Assessing Deepfake Detection Models: A Comparative Study for Misinformation Detection and Prevention

University of Turku
Department of Computing
Master's Thesis
Information & Communication Technology: Cyber Security
June 2025
Chathura Liyana Gamage

Supervisors:
Jouni Isoaho (University of Turku)
Tahir Mohammad (University of Turku)

UNIVERSITY OF TURKU
Department of Computing

Chathura Liyana Gamage: Assessing Deepfake Detection Models: A Comparative Study for Misinformation Detection and Prevention

Master's Thesis, 83 p., 12 app. p.
Information & Communication Technology: Cyber Security
June 2025

Deepfake videos are proliferating faster than existing detection tools can keep pace, yet most prior studies benchmark detectors only on single datasets (with few exceptions that have generalized models), ignore bitrate degradation, and omit resource costs, leaving practitioners uncertain about real-world reliability. Addressing this gap, this study evaluates the reliability and deployability of selected deepfake detectors and formulates actionable countermeasures against synthetic media misinformation. Three state-of-the-art models, BA-TFD+, Convolutional Cross Efficient ViT, and CLRNet, were analysed on five publicly available datasets: DeeperForensics-1.0, Celeb-DF (v2), LAV-DF, DFD, and DFW under three H.264 compression settings. The extensive inference with measurement of precision, recall, AUC, F1-score, latency, and computational memory usage highlighted calibration gaps visualized through heat maps of AUC and F1 and AUC versus F1 scatter plots. Results show that at least one model achieved F1 approximately 0.80 on every dataset and compression levels and all detectors remain insensitive to bitrate reduction (i.e., higher compressed data), performance is fragmented as Cross ViT generalizes best but have a significant memory usage of 3-5 GB, BA-TFD+ offers near perfect accuracy on familiar dataset (i.e., LAV-DF) yet suffers severe threshold bias elsewhere, while CLRNet is lightweight and fast but not so well performed in terms of detection accuracy everywhere. This study concludes that no single detector rises as a winner in isolation. A combined or layered approach is recommended, such as edge-level pre-filters and cloud-side ensemble, continuous threshold calibration, and cryptographic watermarking techniques. Policy proposals include, but are not limited to, mandatory C2PA signatures, obligatory AI-labeling, and common standard compliance availability worldwide. Together, these technical and policy-wise measures provide a solid roadmap for sustaining trust in visual evidence as generative media evolves.

Keywords: deepfake detection, domain generalization, misinformation mitigation, threshold calibration, cryptographic watermarking

Disclaimer

Artificial Intelligence (AI) tools, including **ChatGPT** (developed by OpenAI) and **Blackbox AI** open source and free versions, were utilized during the development of this thesis project to assist in the generation, structuring, and troubleshooting of Python scripts for dataset preprocessing, frame extraction, and model evaluation experiments. Moreover, graphical tools including **Canva** and **Flourish** were used to generate graphs and plots to visualize the collected results, which supported the deep analysis. Furthermore, the **Grammarly Pro** version was used for proofreading.

In addition, the author carefully reviewed, adapted, and validated all AI-generated outputs. The author is fully responsible for the content, analysis, and conclusions presented in this work.

Contents

List of Figures

List of Tables

List of Abbreviations

3D	Three Dimensional
ACC	Accuracy
AGT	Activity Graph Transformers
AIT	Average Inference Time
API	Application Programming Interface
AP	Average Precision
AR	Average Recall
AV	Audio-Visual
AI	Artificial Intelligence
AFP	Agence France-Presse
AUC	Area under the Receiver Operating Characteristic Curve
BA-TFD	Boundary Aware Temporal Forgery Detection
BA-TFD+	Boundary Aware Temporal Forgery Detection Plus
BMN	Boundary Matching Network
BNN	Binary Neural Network
C2PA	Coalition for Content Provenance and Authenticity
CGI	Computer-Generated Imagery
CLRNet	Convolutional LSTM-based Residual Network
CMS	Content Management System
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DDS	Deepfake Detection System
DEG	Dataset-Embedding Generator

