Thesis Final Report

Enhancing Cross-Domain Deepfake Detection through an Xception-Based Multi-Branch Model: Challenges, Insights and the VeriFake Dataset

by

Zarin Syara Eqra
23101552
Neloy Kumer Saha
21201556
Tayeba Rounak Karim
23141029
Tafsirul Hoque
21201517
Faria Naz Tabassum
21201815

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

Department of Computer Science and Engineering
BRAC University
August 2025

Declaration

It is hereby declared that

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

Student's Full Name & Signature:

Zarin Syara Eqra
23101552

Neloy Kumer Saha
21201556

Tayeba Rounak Karim
23141029

Tafsirul Hoque
21201517

Faria Naz Tabassum
21201815

Approval

The thesis/project titled "Enhancing Cross-Domain Deepfake Detection through an Xception-Based Multi-Branch Model: Challenges, Insights and the VeriFake Dataset" submitted by

1. Zarin Syara Eqra (23101552)
2. Neloy Kumer Saha (21201556)
3. Tayeba Rounak Karim (23141029)
4. Tafsirul Hoque (21201517)
5. Faria Naz Tabassum (21201815)

Of Summer, 2025 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on August 09, 2025.

Examining Committee:

Supervisor:
(Member)

Dr. Muhammad Iqbal Hossain
Associate Professor
Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

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

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

Deepfake technology has advanced rapidly, producing highly realistic synthetic videos that can evade state-of-the-art detectors. However, most models trained on curated benchmarks fail to generalize to unseen datasets. This work examines the causes of such failures and explores solutions, introducing VeriFake, a custom dataset of 500 real and 500 fake videos generated using Roop and FaceFusion to reflect modern high-quality manipulations. We trained CNN and Transformer-based models on various combinations of VeriFake, Celeb-DF, and FaceForensics++ (FF++), with and without domain adaptation techniques such as Gradient Reversal Layers (GRL) and heuristic features. While VeriFake-trained models excelled in-domain, they generalized poorly to DFDC and Celeb-DF due to reliance on dataset-specific artifacts. To address this, we developed a multi-branch Xception-based framework with disentangled feature learning, a reconstruction decoder, and a GRL-powered domain-adversarial module and trained on FF++, achieving 0.7509 AUC on DFDC and 0.8183 AUC on Celeb-DF, surpassing results from existing works, though with a trade-off on VeriFake (0.7054 AUC). These findings highlight the importance of diverse, well-designed benchmarks and domain-invariant features for robust real-world deepfake detection.

Keywords: Deepfake detection; Generalization; Cross-dataset performance; Verifake dataset; Roop; FaceFusion; FaceForensics++; Celeb-DF; DFDC; Convolutional Neural Networks; Transformers; Contrastive learning; Reconstruction-based learning; Disentanglement; Same-dataset testing; Feature visualization; Robust feature learning; Benchmarking

Acknowledgement

To begin with, we are thankful to the Almighty for giving us the strength, endurance, and knowledge to accomplish this thesis. Next, we are extremely grateful to our supervisor, Dr. Muhammad Iqbal Hossain, who has always been supportive and provided thoughtful advice which has helped us to successfully complete the thesis. Lastly, we are also thankful to our parents, whose kind support and prayers have made it possible for us to complete this thesis.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

AUC Area Under the Receiver Operating Characteristic Curve

Celeb-DF Celeb-DF dataset

CNN Convolutional Neural Network

DFDC DeepFake Detection Challenge

DFD Deepfake Detection Dataset

ELA Error Level Analysis

EM Expectation–Maximization algorithm

FF++ FaceForensics++

GAN Generative Adversarial Network

GFPGAN Generative Facial Prior GAN

GRL Gradient Reversal Layer

HQ High Quality

HTER Half Total Error Rate

InSwapper Face-swapping model

LFW Labelled Faces in the Wild

LQ Low Quality

MLP Multi-Layer Perceptron

MS1M v2 Cleaned large-scale face recognition dataset(MS-Celeb-1M v2)

MTCNN Multi-task Cascaded Convolutional Networks

ResNet Residual Network

RGB Red–Green–Blue

RL-TTA Reinforcement Learning based Test-Time Augmentation

ROC Receiver Operating Characteristic

SimSwap Identity-preserving face-swap method

Swin-Tiny Swin Transformer

t-SNE t-distributed Stochastic Neighbor Embedding

Temp Temporal module capturing frame-to-frame features from samples

VeriFake Custom deepfake dataset created for this study

Xception Depthwise-separable convolutional CNN

Chapter 1

Introduction

The root of the deepfake problem can be traced back to 2017 when a Reddit user sparked controversy by posting doctored videos of well-known celebrities in compromising scenes [1]. Since then, Deepfakes have been increasingly misused. Deepfakes involve seamlessly merging one person's image into another's picture or video and have the ability to produce realistic-looking content that may be utilized for illegal activities. From creating fake news to manipulating political speeches and committing fraud, it poses a significant threat to security, privacy, and public trust. With advancing AI and deep learning technologies, it is very much possible to create fake content easily that can not be distinguishable even by human beings. While these advancements hold potential for entertainment and creative industries, they also pose significant risks and challenges for society. It is often seen that these technologies are used to create videos that put a person's reputation at stake. In 96% of cases, these technologies are abused to damage people's reputations by generating explicit content, such as superimposing innocent faces onto criminal scenes and manipulating political leaders' faces for controversial statements [2]. Furthermore, deepfakes can be used to fabricate evidence, blackmail individuals, and even influence elections by spreading false information, highlighting the urgent need for robust detection and countermeasures to protect individuals and maintain public trust.
Generating deepfakes has become increasingly accessible due to advancements in technology. The availability of user-friendly tools and online tutorials has made it possible for anyone, even without a background in deep learning or AI, to create fake videos and images. Simple tutorial videos on various social media platforms provide step-by-step instructions, enabling users to generate deepfakes using software like Photoshop, various editing tools, and machine learning models [3]. The tools available for creating deepfakes use several advanced technologies to make fake data that can easily deceive human eyes. The main technology that is used for deepfake generation is Generative Adversarial Networks (GANs), which have two parts: one creates fake images or videos, and the other tries to spot the fakes. This back-and-forth process helps make the fakes more convincing over time. These tools also use Convolutional Neural Networks (CNNs) to process images and videos, making it easier to change facial features and expressions accurately. Autoencoders are another technology used to encode and decode facial data, which helps in swapping faces seamlessly.
However, the existing models are limited to existing and they often fail to generalise new generation techniques [4]. The motivation behind creating a dataset using

Roop and FaceFusion stemmed from an exploratory intent to better understand the limitations and challenges of deepfake detection systems under real-world conditions. The existing datasets were created using old generation methods and controlled pipelines, tools like Roop and FaceFusion are more current and easily accessible that closely portrays how deepfakes are actually generated nowadays [5]. They are simple to use and ideal for constructing a dataset that reflects the similar synthetic media that has been creating havoc in the internet in recent days [6]. The intention was to produce a more realistic variety in facial blending, lighting, and resolution factors that frequently contribute to detection difficulty and it was done by utilizing these contemporary techniques. The resultant dataset is used as a testbed to assess how effectively the state-of-the-art models detect deepfakes. The dataset is named 'Verifake' and it consists of 500 real and 500 fake samples. The numbers were kept similar to maintain a balance in training and testing of the samples. In order to provide a more realistic and reliable understanding of generalization failure in deepfake detection, this effort sought to simulate real-world detection scenarios and identify potential blind spots in the current model performance.
To combat the dire situation, several CNN models are being used to detect deepfake videos. From our survey, it is seen that the current state-of-the-art models are CNN, YOLO, ResNet, GoogleNet, transformers etc. These models analyze various visual cues, such as inconsistencies in facial expressions, unnatural movements, and irregularities in lighting [7] and shadows, to identify signs of manipulation. However, as GAN models continue to advance, researchers are exploring hybrid approaches that combine the strengths of different detection methods. These hybrid models often incorporate techniques from both CNN-based analysis and another ResNet or YOLO-based technique to enhance detection accuracy. Additionally, researchers are investigating the integration of audio analysis into deepfake detection systems. By examining audio signatures and discrepancies between lip movements and speech or, by observing the visual artifacts [7] in the videos, these methods aim to provide an additional layer of verification for identifying manipulated content. Despite these advancements, detecting deepfakes remains a challenging task, as creators continually adapt their techniques to evade detection algorithms. Twelve different deep learning based deep fake detection models are built and modified from both standard and modified architectures. The models span a range of convolutional and object detection based approaches. The goal is to compare the performance of the models in generalising across the datasets. However, in most of the cases the model becomes biased towards the dataset it was trained on and performs poorly on new or unseen samples [8]. The reasons for that include inadequate temporal modeling in a dataset, limited expressiveness of heuristic features and many more.

1.1 Problem Statement

Not long ago, one would have needed access to certain technologies to create realistic fake content. To afford these technologies, people would need to spend a fortune. As a result, only high-budget films have traditionally used CGI (Computer Generated Imagery). But as time progressed, various editing software were created which have made the editing relatively much easier. Rapid advancements in technology, especially AI and machine learning, have further simplified the process of generating deepfake content, making it more realistic. Improved algorithms can now produce

high-quality deepfakes faster and with fewer computational resources. Moreover, these algorithms have been integrated into easy-to-use software and applications, allowing people with little or no technical expertise to create realistic, high-quality, almost flawless deepfake content that can cause distress in society. Some commonly utilized technologies for creating deepfake content include GANs [1], [9], face swapping [10], face re-enactment [11], and voice synthesis [12]. Additionally, autoencoders [1] are used to create deepfakes by compressing the features of a source image or video into a reduced form and then reconstructing them onto a target, producing highly realistic modifications. Initially, deepfakes were employed for beneficial purposes, especially in the film industry for tasks such as dubbing and de-aging actors. However with time lately it is more used for bad purposes than good for example blackmailing people with malicious videos, spreading rumors, and conspiracies, defaming celebrities, and manipulating politicians' statements. For example, a video went viral featuring former President Barack Obama, where the voice of American actor Jordan Peele was used. Peele's voice was later synthesized to match the original video of President Obama [13]. Therefore to distinguish between real and fake and prevent harassment, violation of privacy, manipulation of political statements, and endless misuse of deepfake, researchers are working on deepfake detection techniques. The introduced approaches to detect deepfakes are like CNN, LSTM, RNN, and others. However, each of these methods has drawbacks in cross-validation despite achieving good performance on the specific datasets they are trained on. In most cases, existing deepfake detection methods struggle to perform efficiently on large datasets or different types of deepfake content, and they typically use common datasets like DFDC or FaceForensics++. As a result, models trained on a particular dataset often experience significant drops in accuracy when tested on different datasets, making real-life implementation challenging. Additionally, CNN models have difficulty detecting deepfakes in rotated or tilted images [14]. The limitations observed highlight the need for more robust and versatile detection models that are capable of maintaining high accuracy across different datasets and real-world scenarios [13]. Nonetheless, most deep learning based models become dataset specific. Models become biased on one of the entities 'real' or 'fake' making the accuracy decline. In outline, common datasets, models becoming biased to effect generalising across different datasets, and failure of deep learning models are the main challenges of detecting deepfakes nowadays.

1.2 Research Objectives

We aim to systematically analyze the generalization challenges of deepfake detection models, a persistent limitation that threatens the reliability in real-world applications. Although deepfake detectors often achieve impressive results on benchmark datasets, their performance deteriorates sharply when exposed to unfamiliar manipulations, as frequently observed with datasets such as DFDC. This study aims to bridge the understanding gap by systematically analyzing twelve state-of-the-art and modified architectures, trained and evaluated across multiple datasets, including Verifake, a novel dataset of real and synthetic videos generated using Roop and FaceFusion to capture diverse forms of facial forgeries. Through a combination of large-scale experiments, cross-dataset evaluations, feature space visualizations, and error analysis, this research seeks to:

- Identify model-specific and dataset-specific vulnerabilities that lead to poor generalization and high misclassification rates on unseen data.
- Investigate the inherent biases and overfitting tendencies of current detection models, especially their dependence on dataset-specific artifacts rather than intrinsic forgery cues.
- Examine the role of training data diversity and architectural design in shaping a detector's ability to handle novel manipulations, varied resolutions, and different compression levels.
- Provide actionable recommendations and guidelines for building future detection systems that are more resilient, trustworthy, and capable of maintaining accuracy in unpredictable, real-world environments.

Chapter 2

Related Works

2.1 Generalization Challenges in Deepfake Detection

While many deepfake detection models achieve high accuracy on their training datasets, they often struggle to generalize to unseen data. In cross-dataset evaluations, detectors that perform nearly perfectly on one benchmark can see their performance drop drastically on another due to differences in forgery techniques, compression levels, image resolution, and dataset-specific biases. For instance, an Xception model trained and tested on FaceForensics++ can exceed 99% AUC, yet its AUC may fall below 50% when tested on a different dataset like Celeb-DF [15]. This domain shift problem, where the distribution of test data diverges from training data is widely recognized as the biggest challenge in deepfake detection. In practice, popular datasets such as FF++, DFDC, and Celeb-DF cover only a portion of real-world manipulations; algorithms tuned to these sets tend to overfit to their artifacts and fail to adapt to new or more subtle deepfakes [16], [17], [18]. A recent systematic review found that about 46% of deepfake studies optimistically reported that their techniques could generalize to different types of fakes, yet in reality this remains an open problem – overfitting to limited training data was identified as a key limitation hampering robust cross-dataset performance [16]. The consensus in the community is that current detectors lack open-world robustness, and improving generalization requires further research into both algorithms and data diversity [16]. This lack of robustness has driven researchers to explore various strategies for cross-domain generalization. One major line of work is unsupervised domain adaptation (UDA), which aligns feature distributions between the source (training) domain and a target (unseen) domain without requiring target labels. UDA methods typically incorporate an adversarial domain discriminator that encourages the feature extractor to learn domain-invariant features, thereby reducing the discrepancy between training and testing data. For example, in [19], an adversarial UDA framework ("Towards Open-world Generalized Deepfake Detection") was proposed, integrating a feature extractor with a domain discriminator to align FaceForensics++ with other datasets. This approach improved cross-dataset detection AUC by roughly 13–15% (when training on FF++ and testing on Celeb-DF or DFDC) compared to a baseline Xception model. Crucially, however, even such improved models still fell short of same-dataset performance, as certain unseen manipulation patterns and residual domain shifts could not be fully eliminated. The results underscore that

while adversarial domain alignment helps, generalization gaps persist in scenarios with completely novel fake generation techniques or extreme video compressions. Another approach to enhance generalization is to perform on-the-fly adaptation at inference time. In [20], Nadimpalli et al. introduced a reinforcement learning–based test-time augmentation (RL-TTA) strategy to dynamically adjust the model for each test sample. Their method formulates deepfake detection as a combination of supervised learning (using an EfficientNet-B4 backbone) and an RL agent that selects an optimal set of image augmentations for each input. By averaging the predictions over multiple augmented variants of the test image, the detector can become more robust to distribution shifts. This technique improved the cross-dataset AUC on Celeb-DF to 66.9% (for a model trained on FF++), outperforming standard single-pass CNN detectors on the same task. The RL-TTA framework effectively learns how to "explore" different views of the data (e.g. changes in resolution, color, or compression) to counteract the bias of the training domain. However, its success heavily depends on the augmentation policy: if a novel deepfake artifact lies outside the augmentation space considered, the method may still fail to correct the model's misclassification. In other words, coverage of possible transformations is crucial, and the RL agent might need retraining or tuning when encountering manipulation types or image conditions not anticipated in the augmentation set. A related direction is to use meta-learning and few-shot adaptation so that the detector can quickly adjust to new domains. In [21], Chen et al. proposed OST: One-Shot Test-Time Training, a paradigm where the model performs a brief self-update using the test sample itself before making a prediction. Specifically, for each test video or image, they synthesize pseudo-fakes (for example, by applying small random warps or perturbations) and define a test-time auxiliary task: the model is fine-tuned on these pseudo-fakes vs. the original image (treated as real) for one gradient step prior to inference. Through a meta-learning procedure during training, the model is optimized to benefit from this single-step on-the-fly fine-tuning so that it can rapidly adapt to previously unseen forgeries. This OST method with a ResNet-50 backbone reportedly boosted cross-dataset AUC on DFDC and Celeb-DF by about 8–10% compared to a conventional XceptionNet baseline (which has no test-time adaptation). By tailoring itself to each test instance, the detector can correct for certain distribution shifts (e.g., a new deepfake technique's specific artifacts) at test time. However, the approach incurs significant computational overhead, as it requires running a forward-backward update for each sample, and it can struggle in scenarios with very low-quality inputs – e.g., heavily compressed videos where generating informative pseudo-fakes is difficult. Thus, while one-shot test-time training improves generalization, it may not be practical for real-time or large-scale screening applications without further optimization. Beyond these methods, researchers have also explored data-centric solutions to improve generalization. Some works aggregate multiple training datasets or create augmented training sets that encompass a wider variety of manipulations and post-processing artifacts. Others employ disentangled representation learning to separate identity, expression, and other factors, aiming to isolate features that are invariant to specific fake-generation techniques. For instance, combined face disentanglement with detection, helps the model to focus on forgery-related cues rather than person-specific details – this yielded more robust performance across FF++ and Celeb-DF [22]. There have also been attempts to use GAN-generated adversarial fakes to train detectors: by intentionally crafting difficult fake examples, detectors can learn more

general decision boundaries [23]. Ensemble techniques (e.g., combining CNN, RNN, and Transformer outputs) have been shown to improve overall accuracy and even obtain moderate cross-dataset gains [24]. But they tend to be resource-intensive and still face the "last mile" generalization gap when confronted with completely new fake styles [16]. Recently, continual learning frameworks have been proposed to incrementally update deepfake detectors as new kinds of forgeries emerge. By periodically retraining or fine-tuning on a small batch of newly collected fakes (while retaining knowledge of old ones), a detector can slowly expand its coverage of the fake landscape. For example, an approach by Prathibha et al. (2024) added a self-attention module and used weight regularization to continually learn from new data, achieving near 99% AUC on known benchmarks and 90% AUC on a previously unseen dataset (Celeb-DF) after incremental updates [25]. Continual learning, however, assumes access to a stream of fresh fake examples and can be vulnerable to "catastrophic forgetting" (losing accuracy on earlier data) if not carefully designed. In summary, generalization remains a fundamental hurdle for deepfake detection. Even with the above innovations, most detectors still perform noticeably worse on forgeries that differ from their training data. Going forward, researchers emphasize the need for more comprehensive and diverse training datasets that include a broad spectrum of deepfake generation techniques and real-world perturbations [16]. Such data, combined with algorithms capable of adapting to distribution shifts, will be crucial for detectors to handle the ever-evolving deepfake landscape. In the next section, we review the literature on deepfake detection techniques in detail, examining the specific visual, temporal, and other features they leverage. This survey of methods will further highlight how the choice of features and model architectures can impact a detector's performance and its generalizability across different datasets.

