mechanisms to control the contribution of each input modality. The equations governing the GMU are as follows:

$$\mathbf{z} = \sigma(\mathbf{W}_z[\mathbf{x}; \mathbf{y}] + \mathbf{b}_z)$$

$$\mathbf{h} = \tanh(\mathbf{W}_h[\mathbf{x}; \mathbf{y}] + \mathbf{b}_h)$$

$$\mathbf{h} = \mathbf{z} \odot \mathbf{h} + (1 - \mathbf{z}) \odot \mathbf{x}$$

Here:

- $\mathbf{z}$ is the gating coefficient vector computed using a sigmoid activation function $\sigma(\cdot)$, which outputs values between 0 and 1.
- $\mathbf{W}_z$ and $\mathbf{b}_z$ are the learnable weights and biases for the gating mechanism.
- $\mathbf{h}$ is the transformed intermediate representation using the tanh activation function, where $\mathbf{W}_h$ and $\mathbf{b}_h$ are the corresponding weights and biases.
- The final output $\mathbf{h}$ is a weighted combination of $\mathbf{h}$ and $\mathbf{x}$, controlled by the gate $\mathbf{z}$. The element-wise product ($\odot$) ensures that $\mathbf{z}$ determines how much each component contributes to the output.

The Triple GMU extends the GMU to integrate three input feature vectors $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ (e.g., spatial, frequency, and landmark features). The gating coefficients for each modality are computed and normalized as follows:

$$\mathbf{z}_1 = \sigma(\mathbf{W}_{z1}[\mathbf{x}_1; \mathbf{x}_2; \mathbf{x}_3] + \mathbf{b}_{z1})$$

$$\mathbf{z}_2 = \sigma(\mathbf{W}_{z2}[\mathbf{x}_1; \mathbf{x}_2; \mathbf{x}_3] + \mathbf{b}_{z2})$$

$$\mathbf{z}_3 = \sigma(\mathbf{W}_{z3}[\mathbf{x}_1; \mathbf{x}_2; \mathbf{x}_3] + \mathbf{b}_{z3})$$

$$\mathbf{h} = \tanh(\mathbf{W}_h[\mathbf{x}_1; \mathbf{x}_2; \mathbf{x}_3] + \mathbf{b}_h)$$

The final fused features $\mathbf{h}$ are computed by combining the three input vectors using the normalized gate values:

$$\mathbf{h} = \mathbf{z}_1 \odot \mathbf{x}_1 + \mathbf{z}_2 \odot \mathbf{x}_2 + \mathbf{z}_3 \odot \mathbf{x}_3$$

4.2.4.6 Step 6: Temporal Modelling using Positional Encoding and Transformer

After the spatial, frequency, and landmark features are fused, the model applies positional encoding to capture temporal dependencies. The positional encoding adds a temporal component to the fused features, which are then passed through a Transformer encoder for temporal modelling.

Positional Encoding Positional encodings $\text{PE}(pos, 2i)$ and $\text{PE}(pos, 2i+1)$ are computed as:

$$\text{PE}(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right), \quad \text{PE}(pos, 2i+1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right)$$

where d_{model} is the dimensionality of the model.

Transformer Encoder The temporally encoded features are then processed using a Transformer encoder with multi-head self-attention, defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where Q, K, and V are the query, key, and value matrices.

4.2.4.7 Step 7: Classification

After temporal modelling, the model aggregates the temporal features by averaging them across the time dimension:

$$\mathbf{h}_{\text{agg}} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{h}_t$$

The aggregated features are then passed through a fully connected layer to produce the final classification logits:

$$\mathbf{o} = \text{Linear}(\mathbf{h}_{\text{agg}})$$

where **o** represents the logits for the real or fake classification.

4.2.5 Training Procedure

4.2.5.1 Loss Function and Optimizer

The model is trained using the Cross-Entropy Loss, defined as:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right]$$

where y_i is the ground truth label and $\hat{y}_i$ is the predicted probability. The AdamW optimizer with weight decay is used for regularization:

$$\theta_{t+1} = \theta_t - \eta\left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\varepsilon} + \lambda\theta_t\right)$$

where η is the learning rate, $\hat{m}_t$ and $\hat{v}_t$ are moment estimates, and λ is the weight decay.

4.2.5.2 Gradient Clipping

For a more steady training, gradients are clipped to prevent explosion:

$$\theta \leftarrow \theta - \eta \cdot \text{clip}(\nabla_\theta \mathcal{L}, \text{max\_norm})$$

where clip limits the norm of gradients to a maximum value.

Chapter 5

Experiments

5.1 Evaluation Metrics

To evaluate the performance of the model, several standard classification metrics are used: accuracy, area under the (Receiver operating characteristic)ROC curve (AUC), and equal error rate (EER).

5.1.1 Accuracy

The accuracy is calculated as the ratio of correctly predicted instances to the total number of instances. It is defined as:

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

where TP, TN, FP, and FN represent true positives, true negatives, false positives, and false negatives, respectively.

5.1.2 Area Under the Curve

The Receiver operating characteristic (ROC) curve is a used measure to assess how well binary classification models perform by illustrating the balance between True Positive Rate (TPR) and False Positive Rate (FPR) across different threshold levels, in classification models. The TPR is the ratio of correctly classified positive instances to the total actual positives, and it is given by:

$$\text{TPR} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Similarly, the FPR is the ratio of incorrectly classified negative instances to the total actual negatives, and it is expressed as:

$$\text{FPR} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$$

The ROC curve plots the TPR (y-axis) against the FPR (x-axis) as the decision threshold varies. The resulting curve shows how well the classifier distinguishes between the positive and negative classes at different thresholds. A random classifier would produce a diagonal ROC curve (with an AUC of 0.5), whereas a perfect classifier would achieve an AUC of 1.0, producing a curve that passes through the upper-left corner (TPR = 1, FPR = 0). The AUC value represents the probability that a randomly chosen positive instance is ranked higher than a randomly chosen negative instance by the classifier. Mathematically, AUC is

calculated as the integral of the TPR with respect to the FPR, which can be written as:

$$\text{AUC} = \int_0^1 \text{TPR}(\text{FPR})\, d(\text{FPR})$$

This integral computes the total area under the ROC curve. The larger the area, the better the model is at distinguishing between the positive and negative classes.