DF	Deepfake
DFD	Deepfake Detection
DFDC	Deepfake Detection Challenge
DFF	Deepfake Face
DFT	Discrete Fourier Transform
DFW	Deepfake-in-the-Wild
DWT	Discrete Wavelet Transform
EER	Equal Error Rate
EU	European Union
F1	F1 Score
F2F	Face2Face
FF++	FaceForensics++
FFD	Facial Feature Drift
FFmpeg	Fast Forward Moving Picture Experts Group
FN	False Negative
FP	False Positive
FPR	False Positive Rate
FS	FaceSwap
GAN	Generative Adversarial Network
GM-DF	Generalized Multi-Scenario Deepfake Detection Framework
GPU	Graphics Processing Unit
ICP	Internet Content Providers
LAV-DF	Localized Audio Visual Deepfake
LBP	Local Binary Pattern
LRCN	Long-term Recurrent Convolutional Networks
LSTM	Long Short-Term Memory
LTS	Long-Term Support

MDP	Multi-Dataset Representation
MDEO	Meta-Domain-Embedding Optimizer
MDS	Modality Dissonance Score
Me	Memory Usage
ML	Machine Learning
MP	Member of Parliament
MTCNN	Multi-Task Cascaded Convolutional Network
NLP	Natural Language Processing
NT	NeuralTextures
NeRFs	Neural Radiance Fields
PM	Prime Minister
PO	Civic Platform
PPG	photoplethysmography
Pr	Precision
RAM	Random Access Memeory
Re	Recall
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristics
SVM	Support Vector Machine
SMER	Direction – Social Democracy
StA	Spatiotemporal Adapter
SV2TTS	Speaker Verification to Multispeaker Text-To-Speech Synthesis
TN	True Negative
TP	True Positive
TPR	True Positive Rate
TTI	Text-to-Image
UK	United Kingdom

US	United States
VAE	Variational Autoencoder
VFX	Visual Effects
ViT	Vision Transformer
VOA	Voice of America
WDF	Wild Deepfake

1 Introduction

In a continuously digitizing world, tackling the authenticity of digital media, especially those that circulate on social media platforms and worldwide news forums, has become crucial to maintaining public trust, personal security, and political stability. Deepfake technologies, enabled by the advancement in Artificial Intelligence (AI), together with deep learning, have created the ability to generate hyper-realistic digital media, including synthetic video, audio, and images that are difficult to distinguish from real ones. Although the same technologies may offer creative and commercially valuable opportunities for individuals and society, they also pose a significant risk when exploited for malicious purposes such as misinformation/disinformation campaigns, identity theft, social and political manipulation, etc., which can negatively affect both individuals in society and commercially.

Moreover, synthetic media generation methods have increased in recent years and are becoming increasingly sophisticated, mainly due to the advancement and usage of technologies such as Generative Adversarial Networks (GANs) and other relevant deep learning methods. However, defensive methods might not advance at the same rate as generation methods. Therefore, it is a crucial area to explore further to introduce sophisticated deepfake detection methods. Furthermore, the growing sophistication of synthetic media challenges the existing deepfake detection methods, particularly when the media is shared on numerous platforms, such as social media platforms and other news forums, that alter the content through resolution changes

and compression. Therefore, it is vital that deepfake detection models have the ability not only to perform well in clean datasets with high-quality media but also to be generalized effectively in real-world scenarios where the media quality is often degraded and compression is applied to the media. Despite the various state-of-the-art models claiming high accuracy and better performance in deepfake detection under controlled environments, there remains a gap in understanding how these models behave across diverse datasets and compression scenarios.