2.2 Literature Review

Deepfakes are manipulated videos or images created using AI and DL techniques, often with the intent to deceive viewers into believing false information or events. These manipulations can include altering facial expressions, gestures, or voices to make someone appear to say or do things they never did. Detecting deepfakes is challenging due to their increasingly realistic appearance. To address this challenge, researchers have developed different models and methods for deepfake detection. These models range from traditional machine learning algorithms like support vector machines and k-nearest neighbors to deep learning architectures such as Convolutional Neural Networks (CNNs) along with transfer learning methods(TL) [2] such as Xception, NAS-NET, Mobile Net, VGG-16, and recurrent neural networks (RNNs). Additionally, researchers have identified various features that can be used for detection. These features include visual artifacts such as inconsistencies in facial features, unnatural movements, or distortions in the background, as well as temporal features like mismatched patterns or abrupt changes in motion over time.

2.2.1 Visual Artifacts

Visual artifacts refer to irregularities or anomalies in manipulated media, such as inconsistencies in facial features, unusual movements, mismatched lighting or shadows, or distortions in the background. Different methods were employed that ana-

lyze visual artifacts to detect deepfakes [9], [26], [27], [28], [29], [30], [31]. A hybrid model YOLO-InceptionResNetV2-XGBoost consisting of YOLO, CNN, and XGBoost was proposed in [9]. It was trained using the c23 dataset and has achieved an accuracy of 90.73%. In [26], deepfakes were detected with an accuracy of 96.5% (DFDC); LQ 99.68%, HQ 91.88% (DeepfakeTIMIT). In this paper, a decision fusion method was proposed which averages frame-by-frame individual predictions from VGG16, InceptionV3, and XceptionNet. Each of the models was trained on 25% of the DFDC dataset. This approach proved effective on both DFDC and DeepfakeTIMIT datasets, even under FGSM attack distortions. On the other hand, VGG16, ResNet50, ResNet101, and ResNet152 were individually employed in [27] on UADFV and DeepfakeTIMIT datasets to find the best-performing model. Among these models, ResNet50 was observed to achieve the highest AUC performance score. According to [28], an unsupervised ML model called EM algorithm was used with the CelebA dataset and 5 different datasets generated by GANs (GDWCT, STARGAN, ATTGAN, STYLEGAN, STYLEGAN2). The final model achieved the highest accuracy of 99.81% with STYLEGAN2. Moreover, in [29], both ML and DL methods were used on Yonsei University's Computational Intelligence and Photography Lab compiled dataset which contains images of both real and fake human faces. Here, ELA was used to detect image alterations at the pixel level. These ELA-validated images were then supplied to SqueezeNet, ResNet18 & GoogLeNet for deep feature extraction and later classified via SVM and KNN, achieving an accuracy of 89.5%. As outlined in [30], a hybrid model consisting of CNN and LSTM was used with both DFDC and the CipLab dataset. The accuracy was 97.32% and 98.24% when tested with DFDC and the CipLab dataset respectively. In the end, [31] proposed a CNN model with only 4 convolutional layers called MesoNet-4. This model was trained and tested on both the Face2Face and the custom deepfake datasets, achieving an accuracy of 96.9% on the custom Deepfake dataset and 95.3% on the face2face dataset.

2.2.2 Temporal Features

Temporal features in deepfakes refer to mismatched or unnatural patterns that appear over time in video sequences. These features include differences in facial expressions or head movements that do not flow smoothly from frame to frame, creating a jerky or strange appearance. In [32], [33], [34], [35] temporal features were analyzed, and various methods were employed for deepfake detection. According to [32], DPNet uses learned dynamic prototypes to distinguish real from fake videos and employs Timed Quality Temporal Logic (TQTL) to ensure consistent and reliable temporal behavior. It was trained on Google's DeepFakeDetection, DeeperForensics, and Celeb-DF datasets. DPNet achieved AUC improvements from 92% to 99% for high-quality videos and from 83% to 91% for low-quality videos, outperforming Xception and Two-branch models. [33] proposes a Convolutional LSTM-based Residual Network (CLRNet) for deepfake detection. CLRNet extracts consecutive frames from videos to capture temporal information such as sudden changes in brightness and contrast on small regions of the face, and size variations in facial parts like eyes, lips, and eyebrows between frames. The model uses the FaceForensics++ dataset and demonstrates better generalizability and performance compared to zero-shot models across various deepfake datasets. [34] tested base CNN and

VGG models separately in order to detect the deepfakes. Both models were trained on DFDC dataset where CNN and VGG achieved 94% and 88% respectively. It was also proposed that CNN is better than the VGG as loss in VGG was higher compared to the CNN. Lastly [35] uses CNN and SVM to detect deepfake images from the 140k Real and Fake Faces dataset containing 70,000 real and 70,000 GAN-generated images, achieving accuracies of 89.93% and 84.82%, respectively. Their method includes convolutional layers, max pooling, dropout layers, and a sigmoid-activated output layer for binary classification, optimized with Adam and binary cross-entropy loss.

2.2.3 Texture Features

Texture features in deepfakes refer to the unique patterns and details present in an image, such as skin texture or fabric patterns. In [36] we found a deepfake detection method that utilizes YOLO v3 for face detection in videos and HRNet for CNN-based feature extraction from detected faces. Local Binary Patterns (LBP) are used for texture analysis to enhance detection accuracy. These features are concatenated into a single vector and input into a Capsule Network for classification. The dataset includes samples from the DeepFakeDetectionChallenge-Preview (DFDC-P) and Celeb-DF.

2.2.4 Facial Features

Deepfake detection by facial features involves identifying altered videos or images by analyzing various elements and attributes of human faces. This process includes a thorough examination of both the static and dynamic properties of facial features to spot irregularities and inconsistencies that indicate deepfake manipulation. In [1], [3], [37], [38] various techniques were utilized to analyze facial features for deepfake detection. Firstly in [1], a Convolutional Vision Transformer which consists of CNN and ViT has been used to detect deepfake videos. This model was trained on the popular DFDC dataset and achieved an accuracy of 91.5%. In this model, CNN was used to extract the learnable features and ViT helped in categorizing them. Secondly in [37], a 20 Network layer DeepfakeNet combining ideas from ResNet and Inception has been proposed to detect various types of deepfakes without dropping the accuracy. So, to perform well in cross-validation, this model has been trained on FaceForensics++, Kaggle, and TIMIT datasets and performed better compared to VGG, GoogleNet, XceptionNet, ResNet, and ResNeXt on the same dataset. Moreover, [3] employs an EfficientNet model with CNN-RNN and convolutional autoencoder, achieving 94% accuracy on the DFDC dataset and 98% on FaceForensics++ by extracting and focusing on facial features in frames to enhance detection accuracy. Lastly, [38] utilizes a ResNet50 model augmented with four dense swish layers, achieving 99% accuracy on face swap detection and 94% on face reenactment using the DFDC and FaceForensics++ datasets by leveraging improved learning behavior and complex pattern recognition through the swish activation function.

2.2.5 Frequency Features

Images and videos are represented as a combination of different frequency components in the frequency domain. Frequency-based analysis in visual deepfake detection involves transforming images or videos into the frequency domain to detect irregularities, particularly in high-frequency components, which may indicate inconsistencies and help identify fake content. In the case of using a 3-layered Frequency Convolutional Neural Network (fCNN), the extracted faces are converted into the frequency domain using Viola-Jones [39]. Despite having fewer layers compared to state-of-the-art models, it was able to perform quite well. It was trained on the c23 dataset and was able to detect deepfake with an accuracy of 85.24% when tested with the c23 dataset, FaceSwap with an accuracy of 89.28% with raw Deepfake dataset, and Face2face with accuracy of 85.37% with raw Deepfake dataset. On the other hand, using MesoNet along with a preprocessing module filters out the low frequencies and captures the high frequencies [40]. It helped to increase the detection accuracy and achieved higher AUC throughout different datasets. The hybrid model achieved AUC of 0.974 and 0.942 on FaceForensics++ and CelebDF datasets respectively. Overall 88% accuracy was achieved on average over both datasets.

2.2.6 Spatial Features

Spatial features in deep learning are characteristics related to the spatial arrangement of elements in data, such as pixels in an image or positions in a grid. These features capture information about the structure and layout of objects within the data. In [41], a hybrid YOLO EfficientNet-b5 Bi-LSTM model has been trained with a c23 dataset to detect deepfake videos. Here, the YOLO-Face-Detector detects face regions in video frames, EfficientNet-B5 extracts spatial features from faces and, Bi-LSTM brings out the temporal features from the video. Although it was able to achieve 89.38% accuracy in deepfake detection, it was thought that the model could perform even better if the audio content of the videos were taken into consideration. [42] uses deep learning methods combined with Explainable AI (XAI) to detect deepfakes by extracting spatial features. Various CNN models—InceptionResNetV2, DenseNet201, InceptionV3, and ResNet152V2—were employed to classify real and fake images. The Local Interpretable Model-Agnostic Explanations (LIME) algorithm was used to explain which parts of the image led to the model's classification decisions. The dataset consisted of 140,000 images, with 70,000 real images from the Flickr-Faces-HQ (FFHQ) dataset and 70,000 fake images generated by NVIDIA's StyleGAN. The best-performing model, InceptionResNetV2, achieved an accuracy of 99.87%.

2.2.7 Face Verification

Face verification involves comparing two images of the same person, one authentic and one deepfake, to identify any discrepancies or inconsistencies that may indicate a fake image. This process helps in confirming the authenticity of the person's identity by detecting subtle differences between genuine and altered images. This can be found in [43] where deepfakes are detected by calculating similarity scores between face images from a questioned video and reference images of the same person. Using a trained face recognition model, pairwise similarity scores are computed, and the

highest score ($S_{\max}$) is compared to a threshold to classify the video as real or fake. The method uses three datasets: Celeb-DF (v2) for validation, MS1Mv2 for training the face recognition model, and LFW for evaluating accuracy. The model achieved 99.83% accuracy on LFW, with an HTER of 0.020 and an AUC of 0.994 on Celeb-DF (v2).

Chapter 3

Dataset

The development of deepfake detection technology greatly depends on access to high-quality datasets for both training and evaluation. These datasets form one of the core foundations of developing robust algorithms that can reliably differentiate between real and fake content. Existing datasets such as DFDC [44], DFD, FF++, FF, DF-TIMIT-HQ, UADFV [45] and Celeb-DF [46] suffer from significant limitations which are discussed in detail in the following subsection [47]. These easily accessible datasets have also been used to train multiple models, making it difficult to introduce novel approaches solely based on them. To overcome these issues, we developed a new dataset consisting of 500 real and 500 fake videos, aimed at improving the understanding of whether these models are indeed robust and capable of generalizing across different video conditions [48]. The following subsections will discuss the limitations of existing datasets and outline the approach taken to construct the new dataset.

3.1 Existing datasets and their limitations

3.1.1 FaceForensics++

The FaceForensics++ dataset was introduced in 2019 by Rössler et al. This dataset soon became one of the foundational benchmarks for the research of deepfake detection [49]. The dataset is divided into five parts where the first part is composed of 1000 real samples which were collected from YouTube. The samples consist mostly of frontal face recordings of the people speaking under controlled light conditions. These samples were manipulated using six different facial manipulation techniques: Deepfakes, FaceSwap, FaceShifter, InsightFace, Face2Face and NeuralTextures. These six parts comprise 6000 manipulated videos. So each real sample has six different manipulated counterparts. This makes FF++ a rich dataset for training and evaluating forgery detection models. The dataset's main focus is face swapping and facial expression alteration, which enables researchers to examine a variety of forgery-based inconsistencies. Nonetheless, FF++ has some limitations despite its impact and wide adoption in the research community. The dataset has a limited number of facial samples which reduces the diversity of race, age, lighting conditions and camera angles. Furthermore, most videos are shot in very clean locations, which frequently fail to capture real-world complexity like occlusions, changing backgrounds, or many faces. This constraint has been discovered as a fundamental

reason why many models trained on FF++ fail to generalize when tested on more complicated datasets or real-world films[50]. Despite this, FF++ remains a critical dataset for baseline evaluations and has a remarkable influence in the development of several deepfake detection models.

3.1.2 Celeb-DF (v2)

The Celeb-DF v2 is an improved and extended version of the Celeb-DF v1. While the first version of 2019 comprised 408 real and 795 fake videos, the modified version (v2) comprises 590 real and 5639 high-quality fake samples and was released in 2020 by Li et al.[51]. The dataset covers a wide range of appearances and head poses. The real samples of 59 celebrities were manipulated using DeepFake algorithm that minimizes the common artifacts which includes flickering, inconsistent boundaries, and poor lip syncs which were common issues in the earlier version of the dataset. The main contribution of this dataset is to raise the bar for forgery realism while keeping the research scale feasible. The clean pipeline generation of Celeb-DF helps the models that rely too heavily on low level visual artifacts and not actual semantic and temporal errors. However, the dataset suffers from limitations that restrict the robustness in generalising research. Although the modified version eliminated many of the visual limitations seen in Celeb-DF v1, it still has a limited diversity. The dataset contains just 590 genuine videos, all of which are celebrity interviews on YouTube, introducing a homogenous domain bias. This lack of intra-class variability produces an inflated sense of performance when testing on the same dataset but results decline sharply when cross-evaluated on new or unseen datasets. Furthermore, only one face swapping technique which is an improved version of an autoencoder based model is used to produce the fake samples in the dataset. Due to its limited scope, Celeb-DF is often unable to represent the range of manipulation techniques correctly. The methods include, expression transfer, face reenactment, and sophisticated GAN-based approaches like StyleGAN and the First Order Motion Model[52]. This drawback frequently results in deepfake detection models trained on Celeb-DF v2 picking up patterns unique to that one generation technique. As a result, people can have trouble identifying fakes made with various methods and would not be able to generalize effectively to novel, invisible manipulations.

3.1.3 DFDC

In the year of 2020, Facebook AI collaborated with Amazon Web Services and several academic institutions released the Deepfake Detection Challenge (DFDC) dataset as a part of a global competition in focus to drive innovation in deepfake detection techniques[53]. The DFDC dataset stands out the most in deepfake detection research as it is the largest and most diverse dataset which is publicly available. The dataset comprises 100 thousand fake videos and 19,164 real videos which featured 3426 unique paid actors of different age, race and gender. The dataset was created using a variety of deepfake generating methods, such as GAN-based models, autoencoders, and sophisticated face-swapping tools, to mimic real-world situations. The altered movies were subjected to a variety of post-processing techniques, including noise addition, resizing, and re-encoding, in order to introduce compression artifacts and other distortions that are commonly found in deepfake content that circulates

on social media. A notable strength of the DFDC dataset is its scalability and diversity. However, due to its massive size which is over 470 Gigabytes which poses computational challenges for model training, especially for researchers with limited hardware resources. A notable finding from the DFDC dataset was that many deepfake detection models that performed well on smaller, simpler datasets were unable to retain accuracy on DFDC, underscoring the issue of dataset bias and overfitting. Despite its diversity, the fake samples of the dataset exhibit low perceptual quality. Temporal inconsistencies, blurry face boundaries, unrealistic facial movements, color mismatches can be easily detected by the deepfake detecting models [54]. As a result, models trained on DFDC often fail to generalise the higher quality deepfakes from newer datasets or real world videos of the internet.
Currently, the deepfake detection research relies heavily on some widely used datasets such as DFDC(DeepFake Detection challenge), FaceForensics++ and DF-TIMIT-HQ. For instance, in FaceForensics++, there are 1000 real videos and 6000 fake videos. Deepfake generation models like DeepFakes, FaceSwap and Face2Face were used in order to generate the fake samples included in this dataset. The DFDC dataset is a very large dataset comprising around 1,00,000 videos with diverse quality and individuals. Meanwhile, Celeb-DF focuses only on video samples with celebrity faces. The dataset contains 590 real and 5639 fake samples.