5.1.2.1 Interpretation of AUC

The AUC provides a single scalar value that summarizes the overall performance of a classifier across all possible thresholds:

- An AUC of 0.5 indicates a model with no discrimination ability, equivalent to random guessing.
- An AUC between 0.5 and 0.7 suggests poor to fair discrimination.
- An AUC between 0.7 and 0.9 indicates good discrimination.
- An AUC above 0.9 suggests excellent discrimination.

The AUC metric has several advantages:

- **Threshold Independence**: It provides a summary of model performance over all possible classification thresholds.
- **Class Imbalance Robustness**: AUC is less sensitive to class imbalance compared to accuracy, making it more reliable when one class dominates the other.
- **Interpretability**: AUC directly correlates with the ability of the classifier to rank a positive instance higher than a negative instance, providing an intuitive interpretation.

5.1.3 Equal Error Rate

The Equal Error Rate (EER) commonly employed in classification scenarios like biometric and verification systems emphasizes the importance of balancing false positives and false negatives. It signifies the juncture, on the ROC curve, where the False Positive Rate (FRP) aligns with the False Negative Rate (FRN) offering a measure of classifier performance irrespective of the threshold used – indicating the error rate where false positives match false negatives.
Formally, the EER is defined as the threshold at which:

$$\text{EER} = \text{FPR} \quad \text{where} \quad \text{FPR} = \text{FNR}$$

Where:

$$\text{FPR} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}} \quad \text{and} \quad \text{FNR} = \frac{\text{False Negatives}}{\text{False Negatives} + \text{True Positives}}$$

An ideal system would have low false positives and false negatives at the same time, but often reducing one type of error increases the other. The EER provides a convenient way to summarize the trade-off between these two types of errors in a single value. The lower the EER value, the better the system is at balancing false positives and false negatives, indicating higher performance. Conversely, a high EER value suggests that the system has significant misclassification errors, either falsely accepting negative instances (false positives) or falsely rejecting positive instances (false negatives). On the ROC curve, the EER corresponds to the point where the curve intersects the line defined by FPR(False Positive Rate) = FNR(False Negative Rate). On the EER point of the ROC, both error rates are equal, signifying a balance between these two measures. EER is computed by finding the point where the FPR and FNR values are as close as possible. This is typically implemented by calculating the FPR and FNR at various threshold values and selecting the threshold where the absolute difference between FPR and FNR is minimized:

$$\text{EER} = \min\left(|\text{FPR}(t) - \text{FNR}(t)|\right)$$

Where t is the decision threshold. This approach involves iterating over all possible thresholds, computing the FPR and FNR at each threshold, and identifying the point where the absolute difference between FPR and FNR is smallest.

5.1.4 Balanced Accuracy

Balanced accuracy accounts for class imbalance and is computed as the average of recall for each class:

$$\text{Balanced Accuracy} = \frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP}\right)$$

This metric provides a more robust evaluation in the presence of class imbalance.

5.1.5 Loss Calculation

The loss function used for training is the cross-entropy loss, It calculates the difference between the predicted probability distribution and the true distribution. The formula for cross-entropy loss is given by:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right]$$

where N is the total number of samples, y_i represents the true label, and $\hat{y}_i$ is the predicted probability for the positive class. This loss function penalizes incorrect classifications more heavily, ensuring that the model improves its prediction confidence over time.

5.2 Description of Methods

All these methods were trained using a learning rate of 8×10^{-5}, same as [8], a weight decay value of 1×10^{-3} and a batch size of 2. All the methods were validated on the same 200

samples of FF++.

5.2.1 Baseline Method

The Baseline method serves as the reference model for this experiment. It uses:

- **Number of Frames:** 10 frames per video.
- **Number of Training Samples:** 1800 videos from the FaceForensics++ dataset.
- **Frame Selection:** The Baseline method selects frames from the initial section of each video.

5.2.2 Method B

Method B is the same has Baseline model but used different frames from the same samples to train:

- **Number of Frames:** 10 frames per video.
- **Number of Training Samples:** 1800 videos, similar to the Baseline.
- **Frame Selection:** Unlike the Baseline, Method B selects frames from the middle portion of each video. This change in frame selection is expected to include more variability frames.

This approach investigates that is not only the quantity of the frames but their "quality", some frames could be more beneficial than other for predictions.

5.2.3 Method C

Method C focuses on reducing the number of training samples to explore the model's performance under limited data:

- **Number of Frames:** 10 frames per video.
- **Number of Training Samples:** 600 videos, significantly fewer than the Baseline and Method B.
- **Frame Selection:** Similar to the Baseline, Method C selects initial frames from the video.

The aim of Method C is to investigate the impact of reducing training samples on the model's learning and generalization capabilities, in comparison, to the Baseline model.

5.2.4 Method D

Method D reduces the number of frames per video, introducing a significant change in how much temporal information the model can access:

- **Number of Frames:** 3 frames per video.
- **Number of Training Samples:** 1800 videos, the same as the Baseline.
- **Frame Selection:**Method D also chooses frames from the beginning of the video; however, since there are frames involved, it captures a smaller amount of changes over time.

This approach aims to assess the effects of decreasing data while maintaining a number of training examples.