This study aims to fill this gap by conducting a comparative evaluation of three pre-trained deepfake detection models (BA-TFD+, Cross Efficient ViT, and CLR-Net). Each model will test with five selected deepfake datasets (Celeb-DF (V2), DeeperForensics-1.0, Deepfake Detection (DFD), Localized Audio Visual Deepfake (LAV-DF), and Deepfake in the Wild (DFW)) at three compression levels (C0; no compression (RAW), C23; medium compression and C40; high compression). The study evaluates models' accuracy and performance in terms of precision, recall, F1-Score, and Area Under the Curve (AUC), generalization ability across datasets and compression levels, and computational efficiency in terms of average inference time and CPU/GPU memory usage.

Through this comprehensive analysis, this study seeks to provide useful insights into the robustness, limitations, and readiness for deployment in the real-world environments of the selected deepfake detection methods. Furthermore, based on the findings, the study proposes recommendations to deploy more robust and reliable detection and actionable strategies and policies to mitigate deepfake-driven misinformation campaigns.

1.1 Background

The term "deepfake" originated from a Reddit user who used deep learning methods to generate synthetic videos of celebrities by swapping and replacing their faces

in different video clips for entertainment purposes [1]. Since then, the field has advanced and has explored to a greater extent with deep generative models such as GANs, Autoencoders, vision transformer, and Diffusion models etc. Then, it came to a level that even a highly trained human eye could not distinguish the real from the fake.

In addition, at the same time, the ease of accessibility and usage of such deepfake generation tools became popular not only among researchers but also among the general public in the form of open-source, free-to-use, and user-friendly interfaces and tools. This trend has led to use the deepfake not only in entertainment purposes but also for malicious purposes such as weaponized in political misinformation campaigns targeting world leaders and critical national elections of countries, manipulating the social order, personal harassment, corporate sabotage, and numerous other forms of malicious intentions [2], [3].

The increasing popularity of deepfake generation tools highlighted the need for sophisticated and reliable deepfake detection technologies to counter the rising threat. Responding to this, the research community began to study and propose various methods of deepfake detection methods. Although the early techniques used basic handcrafted feature classifiers, with the evolving nature of generation methods, the detection methods also became advanced and sophisticated, embracing deep neural networks, which can learn and detect patterns automatically. Those techniques range from machine learning and blockchain-based methods to even the novel Artificial Intelligence (AI) powered tools.

Despite these advances, generalization remains a persistent challenge in deepfake detection. Many state-of-the-art deepfake detection models performed well on the subsets of datasets they were trained and tested on, but, in most cases, failed on previously unseen datasets, manipulation methods, or media distortions such as media quality degradation, compression, resizing, or re-encoding. This gap between con-

trolled laboratory experiments and real-world robustness raises concerns about the practical deployability of deepfake detection methods in live working environments.

Additionally, computational efficiency is a growing concern. As the deepfake detection models become more complex and deep in calculations and data processing, they become increasingly resource-intensive, which limits their usability on resource-constrained devices such as mobile devices.

Thus, it is essential to evaluate existing deepfake detection models not only for their detection performance and accuracy but also for their generalization ability and efficiency under varied realistic conditions. This study addresses this critical need by conducting a comparative analysis of three selected pre-trained deepfake detection models across multiple datasets and compression levels.

1.2 Research Questions and Objectives

The primary target of this research is to evaluate existing generalized and non-generalized deepfake detection models for their suitability and strength in combating deepfake-based misinformation campaigns and derive actionable insights for deployment.

Completing this study will answer the following Research Questions and achieve the subsequent Research Objectives (RO).

RQ1: How effective are existing pre-trained deepfake detection models in real-world misinformation scenarios?

RQ2: How can simulation-based evaluations reveal these models' strengths and limitations across varied conditions (e.g., video quality, compression levels)?

RQ3: What actionable deployment strategies and policy recommendations can be derived from model performance insights to enhance detection and curb deepfake-based misinformation campaigns?