3.1.4 Limitations of existing datasets

Although these datasets played a crucial role in the advancement of the deepfake detection models, some key limitations still persist. Low quality samples due to compression and presence of visible artifacts caused by the manipulation algorithms is one of the major setbacks. This deteriorates the quality of newly created deepfake and can be easily detectable as they lack realism [44], [47], [55]. Furthermore, the visible imperfections, mainly poor blending, flickering or inconsistent facial expressions noticeably makes the fake defected.These artifacts can be easily detected by the naked eye as they reveal their manipulative nature and do not require any advanced tools for detection.
While existing datasets have been instrumental in advancing deepfake detection research, they have several drawbacks:

- Older generation manipulation techniques: Many datasets, particularly FF++ and DF-TIMIT, rely on face-swapping algorithms from 2018–2019. These forgeries often leave visible artifacts, unnatural facial boundaries, or blending issues that no longer reflect the quality of modern deepfakes.

- Compression and quality inconsistencies: DFDC and Celeb-DF contain many low-resolution or heavily compressed videos, which can bias detectors toward learning compression artifacts instead of intrinsic forgery cues.

- Dataset bias and overfitting risk: Many models trained on FF++ or Celeb-DF learn dataset-specific artifacts rather than manipulation traces, leading to significant performance drops on unseen datasets like DFDC.

3.2 Purpose of creating VeriFake dataset

To address these challenges, a new dataset, Verifake, was created consisting of 500 real and 500 fake videos generated using Roop (2023) and FaceFusion (2024), two state-of-the-art face-swapping tools capable of producing highly realistic manipulations. Unlike older datasets, Verifake reflects the quality and diversity of deepfakes generated by modern publicly available tools, ensuring that detectors are tested against manipulations closer to what is encountered in the wild. Key characteristics of Verifake:

- High-quality forgeries: Generated at high resolution with improved blending, facial alignment, and texture consistency compared to older datasets.
- Diverse conditions: Includes variations in lighting, angles, and facial attributes to reduce dataset bias.
- More realistic challenge: Videos are visually convincing even to human observers, making it harder for detectors to rely on low-level artifacts.

This dataset complements FF++, Celeb-DF, and DFDC used in this study, providing an additional benchmark to analyze how well models trained on existing datasets adapt to modern, high-quality forgeries. The goal is not to replace existing benchmarks but to simulate realistic contemporary deepfakes and provide insights into the generalization weaknesses of current detection models.

3.3 Creation of a High-Quality Dataset

An entirely new dataset called VeriFake has been developed from scratch, considering the limitations of the existing datasets. The most advanced and latest face-swapping tools were used to generate the 500 fake samples and to make the dataset balanced, 500 real videos were also added. The state of the art tools like Roop(2023) [56] and FaceFusion(2024) [57] were used in this dataset to generate the deepfakes. On the other hand, the used datasets rely on the tools from the past generation which may not reflect the advancements in the current deepfake generation technology.

Name	**Release Year**	**Original Footage**	**Deepfake Footage**
FaceForensics [58]	2018	1,004	2,008
FaceForensics++ [59]	2019	1,000	6,000
DFDC [60]	2020	23,654	104,500
Celeb-DF [61]	2020	590	5,639
DF-TIMIT-HQ [62]	2018	320	640

Table 3.1: A comparison of available datasets.[63]

VeriFake focuses on more realistic manipulations of videos, ensuring higher resolution of generated deepfakes. These allow it to surpass other existing datasets, making it almost impossible to be detected by the naked eye.

Additionally, diversity was ensured in terms of facial characteristics, lighting conditions, and scenarios, which makes the dataset more robust for training deepfake detection models. The higher quality and diversity address the current gap in datasets where deepfake videos often lack enough variation and realism, limiting the ability of detection models to generalize to real-world conditions. By offering a more challenging and diverse dataset, the goal is to enhance the robustness and accuracy of detection algorithms, pushing the boundaries of deepfake detection research. VeriFake is supposed to be more fitting for the real world scenarios compared to the shortfall of the other datasets. Moreover, it is expected to serve as a solid foundation to train existing and future deepfake detection models.
Among the existing tools that are using GAN models, FaceFusion [57] and Roop [56] were opted. Both tools are specialized in swapping faces by processing the video frame by frame. By analyzing the target video and source image, both tools swap the faces by adjusting movements, orientation, lighting conditions and facial expressions in the target video. Here, each of the frames are processed individually to ensure a seamless transition, resulting in a video where one person's face is replaced by another, while maintaining the original body and background. While FaceFusion provides a wide range of models for swap and other changes, Roop only uses a single model for face swapping.

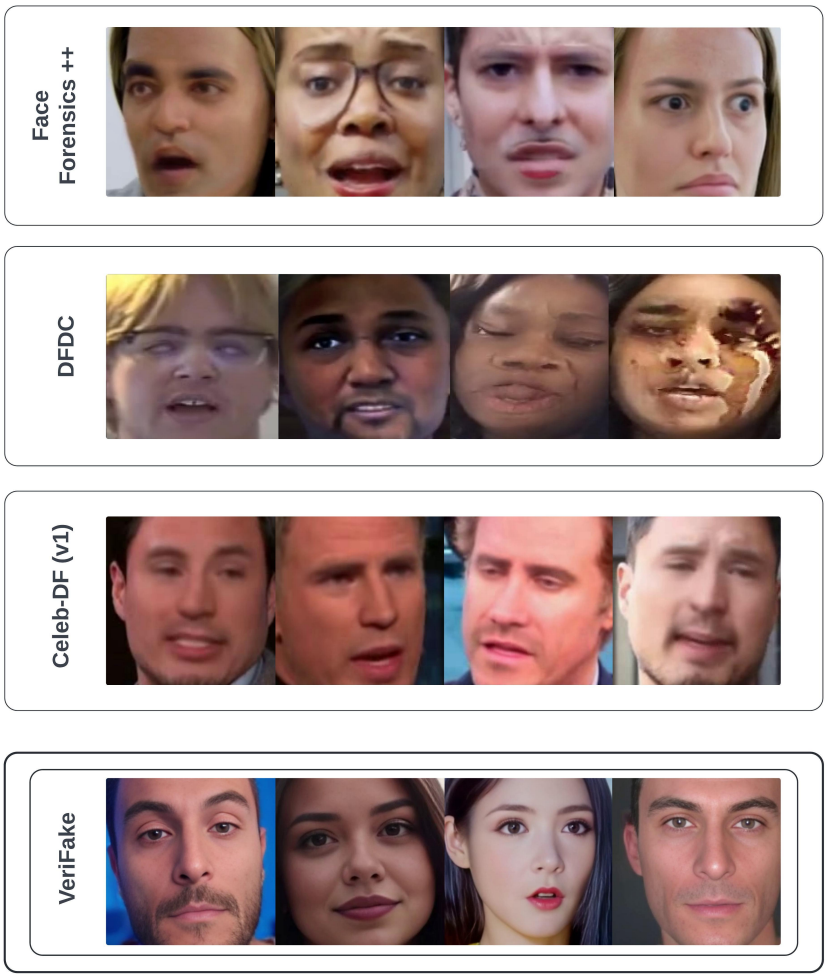

Figure 3.1: Quality comparison of extracted faces between existing datasets and VeriFake

3.3.1 Face Swapping Workflow

Despite the differences described in the previous subsection, both tools follow a similar workflow: they take a target video and a source image as inputs, and after processing each frame successively, they perform the face swap. To further enhance the final output, both Roop and FaceFusion incorporate face enhancers that improve the quality and realism of the swapped face. The general workflow includes:

1. **Face Detection:** Algorithms like Dlib, MTCNN, and CNNs are used to detect facial landmarks (eyes, nose, mouth, chin) in each frame of the video. These tools identify the positions of the key facial features of both the source and target faces.

2. **Landmark alignment:** In this step, the key facial landmarks from both the source and target faces are aligned to ensure they are positioned correctly. Here, the pre-trained deep learning models, like GANs or autoencoders are used to make sure the faces are oriented and scaled properly

3. **Feature Blending:** After aligning the faces, the facial features are blended using several methods:

   - **Poisson Blending:** This method blends the facial expressions, head movements, and lighting between the two faces.
   - **Weighted Average:** With specific weights, pixels or regions from both faces are averaged and it determines the influence amount each face has on the final image.
   - **Morphing:** This technique gradually transforms one face into the other by smoothly transitioning between corresponding landmarks.
   - **Alpha Blending:** This method uses transparency masks to combine the images, controlling how visible each face is in the final result.

4. **Face Swapping:** Two different GAN-based models, SimSwap and InSwapper, are used for the face swap:

   - **InSwapper** focuses on preserving the identity of the target face, accurately maintaining its expression for a highly realistic outcome. However, this process takes more time than SimSwap.
   - **SimSwap** swaps faces while keeping the structure and appearance of the target face intact. It is comparatively faster, but produces slightly lower quality results when compared to InSwapper.

5. **Color Matching:** To ensure the swapped face matches the skin tone, lighting, and shadows of the target video, color correction techniques are applied. This includes histogram adjustments, color transfer algorithms, and lighting corrections, helping the new face blend in naturally.

6. **Face Enhancement:** In order to make the manipulated face more realistic by eliminating any artifacts or distortions and by refining swapped face's structures, texture, and definition, enhancement is performed with the help of GFPGAN model

7. **Final Video Creation:** A final video with the swapped face is compiled together after processing and blending all the frames. Finally, it was ensured that the created deepfake appears natural and seamless throughout.

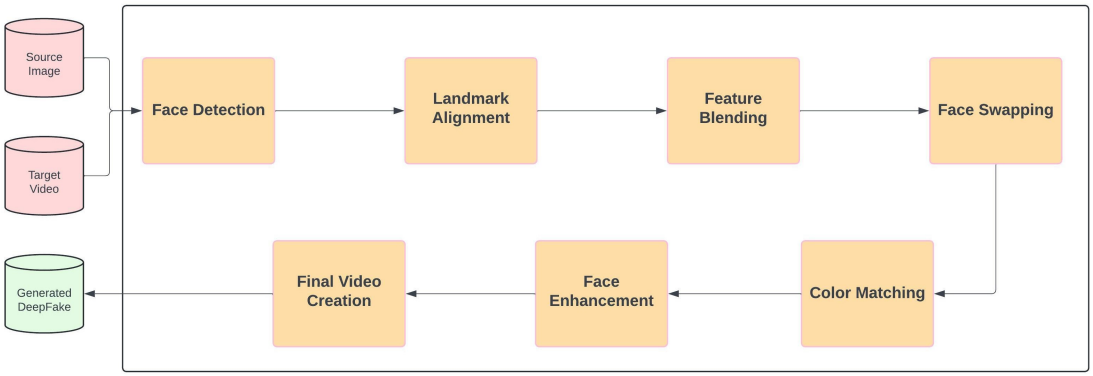

Figure 3.2: General workflow of deepfake creation

Chapter 4

Methodology

4.1 Data Preprocessing

To maintain consistency in input representation and enable reliable comparisons across models, a unified preprocessing pipeline was applied to all videos. From each input video, 100 frames were uniformly sampled to capture a representative range of motion and expression while keeping computational demands manageable. For validation purposes, a smaller subset of 20 frames was randomly drawn from the first 100 to detect potential corruption or unreadable video segments. Each sampled frame was processed to extract facial regions using the RetinaFace detector, implemented via the InsightFace library. This detector was chosen after comparative testing with alternatives such as Dlib and MTCNN, which were ultimately dismissed due to inconsistent performance under challenging conditions. RetinaFace consistently outperformed others, particularly when handling occluded faces, non-frontal angles, and low-light settings. RetinaFace is a single-stage dense face detector based on a ResNet backbone with a pixel-wise facial landmark regression module. It provides:

- High accuracy in detecting faces under varied poses, lighting, and occlusion.
- Precise landmark localization, which is essential for effective alignment.
- Support for 5-point and 106-point landmarks, with flexibility depending on the use case.
- Real-time performance on both GPU and CPU, allowing scalable processing of large video datasets.

These qualities make RetinaFace particularly suitable for deepfake datasets, which often include manipulated faces with unnatural warping or expressions. Accurate landmark detection ensures consistent alignment even in such altered frames, which is critical for minimizing variability and increasing detection reliability. Once a face was detected, the first detected face in each frame was aligned to a standard orientation using five facial landmarks specifically, the eye corners, nose tip, and mouth corners. OpenCV's estimateAffinePartial2D and warpAffine functions were used to compute and apply the transformation, respectively. All models, except the final one, used 112×112 pixel aligned face crops after facial alignment. For the final model, the crop size was increased to 256×256 to preserve more spatial detail and potentially capture finer manipulation cues. This alignment step played

a crucial role in reducing variation due to pose or camera angle, allowing models to focus on manipulation-related artifacts. The aligned faces were then converted

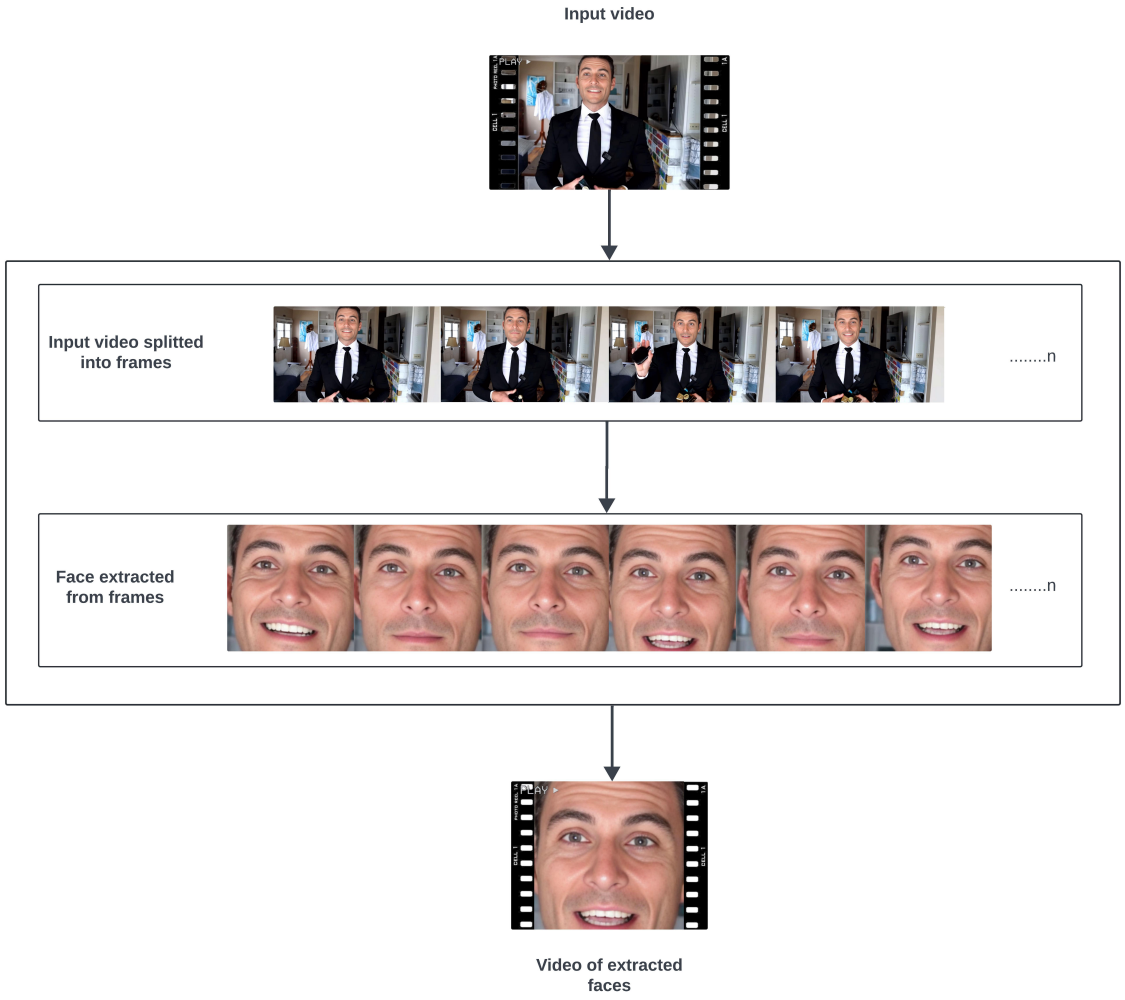

Figure 4.1: Overview of data preprocessing

to RGB format and normalized using ImageNet channel-wise statistics: means of [0.485, 0.456, 0.406] and standard deviations of [0.229, 0.224, 0.225]. No additional augmentations were applied during preprocessing in order to retain the subtle manipulation artifacts critical for deepfake detection. For the models using temporal features, videos containing fewer than 15 usable frames were discarded to ensure sufficient temporal information for downstream analysis. In addition, a corruption check was performed by attempting to extract and align 20 frames; any video that failed this test was excluded from the dataset. After preprocessing, the aligned frames were compiled into new 112×112 resolution videos at 30 frames per second using OpenCV's VideoWriter, ready for model training and evaluation. This preprocessing protocol was consistently applied to all models throughout the study. Its emphasis on spatial alignment and data integrity contributed to more reliable model performance, especially in scenarios requiring precise facial feature analysis.