5.2.5 Method E

Method E further reduces both the number of frames and the number of training samples:

- **Number of Frames:** 3 frames per video.
- **Number of Training Samples:** 600 videos, the same as Method C.
- **Frame Selection:** Frames are selected from the initial part of the video.

Method E assesses the models' ability to learn and adapt in challenging situations, with very few frames and training data inputs.

5.2.6 Summary of Method Differences

The methods in this experiment vary across the following key dimensions:

- **Number of Frames:** Baseline, Method B, and Method C use 10 frames, while Method D and Method E use only 3 frames.
- **Number of Training Samples:** Baseline, Method B, and Method D use 1800 videos, while Method C and Method E use only 600 videos.
- **Frame Selection:** Baseline, Method C, Method D and Method E select frames from the initial part of the video, while Method B selects frames from the middle.

These different scenarios offer understanding on how the quantity of frames used for training samples and the strategies, in selecting frames, influence the model's effectiveness and ability to generalize.

Table 5.1: Training, Validation (FF++) over 10 epochs, and Test (Celeb-DF) results

Method	Frames	T.Samples	FF++ Train		FF++ Val		Celeb-DF Test			
			Acc (%)	AUC	Acc (%)	AUC	Acc (%)	Bal Acc (%)	AUC	Loss
Baseline	10	1800	98.78	0.9878	99.5	0.9950	25.65	55.30	0.6114	4.9225
Method B	10	1800	98.89	0.9889	97.5	0.9750	11.38	49.99	0.5468	8.7850
Method C	10	600	96.17	0.9492	93.5	0.9350	26.18	55.06	0.6273	8.4146
Method D	3	1800	96.94	0.9694	98.0	0.9800	14.48	51.17	0.5136	6.7366
Method E	3	600	92.67	0.8975	93.0	0.9300	14.71	51.45	0.5506	5.9235

5.3 Key Insights and Analysis

5.3.1 High Training and Validation Performance on FF++

The model shows high accuracy and AUC scores on both the training and validation sets of the FF++ dataset even across various modifications, to the baseline approach. As shown in Table 5.1, the baseline method reaches a training accuracy of 98.78% and a validation accuracy of 99.5%. Method C, which uses less data, still manages a training accuracy of 96.17% and a validation accuracy of 93.5%. This demonstrates the model is successfully picking up on the patterns and traits associated with the FF++ dataset; this implies its ability to differentiate between authentic and altered content in this dataset.

5.3.2 Impact of Number of Frames and Training Videos

The performance of the model is greatly influenced by the quantity of frames and training videos used. As portrayed in Table 5.1, the baseline method, with 10 frames and 1800 videos performs better, than Method D that utilizes 3 frames but with an equal number of videos, in both training (98.78% vs. 96.94%) and validation accuracy (99.5% vs. 98.0%). The test accuracy on the Celeb-DF dataset also highlights this disparity (25.65% vs. 14.48%), indicating that having a higher number of frames results in improved generalization abilities in the model's performance. When looking at Method C and Method E side, by side, it is evident that Method C outperforms Method E in all aspects despite both methods using the same number of videos for training (600). This indicates that providing the model with a higher number of frames allows it to accommodate a wider array of variations and enhances its ability to comprehend intricate patterns found within the training dataset.

5.3.3 Effect of Increasing Training Samples

Increasing the number of training videos also enhances the model's learning capabilities. Table 5.1 compares the baseline method (using 1800 videos) with Method C (using 600 videos), while keeping the number of frames constant (10 frames). The results demonstrate that the baseline method achieves higher training (98.78%) and validation accuracy (99.5%) compared to Method C (96.17% and 93.5%, respectively). Interestingly, despite the differences in training and validation performance, both methods show similar accuracy on the Celeb-DF test dataset (25.65% for the baseline vs. 26.18% for Method C), indicating that although more training data improves the model's learning on the same dataset, its cross-dataset generalization may still be limited. This suggests that increasing the number of training samples

improves performance within the same dataset, but does not necessarily translate to higher generalizability.

5.3.4 Frame Selection Affects Performance

The choice of which frames to select also plays a crucial role in determining the model's effectiveness. Table 5.1 shows that the baseline method, which uses initial frames, performs better than Method B, which uses middle frames, in terms of validation accuracy (99.5% vs. 97.5%) and Celeb-DF test accuracy (25.65% vs. 11.38%), despite both having comparable training accuracy (98.78% vs. 98.89%). Furthermore, when comparing Method D (using fewer initial frames) with Method B (using middle frames), Method D shows a higher Celeb-DF test accuracy (14.48% vs. 11.38%). This pattern persists when comparing Method E to Method B, suggesting that selecting frames based on consistent facial features or reduced blurriness holds greater significance than simply adding more frames or training samples in the process. These results underscore the importance of focusing on the quality and appropriateness of chosen frames, as it promotes enhanced learning outcomes and improved performance. To underscore the significance of frame quality, even more, The Baseline and Method B were evaluated using another group of 10 frames from the Celeb-DF dataset. The outcomes varied considerably, as shown in Table 5.2. This study shows that choosing the frames is crucial not just for training but also, for ensuring the precision of deepfake detection models. **Baseline Method**: When the new frames were tested with, the Baseline method applied to them resulted in an accuracy of 90.53% and a reduced loss of 0.4991. The significant enhancement compared to the test results (which stood at 25.65%) highlights that certain frames offer clearer indications for identifying manipulations. However, the decline in AUC to 0.4851 and the model's accuracy of 50% suggests that it performs poorly on Celeb-DF. Highlighting its inconsistency **Method B**: On the other hand, Method B's performance dropped drastically, with an accuracy of only 10.26%, an accuracy of 49.37%, and a significantly higher loss of 1.1570. Despite Method B's slight improvement in AUC (0.5086) compared to Baseline, indicating a marginally more balanced separation of classes, but still a poor result, its very low accuracy and high loss show that it also struggles with the new frame set. This study underscores the importance of choosing the frames for spotting deepfakes effectively. Certain frames display signs or visual distortions that help in identifying alterations, while others may have interference or obstructions that make it harder to classify accurately and end up just adding noise. Just adding frames or training data doesn't automatically lead to improved results, but it could offer a wider range of valuable and diverse information to the model, thus improving its capacity to adapt and apply knowledge across different scenarios. To achieve outcomes in model training, it is essential to mixture frame quality and frame quantity. This strategy exposes the model to an array of frames, ranging from obvious forgeries, to subtly manipulated ones.