RO1: Evaluate the effectiveness of selected pre-trained deepfake detection mod-

els under real-world misinformation scenarios. **(maps to RQ1)**

RO2: Perform simulation-based evaluations to identify model strengths and limitations across varied conditions (video quality, compression levels). **(maps to RQ2)**

RO3: Derive actionable deployment strategies and formulate policy recommendations based on evaluation insights to mitigate the spread of deepfake-driven misinformation campaigns. **(maps to RQ3)**

1.3 Scope of Research

This study is scoped explicitly to a comparative evaluation of three pre-trained deepfake detection models, focusing on five datasets (Celeb-DF (v2), DeeperForensics 1.0, DFD, LAV-DF, and DFW) in each containing data with three different compression levels (C0, C23, and C40). Moreover, four evaluation metrics (Precision, Recall, F1-Score, and AUC) will be used to evaluate the model's performance, while the generalization ability of each model is assessed based on how stable and consistent the models across unseen datasets and compression levels. In addition, the computational efficiency of each model will be evaluated using two metrics (Average Inference Time and CPU/GPU memory usage).

The selected model will be evaluated under controlled experiment settings using standard evaluation metrics, and equal classes (i.e., real and fake) of subsets of selected datasets will be used to ensure the experiment is unbiased and fair. This study does not aim to propose a novel deepfake detection model architecture but to comparatively analyse existing high-performing pre-trained models to understand their deployment capabilities in real-world scenarios involving compressed data and manipulative data.

The detailed explanation on selected pre-trained deepfake detection models, selected datasets with compression levels, and selected standard evaluation metrics

will be discussed in Chapter 3.

1.4 Research Methodology

This study adopts a quantitative experimental approach to assess pre-trained deepfake detection models based on their accuracy and performance, generalization ability, and computational efficiency. The experiment setup is designed to answer the first two research questions as follows;

RQ1: How effective are existing models detecting deepfakes under real-world misinformation scenarios?

RQ2: How do different compression levels and quality degradation of video data impact the models' performance?

To achieve this, pre-trained deepfake detection models are tested on multiple datasets, including lab-created deepfakes such as DFD (Deepfake Detection) and real-world misinformation deepfakes such as DFW (Deepfake in the Wild) and Celeb-DF (v2). The models are also evaluated under different video compression levels (RAW (C0), low compression (C23), high compression (C40)) to analyze robustness in social media-based misinformation settings.

Below is a summary of the experimental setup, which consists of three main components. The detailed experimental setup will be explained in Chapter 3.

1. **Pre-trained deepfake detection model selection:** A diverse set of pre-trained deepfake detection models will be selected based on different architectures, such as CNN-based, transformer-based, etc. Then, each model is evaluated based on accuracy, robustness, generalization capabilities, and computational efficiency in different conditions.

2. **Dataset selection and preprocessing:** To test the generalization ability of each model, multiple datasets will be used. Moreover, based on the certain

model's requirements, each video will be preprocessed by extracting frames, resizing, and normalizing images before passing them into the models for classification.

3. **Real-world testing and compression simulation:** Since misinformation spreads through social media in most times by using low-quality compressed videos, the models will be tested under different conditions of the data such as RAW videos (original quality (C0), low compression videos (C23), and high compression videos (C40). Moreover, compression is applied using the "FFmpeg" free and open source package.

As per the evaluation metrics, the selected pre-trained models will be evaluated based on three key metrics;

1. **Model performance:** Precision, Recall, AUC, and F1-score will be used to measure detection effectiveness.

2. **Generalization ability:** How will the models perform regarding the previously unseen deepfake dataset? (i.e., model trained on the DFD and evaluated on the Celeb-DF (v2) dataset).

3. **Computational efficiency:** Average Inference Time (seconds per video/frame/batch) and CPU/GPU memory usage are recorded to assess real-time usability of each evaluated model.