4.2 Data Augmentation

To ensure reproducibility and consistency across experiments, a fixed random seed was applied before dataset splitting. The dataset was then partitioned into 80% training, 10% validation, and 10% test sets. Class distributions were preserved across splits to maintain real/fake balance. Two transformation pipelines were implemented using the Albumentations library: one for training and one for validation/testing. The training augmentation pipeline was designed to simulate a wide range of real-world distortions and inconsistencies to prevent overfitting and improve generalization. It included:

- Horizontal Flipping with a 50% probability to introduce mirrored variations of faces.
- Random Brightness and Contrast Adjustments to simulate variable lighting conditions.
- JPEG Compression Artifacts via ImageCompression, to mimic real-world compression effects common in social media or low-quality video platforms.
- Gaussian Noise Injection, which helped the model learn to ignore subtle noise patterns.
- Blur Effects, introducing mild defocus to account for motion blur or camera shake.
- Normalization using ImageNet mean and standard deviation to standardize input distribution.

For the validation and test sets, only normalization and tensor conversion were applied to ensure that evaluation occurred under consistent and unaltered conditions. The augmented frames were loaded using custom PyTorch DataLoader objects, with each sample containing 15 consecutive aligned frames. This approach preserved temporal information crucial for forgery detection. Shuffling and multiple worker threads ensured efficient and randomized batch preparation. This multi-step pipeline emphasized both data integrity and visual diversity, allowing the models to learn meaningful manipulation cues while avoiding overfitting to superficial or dataset-specific features.

4.3 Models

We first evaluated several models that demonstrated limited generalization when tested on unseen datasets. Building upon the insights gained, we designed a final model architecture aimed at enhancing cross-domain robustness.

4.3.1 Swin Transformer

As part of our investigation, we explored a domain-adversarial approach to deepfake detection by integrating a Gradient Reversal Layer (GRL) into a multi-branch architecture. The model leveraged a Swin Transformer for spatial feature extraction,

a temporal transformer to capture inter-frame dynamics, and frequency-domain features obtained via FFT [64]. These complementary representations were fused and jointly used for binary classification and domain prediction through the GRL pathway, aiming to enforce domain-invariant learning[65]. However, despite the model's

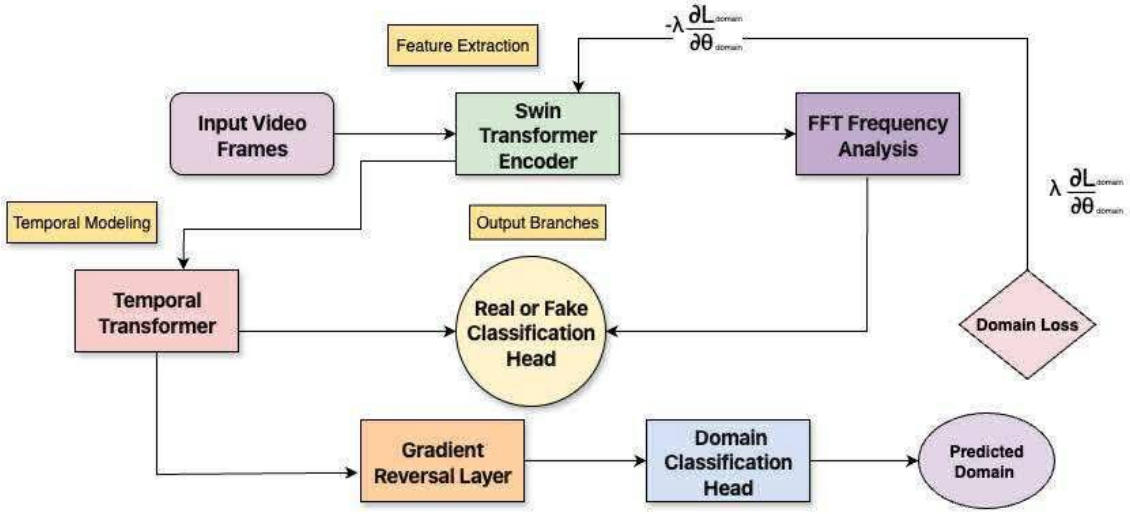

Figure 4.2: Swin Transformer Model working procedure

structural sophistication, it struggled to generalize effectively across datasets. In particular, the adversarial signal introduced by the GRL appeared insufficient to compensate for the wide variability in visual artifacts produced by different manipulation techniques. As a result, we ultimately chose not to include this configuration in our final model pipeline.
The expanded Swin Transformer architecture introduces two key enhancements aimed at refining decision-making: heuristic feature injection and confidence-aware attention.

Heuristic Feature Injection:

Heuristic Feature Injection: To guide the model toward manipulation-specific artifacts, we integrate domain-driven heuristic features—such as motion inconsistencies and facial warping cues—into the classification pipeline. These handcrafted signals are extracted in parallel with visual embeddings from the Swin backbone and merged before the final prediction layer [66]. By incorporating this additional information, the model benefits from an inductive bias that anchors its decisions to recognizable patterns often present in tampered content.

Confidence-Aware Attention:

For improved video-level inference, we adopt a confidence-aware attention mechanism that dynamically adjusts the contribution of each frame. Instead of averaging predictions uniformly, attention weights are assigned based on the confidence scores (softmax probabilities) of individual frames[67], allowing the model to prioritize more reliable inputs during aggregation.

4.3.2 ResNet-18

We implemented a deep learning pipeline based on a fine-tuned ResNet-18 architecture for binary classification of facial videos. ResNet-18 was chosen for its balance between depth and computational efficiency, helping to prevent overfitting on our relatively small dataset[68]. The model was initialized with ImageNet-pretrained weights to leverage transfer learning and accelerate convergence. The final fully connected layer was replaced with a single output node followed by a sigmoid activation. To enhance generalization, we further modified the architecture with a Gradient Reversal Layer (GRL), enabling adversarial domain alignment by encouraging domain-invariant feature learning. Training was performed using the Binary Cross-Entropy loss function. The model was trained for 20 epochs with early stopping and checkpointing based on validation loss. A batch size of 16 was set, likely constrained by GPU memory. Each training iteration involved forward passes, loss computation, backpropagation, and validation at the end of each epoch. Model performance was primarily evaluated using accuracy, with predictions threshold at 0.5 during validation to determine class labels.

4.3.3 XceptionNet

Xception (short for "Extreme Inception") is a deep convolutional neural network introduced by François Chollet in 2017 as an improvement over the Inception V3 architecture. Instead of Inception's multi-branch modules, Xception is built primarily on depthwise separable convolutions – a factorized form of convolution that treats spatial filtering and cross-channel filtering separately, greatly reducing model complexity. The Xception network is organized into three main stages (entry flow, middle flow, and exit flow) comprising 36 convolutional layers in total. In the entry flow, initial convolutions with increasing filters (e.g. 32, 64) and pooling downsample the input while extracting low-level features. The middle flow is the core of Xception: a sequence of 8 identical modules of depthwise separable convolutions (with ReLU and batch normalization) that iteratively refine higher-level feature representations. Finally, the exit flow uses deeper separable convolutions (up to 2048 filters) followed by global average pooling and a dense layer for classification[69]. Residual connections are included around these stacks to facilitate training of the deep network. Overall, Xception's use of separable convolutions means it achieves efficiency similar to Inception V3 but with a more streamlined, single-path architecture. This robust backbone has made Xception a popular choice for transfer learning in image classification and detection tasks, including deepfake detection.

Full Fine-Tuning of the Pretrained Model

Using a pretrained Xception model as the backbone, one training approach we employed was full fine-tuning, meaning all layers of the network (the Xception base and the new classifier layers) are trained on the deepfake dataset from the start. In a transfer learning context, fine-tuning refers to unfreezing the pretrained model's layers so that their weights are gradually adjusted to better fit the new task. This leverages the rich feature extraction capability learned from a large source dataset (ImageNet) and adapts it to our specific classification problem. The advantage of full fine-tuning is that the model's high-level features can be optimally tuned for

the nuances of deepfake detection, potentially improving performance once sufficient training has occurred. This approach is especially effective if the new dataset is large or somewhat similar in nature to the original dataset that the backbone was trained on. However, because the Xception backbone has 22.9 million parameters [69], updating all of them on a limited deepfake dataset can risk overfitting or destabilizing previously learned features if not carefully managed (e.g. using a lower learning rate). In practice, fine-tuning all layers from the beginning means the model can start to "forget" some of the generic visual features and tune them to peculiarities of the training data. This can yield excellent results on the training set, but one must monitor validation performance to ensure the model is not overfitting to the new data distribution. When done properly (often with techniques like a smaller learning rate and sufficient regularization), full fine-tuning allows the pretrained Xception to fully adapt its feature hierarchies to the deepfake detection task, improving its discriminative power on subtle artifacts that distinguish fake from real videos.

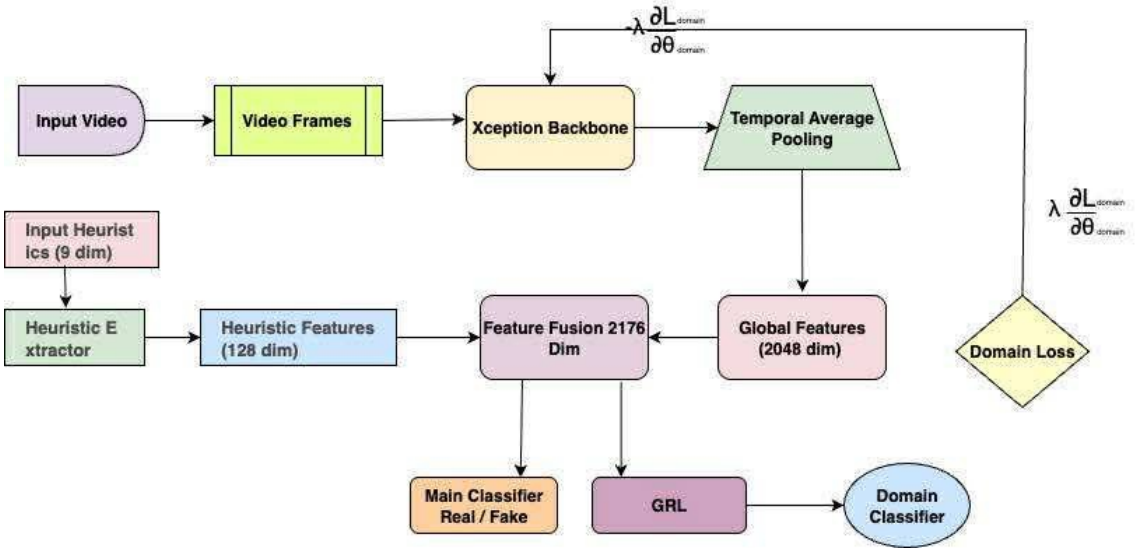

Figure 4.3: XceptionNet Model working procedure

Freezing the Backbone Initially (Transfer Learning Strategy)

An alternative strategy we explored was initially freezing the Xception backbone for a few epochs, then gradually unfreezing it for fine-tuning. Freezing means keeping the pretrained convolutional base un-trainable so that its weights are not updated during those initial training epochs. In this phase, only the newly added layers (for example, the classifier head and any additional modules) learn from the data. The rationale behind this approach is to preserve the valuable general features the backbone has learned (edges, textures, facial structures, etc.) and first allow the new layers to adapt to the task without perturbing the base filters. This has several benefits: it prevents overfitting early on (since the millions of parameters in Xception are not allowed to overly tune to the small dataset immediately), and it speeds up training because backpropagation only updates the top layers. By leveraging the generalized representations in the frozen convolutional layers, the model effectively

uses Xception as a fixed feature extractor initially. After this "warm-up" period (in our case, 3 epochs), we then unfroze the backbone i.e., allow the Xception layers to start training and continue training the entire model end-to-end (often with a reduced learning rate for stability). This two-step process (sometimes called gradual fine-tuning) is a common practice in transfer learning. It allows the new classifier to learn task-specific features first, then slowly fine-tunes the deeper layers to improve higher-level feature alignment with the task.

Additional Enhancements for Better Generalization

Beyond the choice of fine-tuning versus freezing, our model architecture included modifications to the Xception backbone setup intended to improve generalization to different data sources and types of forgeries. First, we incorporated a domain-adversarial training component using a Gradient Reversal Layer (GRL) and an auxiliary domain classifier. In practice, the feature representations output by the Xception backbone (after global average pooling) were fed not only to the main deepfake/real classifier, but also to a domain classifier that tries to predict which dataset or source the input came from. Crucially, this branch is connected through a GRL, which multiplies its gradients by -1 during backpropagation. This setup implements the approach of Ganin et al. (2015), which encourages the learned features to be domain-invariant [70]. In other words, during training the model is pressured to learn feature embeddings that are discriminative for detecting fakes vs reals, while at the same time being unable to distinguish the data source or domain. By adversarially forcing the backbone to "fool" the domain classifier, the model learns more generalizable features that should perform well on unseen domains. This is especially important in deepfake detection, where the evaluation data may come from a different creation method or video source than the training data. Secondly, we augmented the backbone with an auxiliary feature branch to integrate hand-crafted heuristic features that might capture artifacts not easily learned by the CNN. In our implementation, we had a set of 9-dimensional heuristic features (e.g., statistics or signals from the video frames) which we passed through a small fully-connected sub-network and then concatenated with the Xception backbone's output features. Recent research suggests that combining conventional deep CNN features with handcrafted features (for example, frequency-domain artifacts from manipulated images) can provide richer information and boost deepfake detection performance [71]. By fusing these heuristic features, the model can utilize cues like compression artifacts or physiological signals that complement the visual patterns captured by Xception. This multi-stream approach aims to improve robustness: the deep neural network learns complex patterns in the pixel data, while the heuristic input can highlight specific anomalies (e.g., inconsistencies in frequency spectra or other forensic fingerprints) that help the model generalize to different types of fakes.

4.3.4 Final Xception-Based Multi-Branch Model

Architectural Overview

We propose a deepfake detection framework that is designed to maintain a decent AUC in cross-dataset generalization. The architecture encompasses a multi-branch design that attempts to simultaneously promote content preservation, feature disen-

tanglement and domain-invariant representation learning. The model comprises four key components: a shared encoder, two distinct feature projectors, a reconstruction decoder, and a domain-adversarial module. The final prediction is computed via an ensemble of the two classifier branches. The overall pipeline is illustrated in Figure below:

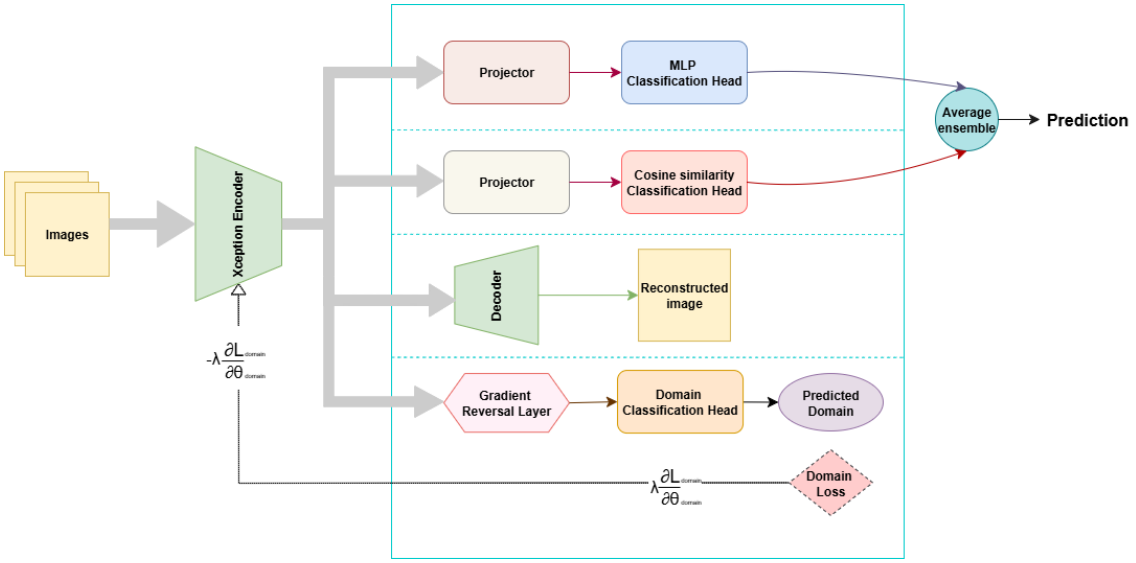

Figure 4.4: Final Model Architecture

Encoder

Our model uses a pretrained Xception network as the encoder to extract high-level features from input video frames. Given an input image x, the encoder extracts a deep feature representation $f = \mathcal{E}(x) \in \mathbb{R}^d$. The extracted features are then processed by two projection heads, a decoder module, and a domain-adversarial module parallelly.