5.3.5 Top 5 Most Challenging Samples from Celeb-DF

This subsection presents the top 5 most challenging samples from the Celeb-DF dataset. These are the samples where the Baseline model was most confident in its prediction, yet the

Table 5.2: Performance comparison of Baseline and Method B on Celeb-DF with 10 different frames. Metrics: Accuracy, Balanced Accuracy, AUC, and Loss.

Method	Accuracy (%)	Balanced Accuracy (%)	AUC	Loss
Baseline	90.53	50.00	0.4851	0.4991
Method B	10.26	49.37	0.5086	1.1570

prediction was incorrect. The following images illustrate the hardest samples, ranked from top 1 to top 5 based on their prediction confidence. Each caption specifies the predicted label, actual label, and the model's confidence score for these challenging cases.

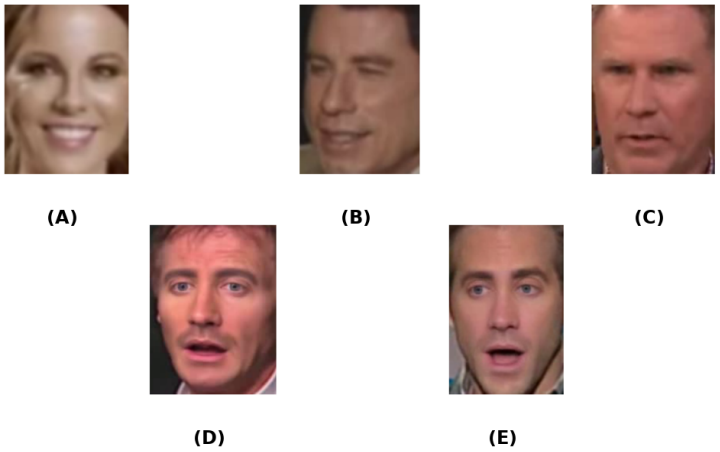


Figure 5.1: (A) Predicted: Fake, Actual: Real, Confidence: 99.9%, (B) Predicted: Fake, Actual: Real, Confidence: 99.9%, (C) Predicted: Fake, Actual: Real, Confidence: 99.9%, (D) Predicted: Real, Actual: Fake, Confidence: 99.9%, (E) Predicted: Real, Actual: Fake, Confidence: 99.9%

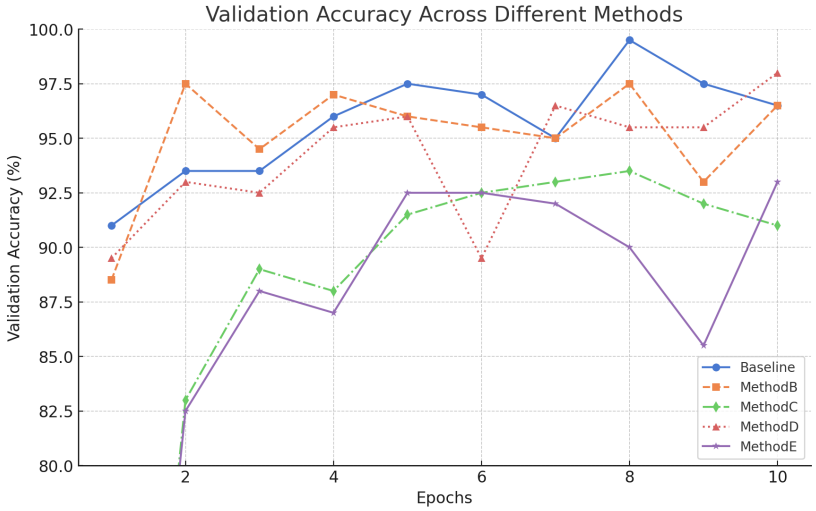
5.3.6 Graph Analysis: Validation Accuracy and Loss

5.3.6.1 Consistency and Stability of Baseline and Method B

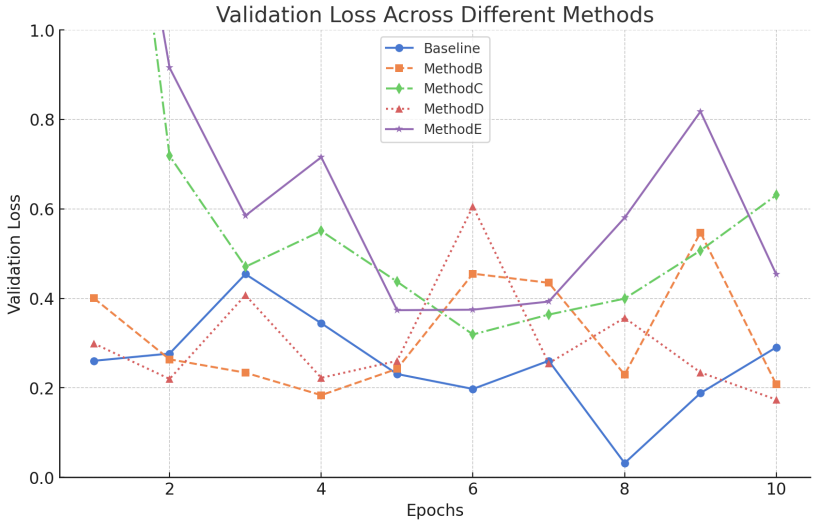
The validation accuracy and loss graphs show that both the Baseline and Method B perform consistently well throughout the epochs with curves indicating strong learning abilities and stability during training with minimal fluctuations, in performance. Their smooth progression indicates a grasp of the data and successful reduction of validation loss.