Then, in the final section, data analysis will be done. The results will be analyzed using various graphs/charts (i.e., heat maps, accuracy plots, etc.) to identify trends. Moreover, comparative analysis is performed to evaluate:

1. Which models generalize best?

2. How much does compression affect performance?

3. Are current detection methods reliable against misinformation deepfakes in those varied conditions of input data?

1.5 Thesis Structure

This thesis is structured as follows:

Chapter 2 provides detailed introduction of deepfake technologies, how the deepfake generation and detection techniques evolved throughput past decades, what are the state-of-art deepfake detection techniques and their limitations, what is it meant by "misinformation campaigns" and the motives behind, what are the possible impacts of those, and finally a summarizing of selected deepfake based incidents and what are the potential remedies involved (if applicable).

Chapter 3 explains the basis of model selection, dataset selection, and pre-processing, followed by the metrics used in this research for model evaluation, and results of the evaluation.

Chapter 4 analyzes the results of the evaluation based on three criteria: model accuracy and performance, generalization ability, and computational efficiency and limitations. Moreover, the latter provides recommendations for the deployment of deepfake detection models and policy recommendations for combating deepfake-based misinformation campaigns, followed by a discussion of the work results.

Chapter 5 discusses limitations of this work and possible future works, followed by the conclusion of this research.

2 Literature Review

This chapter explores the theoretical aspects of deepfake technologies and related concepts, as well as the existing literature on deepfakes and deepfake detection. First, the evolution of deepfake technologies over the last decade will be discussed, followed by an examination of deepfake generation and detection mechanisms. The subsequent section of this chapter discusses misinformation campaigns in terms of impacts and relevant case studies.

2.1 Overview of Deepfake Technologies

Deepfake technologies refer to AI or machine learning-driven techniques to generate highly realistic yet synthetic media by modifying or replacing faces, voice, or even entire video sequences [4]. While deepfakes initially emerged as an innovative form of entertainment using Generative Adversarial Networks (GANs) [5], they later became a crucial cybersecurity threat, particularly in the form of misinformation campaigns. Their potential to manipulate the digital content, including but not limited to political speeches, personal identities, financial fraud, and other news reports, across the internet to gain political, economic, or social advantages raised considerable concern regarding the authenticity and trust of digital content. Now, with the further advancement of technology, it has reached a point where a random individual cannot distinguish between real and fake digital content they see on the internet.

When we consider deepfake, it has both positive and negative applications, although most of the time, the attention is diverted to the negative applications or disadvantages.

Based on the qualitative analysis by Navarro Martínez et al. [6], deepfakes have both health benefits and risks. From the participant's point of view, deepfakes are beneficial in numerous ways, such as economics, commerce, and the entertainment industry [6]. However, they have also identified certain disadvantages of deepfakes, including problems related to information hoaxes (i.e., identity theft, impersonation), cyberbullying, and legal dangers [6] as well as misinformation and propaganda, privacy violations, and ethical concerns [7]. For instance, a deepfake video of Russian VOA (Voice of America) journalist Ksenia Turkova was used to promote a cryptocurrency product in 2023, which appeared authentic; even she herself had difficulty differentiating whether it was real or fake [8]. Political propaganda and misinformation campaigns are a critical concern that will be discussed in more detail with the case studies in section 2.2.2.

On the other hand, the same technology can be used to benefit the healthcare industry, such as early identification of diseases (e.g., cancer) or detection of possible tumors [6]. In addition, deepfake have benefits including but not limited to educational applications, improved visual effects (i.e, VFX in film industry), enhanced simulations (i.e., aviation and military training), and increased accessibility (i.e., subtitles or translations for audio/video content which beneficial for people with hearing impaired or speak different languages), etc. [7][9]. Moreover, journalists and human rights activists can remain anonymous by using syntactic media (i.e., masked audio/video) to counter corrupt government regimes worldwide [9].

The following subsections explore the evolution of deepfake technologies over the past decades, their underlying generation methods, and numerous detection techniques used to counter deepfakes.

2.1.1 Evolution of Deepfake Technologies

This chapter explores how deepfake technologies have evolved during the past decades.