Feature Projection and Classification

The projectors project the encoded features to a 512-dimensional space. Both projectors operate on all samples (real and fake), but are guided by divergent objectives to encourage feature disentanglement. To promote this disentanglement, we apply a disentanglement loss that penalizes overlap between the two projected feature spaces. This regularization reduces redundancy and encourages each branch to specialize on distinct feature subspaces. The two disentangled subspaces are as follows:

- Feature Space 1: $f_1 = \mathcal{P}_r(f)$, optimized for classification via an MLP classifier C_{MLP}.
- Feature Space 2: $f_2 = \mathcal{P}_f(f)$, where classification is performed using cosine similarity classifier head $C_{\text{cosine\_sim}}$ against two learnable prototypes representing real and fake classes.

Decoder for Content Preservation

To regularize the encoder and preserve content-rich features, we incorporate a decoder $\mathcal{D}$ that reconstructs the original image from f. At first, the decoder converts the encoded feature vector to a low-resolution spatial map of (4x4x512) dimension. After this, the spatial map is progressively upsampled by using a series of transposed convolution blocks with BatchNorm and undergoes ReLU activation. Finally, a reconstructed image of dimension 256x256 is created with the final transposed convolution layer along with Tanh activation. The reconstruction loss is defined as the Mean Squared Error (MSE) between the reconstructed image $\hat{x} = \mathcal{D}(f)$ and the denormalized original input x. When used alongside the Gradient Reversal Layer (GRL), the reconstruction task helps counterbalance the adversarial pressure to remove domain-specific cues, acting as a regularizer that promotes robust and generalizable feature learning.

Domain-Adversarial Module

To learn domain-invariant features, a MLP domain classification head is attached to the encoder output via a Gradient Reversal Layer (GRL) as proposed [72] This classifier predicts the dataset origin of each input (e.g., FF++, VeriFake), while the GRL inverts gradients during backpropagation, encouraging the encoder to learn domain-invariant features. The domain loss is computed via cross-entropy, and its adversarial nature aligns feature distributions across domains.

Disentanglement Loss

To ensure the projected real and fake features lie in orthogonal, independent subspaces, we introduce a composite disentanglement loss comprising:

- Orthogonality Loss, $\mathcal{L}_{\text{ortho}}$: Encourages low cosine similarity between f_1 and f_2.
- Covariance Loss, $\mathcal{L}_{\text{cov}}$: Minimizes the Frobenius norm of the cross-covariance matrix between f_1 and f_2.
- Mutual Information Loss, $\mathcal{L}_{\text{mi}}$: Penalizes statistical dependence through a correlation-based approximation of mutual information.

These losses jointly reduce representation overlap and force the projectors to preserve discriminative information in both classification branches.

$$\mathcal{L}_{\text{disentangle}} = \mathcal{L}_{\text{ortho}} + \mathcal{L}_{\text{cov}} + \mathcal{L}_{\text{mi}}$$

Overall Loss Function

The total objective is a weighted combination of classification, domain, reconstruction, contrastive, and disentanglement losses.This combined loss function makes sure to learn discriminative features, preserve spatial and semantic consistency, generalizes across domains, and avoids feature entanglement between C_{MLP} and $C_{\text{cosine\_sim}}$:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{MLP\_cls}} + \mathcal{L}_{\text{cosine\_cls}} + \mathcal{L}_{\text{domain}} + \alpha\mathcal{L}_{\text{recon}} + \beta_{\text{cont}}\mathcal{L}_{\text{contrast}} + \mathcal{L}_{\text{disentangle}}$$

where:

- $\mathcal{L}_{\text{MLP\_cls}}$ and $\mathcal{L}_{\text{cosine\_cls}}$: Binary cross-entropy losses from C_{MLP} and $C_{\text{cosine\_sim}}$.
- $\mathcal{L}_{\text{domain}}$: Binary cross-entropy loss for the domain classifier.
- $\mathcal{L}_{\text{recon}}$: Mean Squared Error (MSE) reconstruction loss between $\hat{x}$ and x.
- $\mathcal{L}_{\text{contrast}}$: Cosine embedding loss between intra-class feature pairs.
- $\mathcal{L}_{\text{disentangle}}$: Composite loss for disentanglement regularization.

Hyperparameters were set to $\alpha_{\text{grl}} = 3.0$, and $\beta_{\text{cont}} = 0.1$.

Inference and Ensemble Prediction

At inference time, the model produces outputs from two branches: The first branch generates class logits using an MLP classifier, while the second branch produces similarity scores via a cosine classifier. To enhance prediction stability and accuracy, we ensemble the outputs from both branches by averaging them. This ensemble, computed externally, leverages the complementary strengths of both representations.

4.4 Experimental setup

All models were implemented using PyTorch and trained on a single NVIDIA GPU. In this section, we describe the training protocols used for the Swin Transformer, ResNet, XceptionNet, and our final multi-branch Xception-based model. Each configuration was optimized for cross-dataset generalization and stability during training.

4.4.1 Swin Transformer-Based Models

For the Swin Transformer architecture, training was conducted using the AdamW optimizer with an initial learning rate of 2e-5 and weight decay of 1e-5. A ReduceLROnPlateau learning rate scheduler was used to reduce the learning rate by a factor of 0.5 if validation loss did not improve over 3 epochs. Models were trained for a maximum of 20 epochs, with early stopping activated if validation accuracy did not improve for 5 consecutive epochs. The batch size was set to 4. To encourage generalization, label smoothing with a factor of 0.1 was applied using the CrossEntropyLoss function. In domain-adaptive versions, a domain classification head and Gradient Reversal Layer (GRL) were introduced, with the domain loss weighted at 0.5. Throughout training, the model was evaluated on a held-out validation set at each epoch. The model checkpoint with the highest validation accuracy was retained for final evaluation. Training and validation loss and accuracy curves were plotted to monitor convergence and potential overfitting.

4.4.2 ResNet and XceptionNet-Based Models

For the ResNet and XceptionNet backbones, the AdamW optimizer was used with an initial learning rate of 2e-4 and weight decay of 5e-5. A StepLR scheduler was applied to reduce the learning rate by a factor of 0.5 every 10 epochs. As with the Swin Transformer, training ran for a maximum of 20 epochs, with early stopping based on

validation accuracy (patience of 5 epochs). The batch size was 4. Label smoothing with a factor of 0.1 was incorporated into the CrossEntropyLoss to improve robustness. In experiments involving domain adaptation, a GRL-based domain classifier was included, with the domain loss weighted at 0.1. Model checkpoints were selected based on validation accuracy, and training dynamics were monitored using loss and accuracy plots for both training and validation sets.

4.4.3 Xception-Based Multi-Branch Model

We evaluate our model using custom and public deepfake datasets. We used FaceForensics++ (FF++) for training as it contains diverse fake samples which are manipulated using 6 different types of deepfake generation techniques. For domain adversarial training, seven domain labels are assigned: one for real samples and six corresponding to six different types of fake samples. Finally the prediction is done using DFDC, Celeb-DF (v1), and our own Verifake dataset to observe cross-dataset generalization performance.
The model was trained using the Adam optimizer with a learning rate of 2×10^{-4} and a batch size of 8 on an NVIDIA GPU GTX 1650 super with 4gb VRAM for up to 30 epochs, applying early stopping if the validation ensemble accuracy did not improve for 5 consecutive epochs. To guide the learning process, several hyperparameters were employed: the gradient reversal layer was set to $\lambda_{\text{grl}} = 0.6$ to encourage domain-invariant feature learning; orthogonality regularization with $\lambda_{\text{ortho}} = 1 \times 10^{-4}$; covariance minimization with $\lambda_{\text{cov}} = 1 \times 10^{-3}$; and mutual information minimization with $\lambda_{\text{mi}} = 1 \times 10^{-3}$ for feature disentanglement.
The dataset was split per domain, with 80% of samples used for training, 10% for validation, and 10% for testing. Each sample contributed 10 cropped face frames for training. During each epoch, the model was trained on the training set and evaluated on the validation set, with the best-performing model (based on validation ensemble accuracy) saved for final evaluation.
Comprehensive evaluation was performed using accuracy, AUC, and domain classification performance. Training dynamics were visualized through loss and accuracy curves to assess learning stability and convergence. To ensure fair evaluation and assess generalization, the VeriFake, CelebDF, and DFDC datasets were strictly excluded from training and used only for testing.

4.5 Ablation Study of the final model

We conduct ablations to quantify the contribution of individual components. Results (on Celeb-DF) are summarized below:
We conduct an ablation study to evaluate the individual and combined effects of the decoder, gradient reversal layer (GRL), and disentanglement module on both in-domain and cross-domain performance. The baseline model, consisting of only an encoder and cosine classifier head, performs well on in-domain data but generalizes poorly across domains. Introducing the MLP classifier and disentanglement improves both in-domain and cross-domain performance, indicating better feature separation. While the GRL slightly reduces in-domain performance due to its regularization effect, it significantly enhances cross-domain robustness by encouraging domain-invariant representations. The full model, which integrates the decoder, strikes a balance—achieving strong in-domain performance

Model Variant	Decoder	GRL	Disentanglement	In-domain AUC	Cross-domain AUC
Encoder + MLP Classifier Head	×	×	×	0.9653	0.6326
+ Cosine Similarity Head	×	×	✓	0.9802	0.6738
+ GRL	×	✓	✓	0.9527	0.7554
Full Model	✓	✓	✓	0.9568	0.8183

Table 4.1: Performance comparison of model variants on in-domain and cross-domain AUC.

while substantially improving cross-domain generalization, highlighting the complementary roles of reconstruction, disentanglement, and adversarial domain learning.

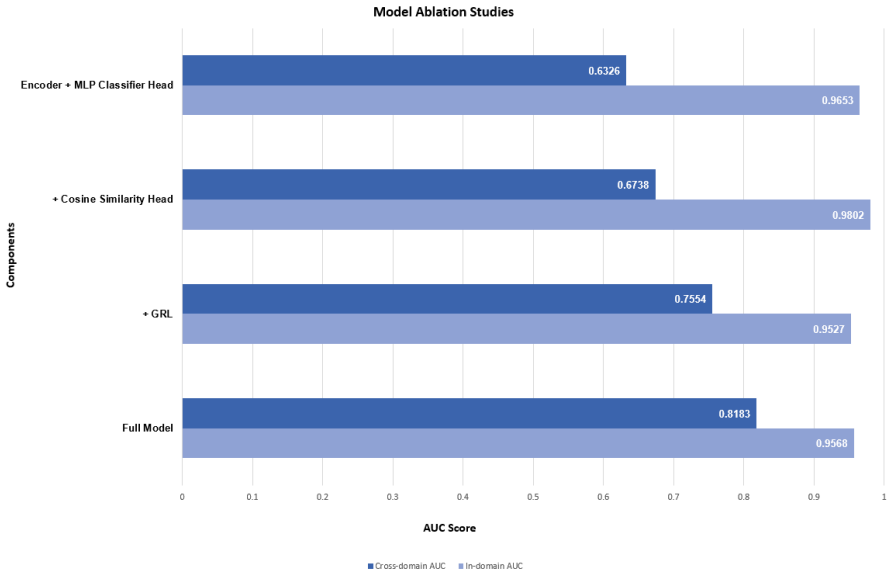

Figure 4.5: Ablation Study of final model

An ablation study is conducted in order to test the individual and combined influence of decoder, gradient reversal layer (GRL), and disentanglement module on both in-domain and cross-domain performance. When the baseline model consists of only an encoder and cosine classifier head, it performs well on in-domain data but generalizes poorly across domains. Adding the MLP classifier and disentanglement improves both in-domain and cross-domain performance, leading to better feature separation. Although the GRL slightly lowers in-domain performance because of its regularization effect, it significantly enhances cross-domain robustness by encouraging domain-invariant representations. The complete model, which includes the decoder, achieves strong in-domain performance and greatly improves cross-domain generalization. This shows the supporting roles of reconstruction, disentanglement, and adversarial domain learning.

Chapter 5

Evaluation and Results

This chapter presents a comprehensive evaluation of multiple deepfake detection models across several benchmark datasets. The analysis aims to quantify performance in both intra-dataset and cross-dataset contexts, assess generalization capabilities, and interpret trends based on architectural choices and training configurations. Evaluation metrics include Accuracy, Precision, Recall, F1-score, and the Area Under the ROC Curve (AUC), with an emphasis on AUC for model ranking and comparison. Performance insights are presented model-wise, complemented by a data ablation study on the best-performing architecture.

5.1 Evaluation Metrics

To assess and compare the performance of the developed models, we employed several standard evaluation metrics: accuracy, precision, recall, F1-score, and Area Under the ROC Curve (AUC-ROC). Each metric provides a different perspective on the classifiers' effectiveness, and together they offer a comprehensive view of model performance. Key metrics are defined as follows:

5.1.1 Accuracy

Accuracy is the simplest evaluation metric, defined as the proportion of all predictions that the model got correct. While intuitive, accuracy can be misleading as a sole measure, especially in imbalanced datasets. A classifier can attain high overall accuracy by merely predicting the majority class every time, yet such a model would fail to detect the minority class of interest. In other words, when the class distribution is skewed, accuracy often reflects the predominance of the negative (majority) class rather than true predictive performance.

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{5.1}$$

Prior research has noted that simple classification accuracy is often a poor metric for evaluating classifiers under these conditions. For example, Provost and Fawcett demonstrated that accuracy alone can be a problematic indicator, which motivated increased use of ROC analysis for performance evaluation on skewed data [73]. Therefore, although we report accuracy for completeness, we do not rely on it as the primary yardstick of model success in this anomaly detection scenario. Instead, accuracy serves as a baseline sanity check while we give more weight to metrics that account for class imbalance and error trade-offs.

5.1.2 Precision and Recall

Precision and recall are more informative metrics in the context of class-imbalanced anomaly detection. Precision (also known as positive predictive value) is the fraction of model-predicted anomalies that are actually anomalous (true positives divided by all positive predictions), whereas recall (also known as sensitivity or true positive rate) is the fraction of true anomalies that the model successfully detected (true positives divided by all actual positives). In formula terms:

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

These metrics focus on the performance of the positive (anomaly) class, which is our minority class of interest. High precision means that when the model flags an anomaly, it is usually correct – important in scenarios where false alarms are costly.

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

High recall means the model catches most of the actual anomalies – critical when missing a positive instance has serious consequences. There is an inherent trade-off between precision and recall: a more lenient model (lower threshold) will catch more true anomalies (higher recall) but also raise more false alarms (lower precision), whereas a stricter model (higher threshold) will fire only on the most confident anomalies (improving precision) at the expense of missing some real ones. Because of this trade-off, one typically cannot maximize both precision and recall simultaneously; the optimal balance depends on the application's priorities. However, both precision and recall are inherently threshold-dependent; their values can change substantially with different decision boundaries. Since optimal thresholds may vary across datasets due to differences in class balance and distribution of prediction scores, direct comparisons of precision and recall between models on different datasets can be misleading.

5.1.3 F1 Score

The F1-score is the harmonic mean of precision and recall, offering a single metric that balances both aspects. It is calculated :

$$\text{F1} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

and reaches its maximum value of 1 only when precision and recall are both high. This makes F1 particularly useful for evaluating imbalanced datasets, where one does not want to optimize precision at the complete expense of recall or vice versa. A high F1-score indicates that the model achieves a good trade-off: it identifies a large portion of the anomalies (high recall) while also keeping false positives reasonably low (high precision). Conversely, if either precision or recall is very low, the F1-score will be low – which correctly flags that the model's performance is deficient in one respect despite a possibly strong showing in the other metric. In our results, we report the F1-score for each model as a summary of its precision/recall balance on the anomaly-detection task. This helps compare models on a single scale that accounts for both types of error. It is worth noting that while F1 is a convenient aggregate metric, it still corresponds to a specific decision threshold. Thus, we use F1 in conjunction with a threshold-independent metric (AUC) to fully characterize model performance.

5.1.4 AUC-ROC (Area Under the ROC Curve)

Receiver Operating Characteristic (ROC) curves plot the true positive rate (recall) against the false positive rate for a binary classifier as the decision threshold is varied. The Area Under the ROC Curve (AUC) condenses this entire curve into a single number, ranging from 0.5 (no better than chance) to 1.0 (perfect separation). Among the evaluation metrics, we place special emphasis on AUC for model comparison. We chose AUC as a primary metric for several reasons, detailed below:

- **Threshold-Independent, Holistic Performance:** AUC provides an aggregate measure of a classifier's performance across all possible classification thresholds, rather than at one fixed operating point. It essentially averages the model's sensitivity and specificity trade-offs over the full range. In practical terms, AUC tells us how well the model ranks positives versus negatives without committing to a particular cutoff [74]. This is advantageous because it evaluates the inherent discriminative capacity of the model independent of any specific decision boundary. In fact, the AUC can be interpreted as the probability that a randomly chosen positive instance will be scored higher by the model than a randomly chosen negative instance [74]. By considering the entire ROC curve, AUC captures performance in a threshold-independent manner, making it a more comprehensive metric than accuracy, precision, or F1 (each of which is computed at a single threshold).