5.3.6.2 Variability in Methods C, D, and E

Methods C, D and E are of all over the place when it comes to their validation accuracy and loss numbers compared to the Baseline method which seems more consistent in comparison. One thing that stands out is that both Method C and Method D have some ups and downs in their loss graphs around the 4th, to 6th epochs, which could mean they had some trouble staying steady during training sessions. Seeing these spikes in loss might indicate that these methods don't handle the validation data well or struggle to reach a point during training sessions. This lack of stability could be because the training data doesn't cover an enough range, or maybe there weren't enough frames used in the process.



(a) Validation Accuracy Across Different Methods



(b) Validation Loss Across Different Methods

Figure 5.2: Comparison of Validation Accuracy and Loss Over Epochs for Different Methods

5.3.6.3 Convergence Speed

Analysing the accuracy graphs reveals that the Baseline and Method B reach high validation accuracy rapidly around epoch 2 and maintain this performance throughout the training process. This rapid convergence implies that these models efficiently capture the critical features of the dataset. In comparison, Method C starts with a lower accuracy and takes longer to reach comparable performance levels, while Methods D and E lag behind in convergence speed and overall accuracy.

5.3.6.4 Challenges Faced by Method E

Method E consistently shows the worst performance in terms of both validation accuracy and loss graphs compared to other methods tested in the study report. The validation accuracy of Method E remains notably lower than that of the methods examined, while its loss values are consistently higher and more fluctuating during training sessions. These observations

Table 5.3: Comparison of the generalization ability on FF++ (c23) and CelebDF using AUC. Table adapted from [8]

Method	**FF++ (c23)**	**CelebDF**
Two-stream [46]	0.701	0.538
Meso4 [5]	0.847	0.548
Meso4Inception4 [5]	0.830	0.548
HeadPose [9]	0.473	0.546
FWA [47]	0.801	0.569
VA-MLP [17]	0.664	0.550
Xception-raw [37]	0.997	0.482
Xception-c23 [37]	0.997	0.653
Xception-c40 [37]	0.996	0.655
Multi-task [48]	0.763	0.543
Capsule [49]	0.966	0.575
DSP-FWA [47]	0.930	0.646
Face-XRay [50]	0.991	0.742
F3Net [51]	0.981	0.652
Two-Branch [46]	0.932	0.734
Efficient-B4 [43]	0.997	0.643
SPSL [21]	0.969	0.769
MD-CSND [52]	0.995	0.688
STIL [53]	0.971	0.756
CFFs [54]	0.976	0.742
MGL [8]	0.995	0.888
Baseline Method	0.995	0.611

suggest that Method E encounters difficulties in learning and generalizing information, as evidenced by its convergence rate and inability to effectively decrease validation loss. The scarcity of frames and training videos available for Method E is likely a contributing factor, to these observed challenges.

5.3.6.5 Instability in Method D

Although Method D outperforms Method E in terms of performance overall. It shows significant fluctuations in validation loss between epochs 4 to 6 which hints at potential challenges with maintaining consistent learning when using fewer frames, in the model training process compared to the Baseline and Method B which exhibit more stability and better performance overall.

5.3.7 Model Performance Limited by Data Diversity and Computational Constraints

The model's high accuracy on the FF++ training and validation sets demonstrates its effective learning mechanism, capable of capturing complex patterns when provided with sufficient and representative data. However, the consistently low performance across all methods on the Celeb-DF test set indicates poor generalization, likely due to significant domain differences between FF++ and Celeb-DF. These differences include variations in video quality, compression irregularities, lighting conditions, facial expressions, and deepfake generation techniques. Training exclusively on FF++ limits the model's exposure to the variety of manipulations present in Celeb-DF, causing it to struggle with unseen types of manipulations

Table 5.4: Comparison of the performance of different methods on FF++ (c23) using AUC, ACC (%), and the number of frames. Adapted from [8]

Method	**Frames**	**AUC**	**ACC (%)**
Steg.Features [55]	270	-	70.9
LD-CNN [56]	270	-	78.5
Face X-ray [50]	270	0.874	-
Cozzolino et al. [56]	270	-	78.5
Bayer & Stamm [57]	270	-	83.0
Rahmouni et al. [58]	270	-	79.1
Xcep-ELA [59]	270	0.948	93.9
Xcep-PAFilters [60]	270	0.902	87.2
Two-Branch [46]	270	0.991	96.9
Efficient-B4 [43]	270	0.992	96.6
SPSL [21]	100	0.953	91.5
MD-CSND [52]	270	0.993	97.3
MGL [8]	32	0.995	97.8
Baseline Method	10	0.995	99.5

Table 5.5: Comparison of Feature Fusion Methods at Frame Level using AUC scores on FF++ and CelebDF. Adapted from [8]

Method	**FF++ (AUC)**	**CelebDF (AUC)**
Linear	0.969	0.834
M2TR	0.991	0.851
SFF Module	0.995	0.888
GMUs	0.995	0.611

and variations. Computational constraints have also impacted performance by necessitating training on fewer frames and videos, thus limiting data diversity and quantity. With enhanced computational resources, the model could process more data—such as frames from entire videos—increasing exposure to diverse examples and improving generalization to different datasets like Celeb-DF. Therefore, while the architecture is effective, its potential is not fully realized due to limitations in data diversity and computational capacity affecting its ability to generalize.