2.1.1.1 Early Foundations (1943-2000s)

The roots of modern deep learning and deep/artificial learning techniques date back to 1943, even before the commercialization of computers. McCulloch and Pitts tried to understand how small artificial neuron networks can mimic brain-like activities [10]. Then, later in 1958, Frank Rosenblatt introduced the "Perceptron", a two-layer learning algorithm that allows two inputs in the input layer and produces one output in the output layer [11]. This network can learn the association between a simple input image and output, in such a way that it determines whether a selected object is present or not in the given input image [10]. Then, after a proof in 1969 on the incapability of Perceptron to learn associations unless the inputs are in a simpler form, funding for the Artificial Intelligence (AI) research was cut down, and started the so-called "The first winter of AI" [10], [12].

In addition, the use of Computer-Generated Imagery (CGI) in the cinema industry in 1958 [13] can be considered a form of early advancement and precursor to modern-day so-called deepfake technology. Then, in the 1990s, researchers began experimenting on how to blend facial features between images. In 1991, Turk and Pentland introduced "Eigenfaces" [14], a method for extracting facial features from a given image and a facial recognition system that identifies an individual by comparing those facial features with an image of a known individual. Furthermore, technologies such as face-swapping, image warping, and texture mapping were introduced and used mainly in the entertainment industry [15]. However, due to high cost, limited resources, and the need for manual labour, the usage and spread of these technologies were limited.

2.1.1.2 The Rise of Machine Learning (ML) and Introduction of Generative Adversarial Networks (GANs) (2010-2016)

We can consider the most significant breakthrough in ML evolution as the introduction of deep learning and GANs. In 2014, Goodfellow et al., introduced a novel ML framework called GAN [16], which has two neural networks, a generator and a discriminator, complete against each resulting in more realistic image manipulation [16]. In 2016, Thies et al. developed and introduced Face2Face [17], a real-time face capture and reenactment method of videos that allows users to manipulate the facial expressions of individuals in existing videos [17]. Therefore, these progressing and newly developed methods demonstrated the potential of using deepfake-based applications, not only in the entertainment industry, but also in misinformation campaigns.

2.1.1.3 Introducing the term "deepfake" and Rise of Deepfakes (2017-2019)

The term "deepfake" first appeared on the Internet in 2017 when a Reddit user named "deepfakes" used deep learning to swap faces of celebrities into pornography videos [1]. Around 2019, another notorious application called "DeepNude" was released as an open source application that allows users to undress photos of women virtually, but was later taken down by the same publishers due to the inappropriate usage despite the high demand [18]. Around the same time, more open-source applications, such as DeepFaceLab [19], FakeApp [20], and FaceSwap [21], were released to enable even non-tech users to access and generate realistic deepfakes with minimal effort easily. In addition, the availability of detailed tutorials for open-source projects such as DeepFaceLab [19] makes things more accessible and more manageable for interested individuals [22].

Furthermore, due to this easy accessibility and availability, concerns about ethi-

cal usage and potential misuse grew significantly. Several incidents involving political deepfakes such as deepfake of former president of the United States (US), Barack Obama [23] implied the possibility of potential usage of deepfakes in misinformation campaigns [24].

2.1.1.4 Advancement and Introduction of Regulations (2020-Present)

From 2020 onward, deepfake generation started to reach unprecedented level of realism with the improvements of AI, AI architectures, and easy and quick access for any individual (i.e., even begun to introduce mobile applications) including but not limited to StyleGAN [25], [26], Neural Radiance Fields (NeRFs) [27], and Transformer-based models such as Vision Transformers (ViT). Moreover, techniques in highly efficient face reenactment and voice synthesis have been developed, making it incredibly difficult to distinguish between real and fake (i.e., AI-generated) content.

On the other hand, simultaneously, deepfake detection techniques have advanced to counter the rising threat of deepfakes in various areas, including technical, social, economic, and psychological. For instance, Deepfake Detection Challenge (DFDC) (both models and datasets) by Meta [28] and FaceForensics++ dataset [29], etc. as well as deepfake detection benchmarks such as DeepfakeBench [30] were made a considerable contribution in deepfake detection research area.