- **Robust to Class Imbalance:** AUC is relatively insensitive to class skew, which is critical in our context of rare anomalies. Since the ROC curve is calculated using rates (TPR and FPR) that normalize over the size of each class, it does not depend on the actual ratio of positive to negative instances [75]. In other words, if the test set's class distribution changes, the ROC curve (and thus AUC) remains the same because it looks only at the relative ranking of predictions, not the absolute counts. Metrics like accuracy or even precision can be heavily biased by the majority class frequency, but AUC evaluates the model's ability to distinguish between classes regardless of how many positives or negatives are present. This property makes AUC well-suited for imbalanced data scenarios – it will not be artificially inflated by a large number of true negatives, for example. In our video anomaly detection problem, where anomalous events are far outnumbered by normal frames, accuracy can become nearly meaningless (as discussed above), whereas AUC remains a fair indicator of how well the model can separate anomalous from normal instances.

- **Effective Model Comparison:** AUC is widely used as a summary statistic to compare classifiers, and it has been recommended by multiple studies as a preferable alternative to accuracy for model evaluation. One reason is that AUC considers the ranking of outputs and thus tends to be more discriminating when comparing models. A higher AUC unambiguously indicates that on average, one model is better at scoring positive cases above negatives than another. Unlike accuracy, which might drastically change with a small shift in threshold or be dominated by class imbalance, AUC provides an objective basis to rank models on overall performance. Indeed, Bradley (1997) found that AUC has desirable statistical properties (such as a lower standard error with more test samples) and concluded that "AUC should be used in preference to overall accuracy for single-number evaluation" [76]. Similarly, formally proved that AUC is a statistically consistent and more discriminating measure than accuracy for comparing learning algorithms [77]. This evidence underlines that when we want to select the best model, AUC is often a more reliable indicator than accuracy or F1 alone. In our analysis, using AUC allowed us to fairly compare models on an "apples-to-apples" basis, since it evaluates the full range of their

sensitivity-specificity behavior.

In light of these advantages, the AUC-ROC is adopted as a primary metric for model selection and evaluation in this thesis. Accuracy, precision, recall, and F1-score are also reported for each model (at the chosen operating threshold) to provide a detailed view of performance; however, AUC serves as the most comprehensive single measure of a model's ability to distinguish anomalous from normal events across all possible thresholds. A higher AUC generally indicates stronger discriminative capability in ranking positive instances above negative ones, regardless of the specific decision boundary. In our experiments, AUC provided a consistent and comparable measure across models of different architectures and output granularities—some producing frame-level scores later aggregated to video-level predictions, others outputting a single prediction per video segment. By computing the AUC-ROC from each model's confidence scores on the test set, we ensured that this metric reflects their full threshold-dependent behavior, enabling fair comparisons. While absolute AUC values vary across datasets and models, this measure allows performance differences to be interpreted in a threshold-independent and class-imbalance–resilient way. In this context, AUC is used not to overstate a single model's dominance, but as a principled basis for assessing relative performance and generalization potential in cross-dataset deepfake detection.

5.2 Experimental Results

5.2.1 Intra-Dataset Results

To establish a baseline for each model's core capacity to detect manipulation, we begin with intra-dataset evaluation: training and testing each model on the same dataset. This eliminates domain shift and isolates a model's ability to learn dataset-specific cues.

Model	**Dataset**	**Test Acc**	**Precision**	**Recall**	**F1-Score**	**AUC**
XceptionNet	Celeb-DF (814 samples)	0.9184	0.91	0.91	0.91	0.9337
Swin Transformer		0.9479	0.975	0.971	0.973	0.987
ResNet-18		0.9271	0.94	0.82	0.88	0.9712
XceptionNet	FF++ (7000 samples)	0.8932	0.93	0.95	0.94	0.985
Swin Transformer		0.9679	0.96	0.95	0.955	0.971
ResNet-18		0.9165	0.90	0.82	0.86	0.94
XceptionNet	VeriFake (1000 samples)	0.9274	0.92	0.91	0.91	0.9337
Swin Transformer		0.8980	0.89	0.88	0.885	0.92
ResNet-18		0.8673	0.86	0.84	0.85	0.9005

Table 5.1: Intra-Dataset Performance Summary

The intra-dataset evaluation results across Celeb-DF, FF++, and VeriFake datasets provide valuable insight into the behavior of three baseline models: XceptionNet, Swin Transformer, and ResNet-18. Each model was trained and tested on the same dataset to understand how well it can capture manipulation-specific patterns under controlled conditions. Performance was assessed using five standard classification metrics: Accuracy, Precision, Recall, F1-score, and AUC (Area Under the ROC Curve), providing a multidimensional

view of each model's strengths and limitations. To visualise the accuracy in graphical method:

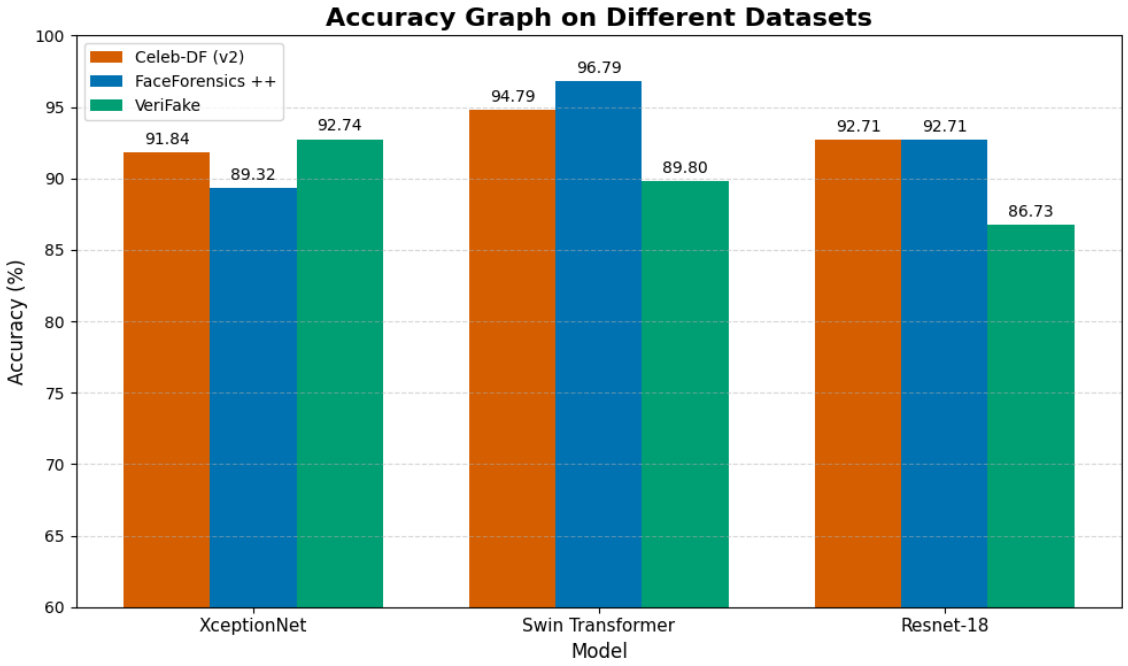

Figure 5.1: Accuracy on intra-dataset testing

5.2.2 Overall Model Comparison

Among the three models, Swin Transformer consistently emerges as the top performer in terms of raw accuracy and AUC. It achieves a remarkable 96.79% accuracy on FF++ and the highest overall AUC of 0.987 on Celeb-DF. These metrics suggest that Swin Transformer is highly effective at capturing spatial inconsistencies and subtle facial artifacts within known domains. The model also maintains a well-balanced precision-recall profile, with F1-scores above 0.95 on both Celeb-DF and FF++, highlighting its reliability across both strict and lenient classification thresholds. XceptionNet, a deep convolutional neural network architecture, offers slightly more balanced but modestly lower results. On FF++, it achieves an AUC of 0.985 with strong recall (0.95) and precision (0.93), resulting in an F1-score of 0.94. On Celeb-DF and VeriFake, it consistently reports an accuracy of around 91.84%–92.74%, with AUC values at 0.9337, matching or closely trailing the Swin Transformer. This performance stability across datasets suggests that XceptionNet is more resilient to intra-dataset variations, making it a solid candidate for scenarios that demand consistent behavior over peak performance. In contrast, ResNet-18, while competitive, consistently lags behind the other two models. Though it achieves an accuracy of 92.71% on Celeb-DF and 91.65 FF++, its recall drops to 0.82, which significantly lowers its F1-score (0.88 and 0.86 respectively). This indicates that ResNet-18 tends to miss more true positives than the other models, which may be attributed to its shallower architecture and limited ability to capture high-level spatiotemporal relationships. On VeriFake, ResNet-18's performance drops further, with an accuracy of 86.73% and the lowest AUC among all results (0.9005), suggesting it is less suited to datasets with noisier or subtler manipulation cues.

5.2.3 Dataset-Wise Trends and Observations

From a dataset perspective, FF++ emerges as the most 'model-friendly' dataset, consistently supporting high recall values across models. This suggests that the manipulations in FF++ are relatively more detectable, possibly due to clearer visual artifacts or higher quality annotations. Celeb-DF, while slightly more challenging, still allows for strong performance, particularly from transformer-based architectures. VeriFake, on the other hand, proves to be the most challenging among the three. Although XceptionNet still performs reliably, both Swin Transformer and ResNet-18 show reduced scores, especially in AUC. This could reflect either less distinct manipulation artifacts, which traditional convolutional backbones like ResNet-18 fail to generalize to effectively.
These results collectively underscore several strategic insights. First, transformer-based models like Swin outperform CNN-based models when trained on datasets with well-defined manipulation features, particularly due to their ability to model long-range dependencies and finer spatial features. Second, models trained and evaluated on FF++ tend to show the highest recall, reinforcing its utility as a benchmark for detection sensitivity. Third, the uniformity and high intra-dataset performance should not be conflated with true robustness; the narrow performance gap within datasets conceals each model's fragility to domain shifts, which is explored in the cross-dataset evaluation. Finally, while intra-dataset evaluation is useful for establishing upper-bound performance, it offers limited insight into a model's practical utility. High performance in this setting often reflects memorization of dataset-specific cues rather than the learning of generalizable deepfake detection features.

5.2.4 Cross Dataset Results

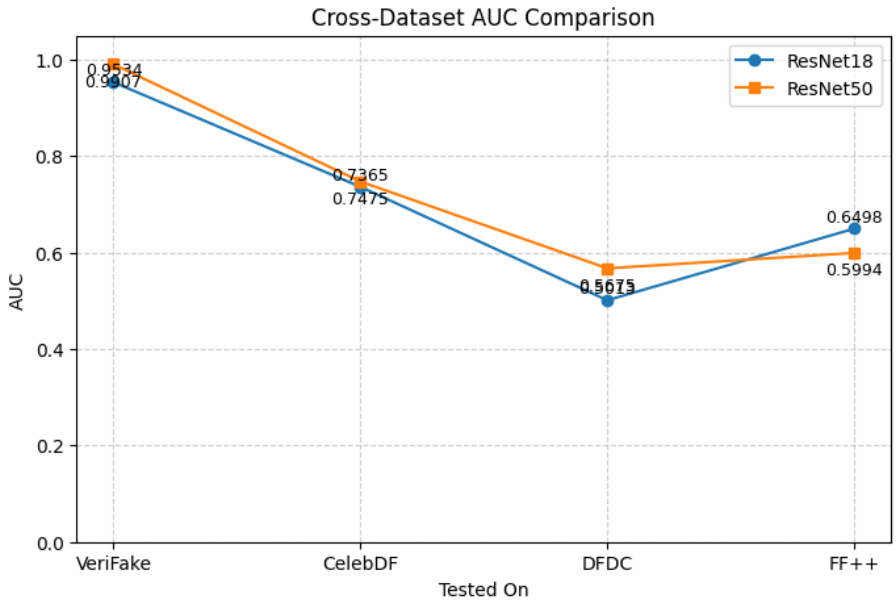

Figure 5.2: AUC analysis of cross dataset testing

Generalizing across datasets remains one of the most persistent challenges in deepfake detection. Our experiments highlight how even well-performing models can fall short when tested on unfamiliar domains. Most of our earlier models were trained primarily on VeriFake, a custom dataset we constructed. While these models showed promising results on in-distribution data, their performance dropped significantly when evaluated on external

Backbone	Trained On	Tested On	Test Acc	AUC
w/o Freezing	VeriFake (1000 samples) + Celeb-DF (50 samples)	VeriFake	0.9422	0.9795
		Celeb-DF	0.7500	0.7635
		DFDC	0.2293	0.5090
		FF++	0.5186	0.5513
	FF++ (2200 samples)	Celeb-DF	0.6098	0.6150
		FF++	0.8591	0.9452
		DFDC	0.5650	0.5975
		VeriFake	0.4980	0.5169
w/ Freezing	VeriFake (1000 samples) + Celeb-DF (50 samples)	VeriFake	0.8972	0.9243
		Celeb-DF	0.7066	0.6748
		DFDC	0.1884	0.5030
		FF++	0.5019	0.5447
	FF++ (2200 samples)	Celeb-DF	0.4904	0.5531
		FF++	0.7318	0.7987
		DFDC	0.6150	0.6310
		VeriFake	0.4551	0.4093

Table 5.2: XceptionNet - Cross-Dataset Results

Trained On	Tested On	Test Acc	AUC
VeriFake(1000 samples) + Celeb-DF(50 samples)	VeriFake	0.9691	0.9534
	Celeb-DF	0.6739	0.7365
	DFDC	0.1575	0.5013
	FF++	0.5875	0.6498

Table 5.3: ResNet-18 - Cross-Dataset Results

Trained On	Tested On	Test Acc	AUC
VeriFake(1000 samples) + Celeb-DF(50 samples)	VeriFake	0.9669	0.9907
	Celeb-DF	0.6534	0.7475
	DFDC	0.5100	0.5675
	FF++	0.6075	0.5994

Table 5.4: ResNet-50 - Cross-Dataset Results

Variant	Trained On	Tested On					
		VeriFake		Celeb-DF		DFDC	
		Acc	AUC	Acc	AUC	Acc	AUC
Swin+Temp	VeriFake + (1000 samples) + Celeb-DF (50 samples)	0.9347	0.9327	0.5906	0.6145	0.4921	0.5081
Swin+Temp+GRL		0.9556	0.9755	0.6318	0.6422	0.5100	0.5614
Swin+Temp+GRL +Penalizer		0.9205	0.9924	0.5841	0.7257	0.4900	0.5414
Swin+Temp+GRL	Celeb-DF (814 samples)	0.5129	0.5566	0.9728	0.9961	0.5650	0.5984
Swin+Temp+GRL	FF++ (2200 samples)	0.4671	0.4973	0.6683	0.6101	0.4800	0.4762

Table 5.5: Swin Transformer Variants - Multi-Dataset Performance

datasets like DFDC and Celeb-DF. To address this, we attempted a minor intervention adding a small sample of Celeb-DF videos (just 50) to the training data, to see if even slight increases in dataset diversity could help. In some cases, this tweak improved the model's robustness on Celeb-DF marginally, but it still did not offer a meaningful improvement in generalization to DFDC. This revealed a key limitation: VeriFake alone, even when augmented with limited external samples, was not sufficient to help models adapt to entirely new domains. The models after being on FF++ did not yield improvements. While Xception saw a modest gain on DFDC, increasing from 0.503 to 0.6310 AUC with frozen weights and from 0.509 to 0.5975 AUC with full fine-tuning. However, performance on VeriFake dropped sharply, and results on Celeb-DF also got worse. Resnet-50 and Swin Transformer model did not even show improvement on the DFDC dataset. It may have happened because FF++ contains seven distinct manipulation types, the models faced difficulty adapting to all of them simultaneously, which reduced their overall deepfake detection capability.
Architectural enhancements like the use of Gradient Reversal Layers (GRL) and heuristic feature inputs offered some improvement. GRL in particular showed potential in helping the model unlearn dataset-specific cues, leading to small boosts in performance when generalizing. However, these changes did not fully close the gap, cross-dataset performance remained unstable.

5.3 Result Analysis

5.3.1 Feature Space Analysis: Why Models Struggled on DFDC

Using t-SNE on the penultimate layers of each backbone, focus was turned to feature space visualizations for having a better understanding on the underperformance on DFDC dataset. These plots capture an important pattern, that is, in the majority of VeriFake trained models, the real and fake embeddings of DFDC formed tight clusters . These clusters were visibly separated from the other tested datasets and differentiated poorly within themselves.