Table 5.6: Performance comparison of different methods on FF++ (c23). Metrics: AUC (higher is better), ACC (higher is better), EER (lower is better). Adapted from [8]

Method	**AUC**	**ACC (%)**	**EER (%)**
HeadPose [9]	0.568	56.81	43.79
FDFClassifier [61]	0.492	49.22	50.53
Xception [22]	0.993	97.04	2.70
MesoNet [5]	0.601	56.33	43.66
Meso-Incep [5]	0.904	81.72	17.30
CapsuleNet [49]	0.984	94.49	5.71
CNN-RNN [62]	0.984	94.86	5.36
F3Net [51]	0.990	97.94	1.99
MGL [8]	0.996	98.13	1.79
Baseline Method	0.995	99.5	1.01

5.3.8 Experiments Conclusion

The results from the experiment demonstrates that the model's structure is quite successful in grasping the detection of deepfake videos, in the FF++ dataset. The strong accuracies achieved during training and validation showcase the model's ability to understand patterns and characteristics linked to identifying deepfake content.
The poor generalization to the Celeb-DF dataset appears to originate from:

- **Limited Diversity in Training Data:** Training solely on FF++ does not equip the model to handle the variations present in other datasets.
- **Insufficient Number of Frames and Videos:** Fewer frames and videos reduce the model's exposure to different scenarios and anomalies.
- **Computational Constraints:** These limitations restrict the amount of data the model can process and the complexity of models that can be trained.

5.3.9 Comparison with Other Models

In this section of the study, it is assessed how well the Baseline Method stacks up against other deepfake detection models. It examines its effectiveness on the FF++ and measures its adaptability on CelebDF. Table 5.6 compares the performances among techniques on the FF++. The initial method sets a standard, with exceptional precision and a remarkably low EER score that highlights its efficiency in identifying deepfakes. Table 5.4 shows the performance of techniques when using varying frame counts, in FF++ specifically looking at AUC and ACC measurements. The Baseline Method demonstrates precision even with a reduced number of frames, revealing its effectiveness. Table 5.3 presents a comparison of how various methods generalize from the FF++ to CelebDF datasets based on the AUC metric. While the Baseline Method demonstrates a good performance on FF++ it exhibits a decline in effectiveness when applied to the CelebDF dataset. Finally, Table 5.5 compares various feature fusion methods on the frame level using AUC for both FF++ (c23) and CelebDF datasets. Advanced methods like M2TR and MGL outperform the Baseline Method in terms of generalization capability.

5.4 Conclusion of Experiments

The tests show that the system can accurately identify videos in the FaceForensics++ (FF++) dataset with consistent accuracy during training and validation phases. It successfully detects the characteristics of deepfake content, as shown by its high AUC and low Equal Error Rate (EER). However, its ability to perform well on the Celeb DF dataset highlights difficulties in adapting to different datasets due to variations, in data. The analysis emphasizes that having a high amount and quality of data is important for improving performance in models used for training with more frames and videos (like the Baseline approach) which usually perform better than those trained with fewer resources (like Method E). The

selection of frames is also crucial, the Baseline approach that chooses frames tends to outperform other methods that use different frame selection techniques. Lastly limitations, in computing capacity limit the variety of training data available which affects how well the model can generalize effectively.

Chapter 6

Conclusion

This research investigated into methods for identifying deepfakes by creating and utilizing a multi featured model that incorporates spatial details as well as frequency and landmark characteristics. The structure displayed results with the FF++ dataset by attaining high accuracy during training and validation phases. Nonetheless, it encountered difficulties, in applying its findings to the Celeb DF dataset, revealing shortcomings in its ability to perform consistently across different datasets. The experiment's main discoveries show that the system can grasp patterns unique to the training set (FF++) but struggles with generalizing effectively across different datasets – highlighting a notable challenge. To bridge the performance disparities between datasets and enhance model adaptability to scenarios like facial details and generation techniques within deepfakes, there is a need, for broader dataset diversity and targeted training approaches. Despite facing these obstacles, the work offered perspectives on choosing frames combining features and the significance of varied data. These factors aid in creating detection models for deepfake content. In essence, this research highlights the importance of refining strategies, for generalization and incorporating datasets to enhance performance across various scenarios.

6.1 Contributions and Achievements

6.1.1 Development of a Multimodal Deepfake Detection Model

This research introduces a method to spot deepfake videos by blending frequency features, with landmark traits using advanced fusion techniques in a detection model design that incorporates GMUs and Triplet Attention mechanisms to pinpoint spatial and temporal anomalies effectively in video content. This approach significantly enhances the model's capacity to identify deepfake videos.

6.1.2 Evaluation Across Multiple Datasets

The model was tested on both the FF++ and Celeb DF datasets. Showed almost perfect accuracy in the FF++ dataset compared to several leading models. Either outperform them or achieving similar results. However, the noticeable decrease in performance, on the Celeb DF dataset, emphasized the model's limitations in terms of generalizing its capabilities. These findings emphasize the need to include a range of datasets during training to enhance the model's adaptability and reliability.

6.1.3 Insights on Frame Selection and Data Quantity

This work offered perspectives on how choosing specific frames and the quantity of training data influence the performance of a model. According to the findings, models trained using frames consistently outperformed those trained with middle frames, highlighting the importance of selecting high quality and consistent frames for successful detection. Moreover, expanding the pool of training videos and frames enabled the model to grasp a range of patterns, thereby enhancing its overall effectiveness.

6.1.4 Comparative Analysis of Feature Fusion Techniques

Through comparing fusion techniques in the studies, investigation revealed how GMUs and Triplet Attention improved the integration of multimodal features effectively. Additionally, the utilization of these methods enhanced the model's capacity for capturing a range of information resulted in better detection performance, on challenging datasets.

6.1.5 Practical Contributions to Deepfake Detection Research

The study made contributions to enhancing the creation of upcoming deepfake detection models by creatively combining various features and utilizing advanced fusion methods to demonstrate the promise of constructing more precise and effective detection systems. It also sets a foundation, for enhancing model versatility through highlighting the significance of using diverse training datasets and conducting thorough temporal analysis.

These contributions collectively advance the state of the art in deepfake detection, offering a framework that can be further refined to achieve greater accuracy and cross-dataset applicability.

Bibliography

[1] L. Weng, "What are diffusion models?" 2021. [Online]. Available: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/ xv, 3, 4, 5

[2] A. Aggarwal, M. Mittal, and G. Battineni, "Generative adversarial network: An overview of theory and applications," *International Journal of Information Management Data Insights*, 2021. xv, 4

[3] Towards Data Science, "Applied deep learning - part 3: Autoencoders," 2018. [Online]. Available: https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798 xv, 5

[4] A. Zucconi, "Understanding the technology behind deepfakes," 2018. [Online]. Available: https://www.alanzucconi.com/2018/03/14/understanding-the-technology-behind-deepfakes/ xv, 6, 7

[5] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "Mesonet: A compact facial video forgery detection network," *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2018. xv, 7, 8, 9, 10, 58, 59

[6] T. Baltrusaitis, A. Zadeh, Y. Lim, and L.-P. Morency, "Openface 2.0: Facial behavior analysis toolkit," *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, 2018. xv, 10

[7] D.-C. Stanciu and B. Ionescu, "Deepfake video detection with facial features and long-short term memory deep networks," *IEEE Access*, 2021. xv, 9, 10

[8] Z. Yan, P. Sun, Y. Lang, S. Du, S. Zhang, W. Wang, and L. Liu, "Multimodal graph learning for deepfake detection," *Journal of Imaging*, 2023. xv, xvii, 11, 14, 15, 16, 18, 32, 41, 51, 58, 59

[9] X. Yang, Y. Li, and S. Lyu, "Exposing deep fakes using inconsistent head poses," *ICASSP IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019. xv, 11, 12, 58, 59

[10] M. Li, B. Liu, Y. Hu, L. Zhang, and S. Wang, "Deepfake detection using robust spatial and temporal features from facial landmarks," *IEEE International Workshop on Biometrics and Forensics (IWBF)*, 2021. xv, 11, 12, 13, 14

[11] P. Korshunov and S. Marcel, "Deepfakes: a new threat to face recognition? assessment and detection," Idiap Research Report, 2018. xv, 13

[12] D. Misra, T. Nalamada, A. U. Arasanipalai, and Q. Hou, "Rotate to attend: Convolutional triplet attention module," *arXiv preprint arXiv:2010.03045*, 2020. xvi, 19, 21, 35

[13] J. Arevalo, T. Solorio, M. M. y Gómez, and F. A. González, "Gated multimodal units for information fusion," *arXiv preprint arXiv:1702.01992*, 2017. xvi, 21, 22, 36, 37, 38

[14] A. Rössler *et al.*, "Faceforensics: A large-scale video dataset for forgery detection in human faces," 2018. [Online]. Available: https://arxiv.org/abs/1803.09179 xvi, 2, 23, 41

[15] Y. Li, X. Yang, S. Sun, and S. Li, "Celeb-df-v2: A new dataset for deepfake forensics," 2019. [Online]. Available: https://github.com/yuezunli/celeb-deepfakeforensics/blob/master/Celeb-DF-v2/demo.png xvi, 2, 24, 41

[16] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-df: A large-scale challenging dataset for deepfake forensics," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 6, 7, 24

[17] F. Matern, C. Riess, and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," *IEEE Winter Applications of Computer Vision Workshops (WACVW)*, 2019. 6, 58

[18] C. Li, L. Wang, S. Ji, X. Zhang, Z. Xi, S. Guo, and T. Wang, "Seeing is living? rethinking the security of facial liveness verification in the deepfake era," *arXiv preprint arXiv:2201.12345*, 2022. 7, 8

[19] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection," *Information Fusion*, 2020. 7

[20] Z. Gu, Y. Chen, T. Yao, S. Ding, J. Li, F. Huang, and L. Ma, "Spatiotemporal inconsistency learning for deepfake video detection," *Proceedings of the 29th ACM International Conference on Multimedia*, 2021. 7

[21] H. Liu, X. Li, W. Zhou, Y. Chen, Y. He, H. Xue, W. Zhang, and N. Yu, "Spatial-phase shallow learning: Rethinking face forgery detection in frequency domain," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 7, 58, 59

[22] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," 2017. 8, 11, 59

[23] M. Kim, S. Tariq, and S. S. Woo, "Cored: Generalizing fake media detection with continual representation using distillation," *Proceedings of the 29th ACM International Conference on Multimedia*, 2021. 8

[24] D. Cozzolino, J. Thies, A. Rossler, C. Riess, M. Nießner, and L. Verdoliva, "Forensictransfer: Weakly-supervised domain adaptation for forgery detection," *arXiv preprint arXiv:1812.02510*, 2018. 8

[25] S. Lee, S. Tariq, J. Kim, and S. S. Woo, "Tar: Generalized forensic framework to detect deepfakes using weakly supervised learning," *IFIP International Conference on ICT Systems Security and Privacy Protection*, 2021. 8

[26] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015. 8

[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, 2014. 8

[28] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 9

[29] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015. 9

[30] W. Shi, F. Jiang, and D. Zhao, "Single image super-resolution with dilated convolution based multi-scale information learning inception module," *arXiv preprint arXiv:1707.07128*, 2017. 9

[31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 11

[32] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," *IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 11, 15

[33] H. Qi, Q. Guo, F. Juefei-Xu, X. Xie, L. Ma, W. Feng, Y. Liu, and J. Zhao, "Deeprhythm: Exposing deepfakes with attentional visual heartbeat rhythms," *Proceedings of the 28th ACM International Conference on Multimedia*, 2020. 15