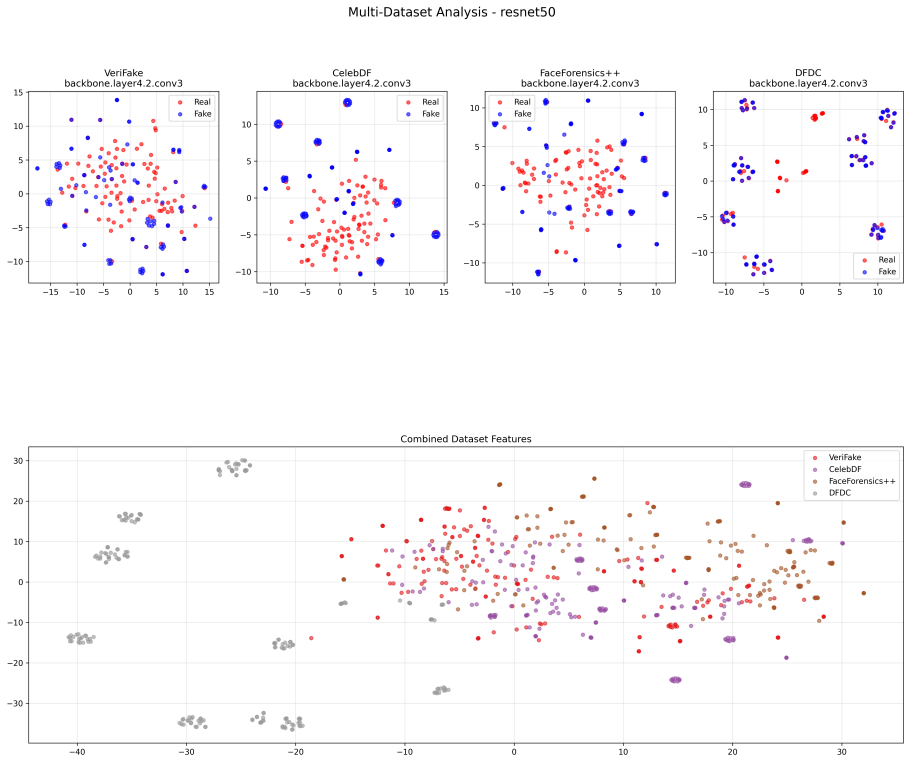

Figure 5.3: Video dataset analysis of ResNet-50

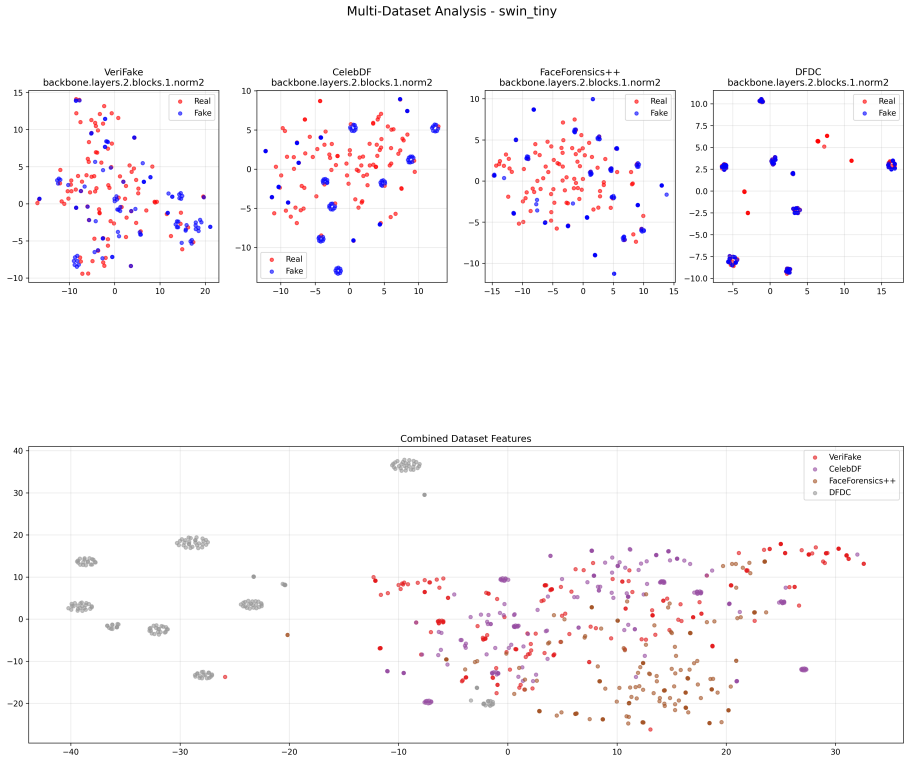

Figure 5.4: Video dataset analysis of Swin Tiny

To put it simply, the models were unable to learn generalizable features across domains. The DFDC dataset was affected the most as it consistently showed low intra-class separation and high intra-dataset divergence. For instance, models like ResNet-50 and Swin

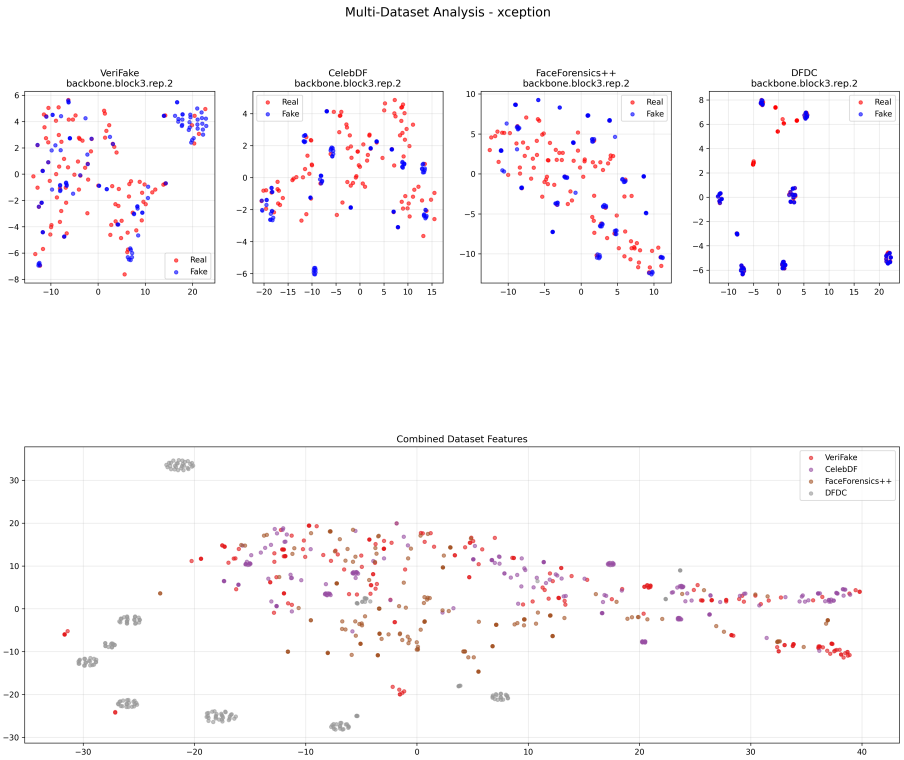

Figure 5.5: Video dataset analysis of XceptionNet

Transformer, the DFDC samples are clustered away from other datasets and tend to overlap between real and fake samples. This anomaly indicates that the features learned from VeriFake were not meaningful for DFDC's data distribution. On the other hand, datasets that share more visual characteristics with our custom dataset Verifake, FaceForensics++ and Celeb-DF showed comparatively better class separation. But in case of DFDC being a diversified, high-resolution and differently compressed dataset, it was often handled as an entirely different distribution by the models. The visual cues agree with the numbers and suggest that the VeriFake trained models often failed to capture the universal features which were necessary for generalization. The t-SNE visuals evoked a deeper reconsideration of our data and the architecture of our models. As the models were unable to align with DFDC's features with the training distributions, then even with more additions, training on VeriFake seemed insufficient. This pushed us to rethink our core foundation entirely and choose an alternate architecture. Our final model was designed specifically to address this issue. Trained entirely on FF++, a widely used and higher-quality benchmark dataset, this model incorporated multiple architectural innovations: a multi-branch classification system (MLP and cosine similarity), disentangled feature learning, a reconstruction decoder for content preservation, and a GRL-powered domain-adversarial module for domain-invariant representation learning.
Interestingly, this model achieved the highest AUC on DFDC (0.7509), outperforming all previous attempts. This is a significant finding, especially considering how poorly other models handled DFDC. Its performance on Celeb-DF (0.8183) was also a massive improvement, considering the fact Celeb-DF was not present in the training data at all. However, what stood out was the drop in AUC on VeriFake (0.7054), the dataset that earlier models were originally trained on. This suggests that while FF++ helped in learning more robust features, it also introduced a domain mismatch with our custom data. The implication is important: training on FF++ allows models to better adapt to real-world test sets like DFDC, which are closer in visual fidelity and manipulation style. In contrast,

Trained on	GRL (λ)	Train Acc	Validation Acc	Test Acc	Test AUC	Predicted on	Real Acc	Fake Acc	Overall Acc	AUC
FF++ (7000 samples)	0.6	0.9506	0.9398	0.9310	0.9568	Celeb-DF(v1) (814 samples)	0.2482	0.9754	0.6118	0.7990
						DFDC (200 samples)	0.7172	0.5960	0.6566	0.7509
						VeriFake (1000 samples)	0.7395	0.5329	0.6360	0.7054
						Celeb-DF(v1) (1203 samples)	0.2482	0.9673	0.7238	0.7815
						Celeb-DF(v2) (6229 samples)	0.2334	0.9828	0.8809	0.8183

Table 5.6: Cross-Dataset evaluation results of Xception based multi-branch model

models trained solely on VeriFake (despite its less diversity in synthesis methods) may struggle with generalization due to differences in quality, lighting, compression artifacts, and synthesis artifacts.
To summarize:

- Models trained on VeriFake generalized poorly to DFDC, even with added Celeb-DF samples.
- GRL and heuristic features offered slight improvements, but not enough to achieve strong generalization.
- The final model, trained on FF++ with architectural enhancements, achieved the best cross-dataset performance, particularly on DFDC, despite showing a dip on VeriFake.
- This trade-off suggests that training on standardized, high-quality datasets like FF++ may lead to more transferable features, even if performance suffers on the original training domain.

5.3.2 Comparison with existing models

To assess the competitiveness of our proposed framework, we compared its cross-dataset generalization performance with several well-known deepfake detection models trained on FF++. The key performance metric considered here is the Area Under the ROC Curve (AUC), evaluated on unseen datasets Celeb-DF. Our final model achieved an AUC of 0.9527 on FF++, which is competitive with strong baselines such as Xception-raw (0.9970) and EfficientNet-B4 (0.9970), while significantly outperforming older approaches like HeadPose (0.4730) and Multi-task learning (0.7630). More importantly, on Celeb-DF, a dataset known for its challenging and realistic forgeries, our model attained an AUC of 0.8183, which is the highest among all models in the benchmarked list.

Methods	**Trained On**	**Tested on FF++ (AUC)**	**Tested on Celeb-DF (AUC)**
HeadPose [78]	FF++	0.4730	0.5460
D-FWA [79]	FF++	0.8100	0.5690
VA-LogReg [80]	FF++	0.7800	0.5510
Xception-c40 [81]	FF++	0.9550	0.6550
Multi-task [82]	FF++	0.7630	0.5430
Capsule [83]	FF++	0.9660	0.5750
DoubleRNN [84]	FF++	0.9318	0.7341
Two-stream [85]	FF++	0.7010	0.5380
Meso4 [86]	FF++	0.8470	0.5480
Xception-raw [87]	FF++	**0.9970**	0.4820
EfficientNet-B4 [88]	FF++	**0.9970**	0.6420
Ours	FF++	0.9568	**0.8183**

Table 5.7: Comparison of cross generalization with existing model and our model

5.4 Discussion

Our experimental results demonstrate a consistent trend: deepfake detection models that perform exceptionally well on their training datasets tend to experience notable performance degradation when evaluated on data from unseen sources. For instance, the baseline encoder paired with an MLP classification head achieved strong in-domain performance (AUC = 0.9653), yet its cross-domain AUC dropped sharply to 0.6326. In contrast, the full architecture, which integrates a decoder, disentanglement modules, and a Gradient Reversal Layer (GRL), delivered the highest cross-domain performance (AUC = 0.8183) while incurring only a modest reduction in in-domain accuracy.

This outcome aligns with the intended purpose of the GRL and disentanglement components, which aim to mitigate over-reliance on dataset-specific artefacts by promoting the extraction of domain-invariant features. The trade-off is predictable: a slight decline in accuracy on the training domain is offset by substantially improved generalization to

unfamiliar data. The inclusion of the decoder further contributes by enforcing reconstruction fidelity, thereby preserving subtle but discriminative cues that might otherwise be discarded by the encoder.

Feature-space analysis using t-SNE visualization corroborated these findings, revealing that representations frequently cluster according to dataset origin rather than real-versus-fake labels. This underscores the persistence of domain-specific encoding and is consistent with the modest yet tangible improvements observed with domain-adversarial training. While such techniques help reduce inter-domain separation in the feature space, they cannot fully compensate for performance losses under novel manipulation methods or extreme distribution shifts.

The introduction of the VeriFake dataset provided an opportunity to assess model robustness against high-fidelity, contemporary manipulations generated by tools such as Roop and FaceFusion. Although VeriFake offered valuable realism, its limited size and diversity constrained its ability to drive broad generalization, reinforcing the need for large-scale and heterogeneous datasets in this domain.

The final multi-branch Xception-based framework proved particularly effective by combining complementary strategies: dual classification heads (MLP and cosine similarity) to capture orthogonal decision cues, disentanglement losses to separate task-relevant and domain-specific representations, a decoder for regularization, and a GRL for adversarial domain alignment. This integrated design produced the strongest cross-domain performance among all evaluated models without imposing a substantial penalty on in-domain accuracy.

Despite these improvements, the best-performing models still exhibited vulnerabilities when confronted with deepfakes produced using entirely new generation techniques, atypical compression workflows, or extreme lighting conditions. Incorporating a small number of such samples into the training set yielded only marginal gains. These observations highlight a fundamental challenge in the field: achieving true robustness will require not only algorithmic improvements but also access to considerably broader and more diverse training corpora.

Chapter 6

Conclusion

Our study highlights a consistent challenge in deepfake detection: models that excel within their training domains often struggle to generalize across unseen data. Architectural innovations such as disentanglement modules, adversarial training, and reconstruction help reduce this gap, but domain bias remains evident. The introduction of the VeriFake dataset added valuable realism yet also revealed the constraints of limited scale and diversity. Together, these findings emphasize the difficulty of achieving true robustness, which frames the discussion of this work's limitations and concluding insights.
While this work explores multiple architectures, domain adaptation techniques, and training strategies for deepfake detection, several limitations remain:

1. **Dataset Bias and Limited Diversity:** The majority of earlier experiments relied heavily on VeriFake, a custom dataset with limited synthesis diversity compared to larger public benchmarks. Even though it contained a range of manipulations, its distribution did not fully capture the visual and artifact patterns present in real-world datasets such as DFDC, FF++ or Celeb-DF. This mismatch contributed to poor generalization performance when trained on it.

2. **Limited Number of Benchmarks:** Cross-dataset evaluation was conducted mainly on DFDC and Celeb-DF. Other available datasets, such as DFD or DeeperForensics, were not tested, leaving gaps in understanding the model's broader generalization capacity.

3. **Computational Constraints:** All experiments were conducted on a single GPU, which limited batch sizes and model exploration. Larger-scale experiments with more extensive hyperparameter tuning or ensembling could potentially yield further gains.

4. **Feature-Level Interpretability:** Although feature visualizations (e.g., t-SNE) were used to diagnose domain separation, interpretability remains limited. A more in-depth artifact analysis could help pinpoint specific cues the models rely on, which in turn could guide data augmentation and model design.

Our work addresses one of the most pressing issues in deepfake detection, the lack of generalization across datasets. We began by introducing VeriFake, a custom dataset of 500 real and 500 fake videos created using Roop and FaceFusion, designed to reflect the quality and style of modern manipulations. Through extensive experiments with CNN and Transformer-based architectures, both with and without domain adaptation techniques such as Gradient Reversal Layers (GRL) and heuristic features, we found that models trained primarily on VeriFake excelled on in-domain tasks but suffered severe performance

degradation on unseen datasets like DFDC and Celeb-DF. Feature visualization analysis revealed that these failures were largely due to overreliance on dataset-specific artifacts and insufficient exposure to varied manipulation types. To address this, we designed a multi-branch Xception-based framework incorporating disentangled feature learning, a reconstruction decoder to preserve content fidelity, and a GRL-powered domain-adversarial module to promote domain-invariant representations. Trained entirely on FF++, the proposed model achieved 0.7509 AUC on DFDC and 0.8183 AUC on Celeb-DF, outperforming comparable methods in the literature and demonstrating significant improvements in cross-dataset robustness. However, this gain came at the cost of reduced performance on VeriFake (0.7054 AUC), highlighting a trade-off between optimizing for standardized benchmarks and retaining effectiveness on custom or niche datasets. These findings reinforce the importance of training data diversity and domain-invariant feature extraction for real-world deployment. Moving forward, further research should explore combining high-quality benchmark datasets with diverse, real-world manipulations, incorporating continual learning to adapt to emerging forgery techniques, and developing architectures that balance benchmark optimization with adaptability to unseen domains. Therefore, future work should focus on both diverse, representative datasets and architectures that explicitly minimize domain bias, while still being sensitive to the subtle manipulation cues that even advanced fakes can't hide.

Bibliography

[1] D. Wodajo and S. Atnafu, "Deepfake video detection using convolutional vision transformer," *arXiv preprint arXiv:2102.11126*, 2021.

[2] A. Raza, K. Munir, and M. Almutairi, "A novel deep learning approach for deepfake image detection," *Applied Sciences*, vol. 12, no. 19, p. 9820, 2022.

[3] S. Suratkar and F. Kazi, "Deep fake video detection using transfer learning approach," *Arabian Journal for Science and Engineering*, vol. 48, no. 8, pp. 9727–9737, 2023.

[4] L. Verdoliva, "Media forensics and deepfakes: An overview," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 5, pp. 910–932, Aug. 2020, ISSN: 1941-0484. DOI: 10.1109/jstsp.2020.3002101 [Online]. Available: http://dx.doi.org/10.1109/JSTSP.2020.3002101

[5] Y. Li et al., "Multi-granularity tracking with modularlized components for unsupervised vehicles anomaly detection," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 2501–2510. DOI: 10.1109/CVPRW50498.2020.00301

[6] Y. Mirsky and W. Lee, "The creation and detection of deepfakes: A survey," *ACM Computing Surveys*, vol. 54, no. 1, pp. 1–41, Jan. 2021, ISSN: 1557-7341. DOI: 10.1145/3425780 [Online]. Available: http://dx.doi.org/10.1145/3425780

[7] J. Sharma, S. Sharma, V. Kumar, H. S. Hussein, and H. Alshazly, "Deepfakes classification of faces using convolutional neural networks.," *Traitement du Signal*, vol. 39, no. 3, 2022.

[8] T. Liu and H.-Z. Lu, "Analytic solution to pseudo-landau levels in strongly bent graphene nanoribbons," *Phys. Rev. Res.*, vol. 4, p. 023 137, 2 May 2022. DOI: 10.1103/PhysRevResearch.4.023137 [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevResearch.4.023137

[9] A. Ismail, M. Elpeltagy, M. S. Zaki, and K. Eldahshan, "A new deep learning-based methodology for video deepfake detection using xgboost," *Sensors*, vol. 21, no. 16, p. 5413, 2021.

[10] I. Korshunova, W. Shi, J. Dambre, and L. Theis, "Fast face-swap using convolutional neural networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 3677–3685.

[11] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2387–2395. DOI: 10.1109/CVPR.2016.262

[12] J. Lorenzo-Trueba et al., "The voice conversion challenge 2018: Promoting development of parallel and nonparallel methods," *arXiv preprint arXiv:1804.04262*, 2018.

[13] T. Wang and K. P. Chow, "Noise based deepfake detection via multi-head relative-interaction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 2023, pp. 14 548–14 556.

[14] G. H. Ishrak, Z. Mahmud, M. Farabe, T. K. Tinni, T. Reza, and M. Z. Parvez, "Explainable deepfake video detection using convolutional neural network and capsulenet," *arXiv preprint arXiv:2404.12841*, 2024.

[15] L. Chen, Y. Zhang, Y. Song, J. Wang, and L. Liu, "Ost: Improving generalization of deepfake detection via one-shot test-time training," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 24 597–24 610. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/9bf0810a4a1597a36d27ceea58667d92-Paper-Conference.pdf

[16] R. Ramanaharan, D. B. Guruge, and J. I. Agbinya, "Deepfake video detection: Insights into model generalisation — a systematic review," *Data and Information Management*, p. 100 099, 2025, ISSN: 2543-9251. DOI: https://doi.org/10.1016/j.dim.2025.100099 [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2543925125000075

[17] Y. Li, D. Zhu, X. Cui, and S. Lyu, *Celeb-df++: A large-scale challenging video deepfake benchmark for generalizable forensics*, 2025. arXiv: 2507.18015 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2507.18015

[18] T. Fernando, C. Fookes, S. Sridharan, and S. Denman, *Cross-branch orthogonality for improved generalization in face deepfake detection*, 2025. arXiv: 2505.04888 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2505.04888

[19] M. Guo, Q. Yin, W. Lu, and X. Luo, *Towards open-world generalized deepfake detection: General feature extraction via unsupervised domain adaptation*, 2025. arXiv: 2505.12339 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2505.12339

[20] A. V. Nadimpalli and A. Rattani, *On improving cross-dataset generalization of deepfake detectors*, 2022. arXiv: 2204.04285 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2204.04285

[21] L. Chen, Y. Zhang, Y. Song, J. Wang, and L. Liu, "Ost: Improving generalization of deepfake detection via one-shot test-time training," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 24 597–24 610. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/9bf0810a4a1597a36d27ceea58667d92-Paper-Conference.pdf

[22] J. Hu, S. Wang, and X. Li, "Improving the generalization ability of deepfake detection via disentangled representation learning," in *2021 IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 3577–3581. DOI: 10.1109/ICIP42928.2021.9506730

[23] G. Gupta, K. Raja, M. Gupta, T. Jan, S. T. Whiteside, and M. Prasad, "A comprehensive review of deepfake detection using advanced machine learning and fusion methods," *Electronics*, vol. 13, no. 1, 2024, ISSN: 2079-9292. DOI: 10.3390/electronics13010095 [Online]. Available: https://www.mdpi.com/2079-9292/13/1/95

[24] H. Wahab, H. Ugail, and L. Jaleel, *Ensemble-based deepfake detection using state-of-the-art models with robust cross-dataset generalisation*, 2025. arXiv: 2507.05996 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2507.05996

[25] H. Huang, N. Sun, X. Lin, and N. Moustafa, "Towards generalized deepfake detection with continual learning on limited new data," in *2022 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, 2022, pp. 1–7. DOI: 10.1109/DICTA56598.2022.10034569

[26] S. A. Khan, A. Artusi, and H. Dai, "Adversarially robust deepfake media detection using fused convolutional neural network predictions," *arXiv preprint arXiv:2102.05950*, 2021.

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

[28] L. Guarnera, O. Giudice, and S. Battiato, "Deepfake detection by analyzing convolutional traces," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, Jun. 2020.

[29] R. Rafique, R. Gantassi, R. Amin, J. Frnda, A. Mustapha, and A. H. Alshehri, "Deep fake detection and classification using error-level analysis and deep learning," *Scientific Reports*, vol. 13, no. 1, p. 7422, 2023.

[30] O. A. H. H. Al-Dulaimi and S. Kurnaz, "A hybrid cnn-lstm approach for precision deepfake image detection based on transfer learning," *Electronics*, vol. 13, no. 9, p. 1662, 2024.

[31] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "Mesonet: A compact facial video forgery detection network," Dec. 2018, pp. 1–7. DOI: 10.1109/WIFS.2018.8630761

[32] L. Trinh, M. Tsang, S. Rambhatla, and Y. Liu, "Interpretable and trustworthy deepfake detection via dynamic prototypes," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Jan. 2021, pp. 1973–1983.

[33] S. Tariq, S. Lee, and S. S. Woo, "A convolutional lstm based residual network for deepfake video detection," *arXiv preprint arXiv:2009.07480*, 2020.

[34] D. S. AbdElminaam, N. Sherif, Z. Ayman, M. Mohamed, and M. Hazem, "Deepfakedg: A deep learning approach for deep fake detection and generation," *Journal of Computing and Communication*, vol. 2, no. 2, pp. 31–37, 2023.

[35] L. Pryor, R. Dave, M. Vanamala, et al., "Deepfake detection analyzing hybrid dataset utilizing cnn and svm," *arXiv preprint arXiv:2302.10280*, 2023.

[36] S. S. Khalil, S. M. Youssef, and S. N. Saleh, "Icaps-dfake: An integrated capsule-based model for deepfake image and video detection," *Future Internet*, vol. 13, no. 4, p. 93, 2021.

[37] D. Gong, Y. J. Kumar, O. S. Goh, Z. Ye, and W. Chi, "Deepfakenet, an efficient deepfake detection method," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 6, pp. 201–207, 2021.

[38] M. Nawaz, A. Javed, and A. Irtaza, "Resnet-swish-dense54: A deep learning approach for deepfakes detection," *The Visual Computer*, vol. 39, no. 12, pp. 6323–6344, 2023.

[39] A. Kohli and A. Gupta, "Detecting deepfake, faceswap and face2face facial forgeries using frequency cnn," *Multimedia Tools and Applications*, vol. 80, no. 12, pp. 18 461–18 478, 2021.

[40] Z. Xia, T. Qiao, M. Xu, X. Wu, L. Han, and Y. Chen, "Deepfake video detection based on mesonet with preprocessing module," *Symmetry*, vol. 14, no. 5, p. 939, 2022.

[41] A. Ismail, M. Elpeltagy, M. Zaki, and K. A. ElDahshan, "Deepfake video detection: Yolo-face convolution recurrent approach," *PeerJ Computer Science*, vol. 7, e730, 2021.

[42] W. H. Abir et al., "Detecting deepfake images using deep learning techniques and explainable ai methods," *Intelligent Automation & Soft Computing*, vol. 35, no. 2, pp. 2151–2169, 2023.

[43] P. M. G. I. Reis and R. O. Ribeiro, "A forensic evaluation method for deepfake detection using dcnn-based facial similarity scores," *Forensic Science International*, vol. 358, p. 111 747, 2024.

[44] A. Kaur, A. Hoshyar, V. Saikrishna, S. Firmin, and F. Xia, "Deepfake video detection: Challenges and opportunities," *Artificial Intelligence Review*, vol. 57, May 2024. DOI: 10.1007/s10462-024-10810-6

[45] S. Lyu, "Deepfake detection: Current challenges and next steps," *2020 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pp. 1–6, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:214605906

[46] S. Chhabra, K. Thakral, S. Mittal, M. Vatsa, and R. Singh, "Low-quality deepfake detection via unseen artifacts," *IEEE Transactions on Artificial Intelligence*, vol. 5, no. 4, pp. 1573–1585, 2024. DOI: 10.1109/TAI.2023.3299894

[47] A. Rahman et al., "Short and low resolution deepfake video detection using cnn," in *2022 IEEE 10th Region 10 Humanitarian Technology Conference (R10-HTC)*, 2022, pp. 259–264. DOI: 10.1109/R10-HTC54060.2022.9929719

[48] H. Lee et al., *The tug-of-war between deepfake generation and detection*, 2024. arXiv: 2407.06174 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2407.06174

[49] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, *Faceforensics++: Learning to detect manipulated facial images*, 2019. arXiv: 1901.08971 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1901.08971

[50] Y. Li and S. Lyu, *Exposing deepfake videos by detecting face warping artifacts*, 2019. arXiv: 1811.00656 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1811.00656

[51] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, *Celeb-df: A large-scale challenging dataset for deepfake forensics*, 2020. arXiv: 1909.12962 [cs.CR]. [Online]. Available: https://arxiv.org/abs/1909.12962

[52] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "Mesonet: A compact facial video forgery detection network," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, IEEE, Dec. 2018, pp. 1–7. DOI: 10.1109/wifs.2018.8630761 [Online]. Available: http://dx.doi.org/10.1109/WIFS.2018.8630761

[53] B. Dolhansky et al., *The deepfake detection challenge (dfdc) dataset*, 2020. arXiv: 2006.07397 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2006.07397

[54] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, *Deepfakes and beyond: A survey of face manipulation and fake detection*, 2020. arXiv: 2001.00179 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2001.00179

[55] L. Y. Gong and X. J. Li, "A contemporary survey on deepfake detection: Datasets, algorithms, and challenges," *Electronics*, vol. 13, no. 3, 2024, ISSN: 2079-9292. DOI: 10.3390/electronics13030585 [Online]. Available: https://www.mdpi.com/2079-9292/13/3/585

[56] S0md3v, *Roop: One-click face swap*, https://github.com/s0md3v/roop, GitHub repository, 2023.

[57] F. Contributors, *Facefusion: Industry leading face manipulation platform*, https://github.com/facefusion/facefusion, GitHub repository, 2024.

[58] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "Faceforensics: A large-scale video dataset for forgery detection in human faces," *arXiv preprint arXiv:1803.09179*, 2018.

[59] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "Faceforensics++: Learning to detect manipulated facial images," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1–11.

[60] B. Dolhansky et al., "The deepfake detection challenge (dfdc) dataset," *arXiv preprint arXiv:2006.07397*, 2020.

[61] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-df: A large-scale challenging dataset for deepfake forensics," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 3207–3216.

[62] P. Korshunov and S. Marcel, *Deepfakes: A new threat to face recognition? assessment and detection*, 2018. arXiv: 1812.08685 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1812.08685

[63] S. Barrington, M. Bohacek, and H. Farid, "Deepspeak dataset v1. 0," *arXiv preprint arXiv:2408.05366*, 2024.

[64] G. Bertasius, H. Wang, and L. Torresani, "Is space-time attention all you need for video understanding?," 2021. arXiv: 2102.05095 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2102.05095

[65] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," 2015. arXiv: 1409.7495 [stat.ML]. [Online]. Available: https://arxiv.org/abs/1409.7495

[66] Y. Li, M.-C. Chang, and S. Lyu, "In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking," 2018. arXiv: 1806.02877 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1806.02877

[67] A. Albazony, H. Al-Wzwazy, A. S. Al-Khaleefa, M. Alazzawi, M. Almohamadi, and S. Alavi, "Deepfake videos detection by using recurrent neural network (rnn)," pp. 103–107, Jul. 2023. DOI: 10.1109/AICCIT57614.2023.10217956

[68] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. arXiv: 1512.03385 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1512.03385

[69] F. Chollet, *Xception: Deep learning with depthwise separable convolutions*, 2017. arXiv: 1610.02357 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1610.02357

[70] Y. Ganin et al., *Domain-adversarial training of neural networks*, 2016. arXiv: 1505.07818 [stat.ML]. [Online]. Available: https://arxiv.org/abs/1505.07818

[71] A. Hinke-Navarro, M. Nieto-Hidalgo, J. M. Espin, and J. E. Tapia, *Enhanced deep learning deepfake detection integrating handcrafted features*, 2025. arXiv: 2507.20608 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2507.20608

[72] Y. Ganin et al., *Domain-adversarial training of neural networks*, 2016. arXiv: 1505.07818 [stat.ML]. [Online]. Available: https://arxiv.org/abs/1505.07818

[73] T. Fawcett and F. Provost, "Adaptive fraud detection," *Data Mining and Knowledge Discovery*, vol. 1, no. 3, pp. 291–316, 1997. DOI: 10.1023/A:1009700419189

[74] P. Flach, J. Hernández-Orallo, and C. Ferri, "A coherent interpretation of auc as a measure of aggregated classification performance," ser. ICML'11, Bellevue, Washington, USA: Omnipress, 2011, pp. 657–664, ISBN: 9781450306195.

[75] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006, ROC Analysis in Pattern Recognition, ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2005.10.010 [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016786550500303X

[76] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997. DOI: 10.1016/S0031-3203(96)00142-2

[77] C. X. Ling, J. Huang, and H. Zhang, "Auc: A statistically consistent and more discriminating measure than accuracy," in *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, ser. IJCAI'03, Acapulco, Mexico: Morgan Kaufmann Publishers Inc., 2003, pp. 519–524.

[78] X. Yang, Y. Li, and S. Lyu, *Exposing deep fakes using inconsistent head poses*, 2018. arXiv: 1811.00661 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1811.00661

[79] Y. Li and S. Lyu, *Exposing deepfake videos by detecting face warping artifacts*, 2019. arXiv: 1811.00656 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1811.00656

[80] F. Matern, C. Riess, and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," in *2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*, 2019, pp. 83–92. DOI: 10.1109/WACVW.2019.00020

[81] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, "Faceforensics++: Learning to detect manipulated facial images," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 1–11. DOI: 10.1109/ICCV.2019.00009

[82] H. H. Nguyen, F. Fang, J. Yamagishi, and I. Echizen, *Multi-task learning for detecting and segmenting manipulated facial images and videos*, 2019. arXiv: 1906.06876 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1906.06876

[83] H. H. Nguyen, J. Yamagishi, and I. Echizen, *Capsule-forensics: Using capsule networks to detect forged images and videos*, 2018. arXiv: 1810.11215 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1810.11215

[84] I. Masi, A. Killekar, R. M. Mascarenhas, S. P. Gurudatt, and W. AbdAlmageed, *Two-branch recurrent network for isolating deepfakes in videos*, 2020. arXiv: 2008.03412 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2008.03412

[85] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, *Two-stream neural networks for tampered face detection*, 2018. arXiv: 1803.11276 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1803.11276

[86] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "Mesonet: A compact facial video forgery detection network," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2018, pp. 1–7. DOI: 10.1109/WIFS.2018.8630761

[87] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, "Faceforensics++: Learning to detect manipulated facial images," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 1–11. DOI: 10.1109/ICCV.2019.00009

[88] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, Sep. 2019, pp. 6105–6114. [Online]. Available: https://proceedings.mlr.press/v97/tan19a.html