[34] X. Li, Y. Lang, Y. Chen, X. Mao, Y. He, S. Wang, H. Xue, and Q. Lu, "Sharp multiple instance learning for deepfake video detection," *Proceedings of the 28th ACM International Conference on Multimedia*, 2020. 15

[35] Y. Qian, G. Yin, L. Sheng, Z. Chen, and J. Shao, "Thinking in frequency: Face forgery detection by mining frequency-aware clues," *European Conference on Computer Vision (ECCV)*, 2020. 16

[36] J. Wang, Z. Wu, J. Chen, and Y.-G. Jiang, "M2tr: Multi-modal multi-scale transformers for deepfake detection," *arXiv preprint arXiv:2104.09770*, 2021. 16

[37] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "Faceforensics++: Learning to detect manipulated facial images," *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 23, 58

[38] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multi-task cascaded convolutional networks," *IEEE Signal Processing Letters*, 2016. 28, 41

[39] V. Nair, M. Chatterjee, N. Tavakoli, A. S. Namin, and C. Snoeyink, "Fast fourier transformation for optimizing convolutional neural networks in object recognition," arXiv preprint arXiv:2010.04257, 2020. 28, 42

[40] B. Wang, X. Wu, Y. Tang, Y. Ma, Z. Shan, and F. Wei, "Frequency domain filtered residual network for deepfake detection," *Mathematics*, 2023. 28

[41] M. Jin, "A study of face alignment methods in unmasked and masked face recognition," 2023. [Online]. Available: http://uu.diva-portal.org/smash/get/diva2:1805927/FULLTEXT01.pdf 32

[42] N. Jain, S. Borade, B. Patel, V. Kumar, M. Godhrawala, S. Kolaskar, Y. Nagare, P. Shah, and J. Shah, "Deepfake detection using efficientnetb7: Efficacy, efficiency, and adaptability," *Electronics*, 2023. 33

[43] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *International Conference on Machine Learning*, 2019. 33, 58, 59

[44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017. 37

[45] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. V. Gool, "Temporal segment networks for action recognition in videos," *arXiv preprint arXiv:1705.02953*, 2017. 37

[46] I. Masi, A. Killekar, R. M. Mascarenhas, S. P. Gurudatt, and W. AbdAlmageed, "Two-branch recurrent network for isolating deepfakes in videos," *European Conference on Computer Vision (ECCV)*, 2020. 58, 59

[47] Y. Li and S. Lyu, "Exposing deepfake videos by detecting face warping artifacts," *arXiv preprint arXiv:1811.00656*, 2018. 58

[48] G. Lu, X. Zhao, J. Yin, W. Yang, and B. Li, "Multi-task learning using variational auto-encoder for sentiment classification," *Pattern Recognition Letters*, 2020. 58

[49] H. H. Nguyen, F. Pasquini, J. Yamagishi, and I. Echizen, "Multi-task learning for detecting and segmenting manipulated facial images and videos," *IEEE 10th International Conference on Biometrics Theory Applications and Systems (BTAS)*, 2019. 58, 59

[50] L. Li, J. Bao, T. Zhang, H. Yang, D. Chen, F. Wen, and B. Guo, "Face x-ray for more general face forgery detection," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 58, 59

[51] Y. Qian, G. Yin, L. Sheng, Z. Chen, and J. Shao, "Thinking in frequency: Face forgery detection by mining frequency-aware clues," *European Conference on Computer Vision (ECCV)*, 2020. 58, 59

[52] A. Agarwal, A. Agarwal, S. Sinha, M. Vatsa, and R. Singh, "Md-csdnet: Multi-domain cross-stitched network for deepfake detection," *16th IEEE International Conference on Automatic Face and Gesture Recognition*, 2021. 58, 59

[53] Z. Gu, Y. Chen, T. Yao, S. Ding, J. Li, F. Huang, and L. Ma, "Spatiotemporal inconsistency learning for deepfake video detection," *Proceedings of the 29th ACM International Conference on Multimedia*, 2021. 58

[54] P. Yu, J. Fei, Z. Xia, Z. Zhou, and J. Weng, "Improving generalization by commonality learning in face forgery detection," *IEEE Transactions on Information Forensics and Security*, 2022. 58

[55] J. Fridrich and J. Kodovský, "Rich models for steganalysis of digital images," *IEEE Transactions on Information Forensics and Security*, 2012. 59

[56] D. Cozzolino, G. Poggi, and L. Verdoliva, "Recasting residual-based local descriptors as convolutional neural networks: An application to image forgery detection," *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, 2017. 59

[57] B. Bayar and M. C. Stamm, "A deep learning approach to universal image manipulation detection using a new convolutional layer," *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, 2016. 59

[58] N. Rahmouni, V. Nozick, J. Yamagishi, and I. Echizen, "Distinguishing computer graphics from natural images using convolutional neural networks," *IEEE Workshop on Information Forensics and Security (WIFS)*, 2017. 59

[59] T. S. Gunawan, S. A. M. Hanafiah, M. Kartiwi, N. Ismail, N. F. Za'bah, and A. N. Nordin, "Development of photo forensics algorithm by detecting photoshop manipulation using error level analysis," *Indonesian Journal of Electrical Engineering and Computer Science*, 2017. 59

[60] M. Chen, V. Sedighi, M. Boroumand, and J. Fridrich, "Jpeg-phase-aware convolutional neural network for steganalysis of jpeg images," *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, 2017. 59

[61] B. M. Le and S. S. Woo, "Exploring the asynchronous of the frequency spectra of gan-generated facial images," *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2022. 59

[62] D. Güera and E. J. Delp, "Deepfake video detection using recurrent neural networks," *15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2018. 59