In terms of legal and policy, both governments and tech companies have introduced numerous measures to mitigate the spread of deepfake misinformation. Meta's AI policy [31] and the European Union (EU)'s AI Act [32] require transparency in AI-generated content.

2.1.2 Methods of Deepfake Generation

Deepfake generation uses a deep learning algorithm to make synthetic audio, video, images, or text content [33]. The core principle behind deepfake generation is to train AI models to manipulate or synthesize audio, video, image, or text to create highly realistic but fake content. With the rapidly evolving technology, deepfake generation has become much easier and accessible to anyone, even with limited knowledge in computing. This chapter provides an in-depth examination of the methods used in deepfake generation.

There is a wide range of methods available to generate deepfake content, with significant research development in the area and technological advancements, especially in the field of AI. For instance, if we consider facial forgery, deepfake generation can be divided into two main methods of learning-based facial forgery, namely, face-swapping and face-reenactment [30].

Face-swapping is an autoencoder-based manipulation in which the face of an individual in a source video is replaced by the face of the individual in the target video [22], [30]. It has three steps. First, it detects the face from the source image and selects the target face from the library that matches the facial appearance and pose of the source [22]. Then, it replaces the eyes, nose, and mouth of the face and adjusts the lighting conditions and color of the target to match the input source image's characteristics, thereby blending in [22]. Finally, it calculates the distance of the replacement regions to ensure successful overlap with the target face [22]. However, it might look unnatural as it fails to replace the facial expression of the source with that of the target [22]. On the other hand, unlike face-swapping, face-reenactment techniques are graphics-based and they imitate the expression of the source to the target [30]. Face reenactment can transfer facial gestures, eye movements, and mouth movements to the target video, which is often used in the entertainment industry, for example, for dubbing [22].

Moreover, deepfake generation can be broadly categorized into two main categories: audio and visual manipulations, depending on the target modification type [22]. Audio deepfake can be further classified into two main categories: text-to-speech synthesis and voice conversion [22]. Then, visual-based deepfakes can be further classified into five categories: face-swap, lip-syncing, face-reenactment, entire face synthesis, and facial attribute manipulation [22].

On the other hand, based on different machine learning and deep learning architectures and techniques, deepfake generation can be classified into three main categories as follows.

2.1.2.1 Generative Adversarial Networks (GANs)

GANs can be considered the most widely used and popular method of deepfake generation, due to their ability to generate hyper-realistic synthetic media. First introduced by Goodfellow et al. (2014) [16], GAN consists of two competing neural networks, called the generator and the discriminator, which engage in a continuous learning process to produce realistic synthetic media [16]. The generator is a neural network used to modify the videos as credibly as possible so that the counterpart (i.e., discriminator) can be deceived [34]. On the other hand, the discriminator acts as a detector to identify whether the input video is fake or real by comparing it against the real data [34], [35]. Due to this continuous process, over time, resulting in very realistic and credible output (i.e., deepfake) [34]. There are notable implementations of GAN over time, such as StyleGAN [26], StarGAN V2 [36], and DiscoGAN [37], etc. Moreover, GAN-based generation methods including Face2Face [17] and FaceSwap [38] are worth mentioning [39].

2.1.2.2 Autoencoders (Variational Autoencoders - VAEs)

Autoencoders, particularly Variational Autoencoders (VAEs), also play a significant role in the generation of deepfakes. It uses a pair of encoders and decoders to decompose and recompose two distinct faces, where 4 or 5 layers represent the encoding function, while the rest of the layers represent the decoding part [33], [39]. VAEs compress input data (i.e., input image of a face) into a low-dimensional latent space and reconstruct it with the learned variables, which mimic the realism of the output image by performing modifications to the facial features [35]. The process is in a couple of steps. During the training process, the original face's latent features are learned using an encoder-decoder pair and reconstruct the original face using those learned features. Then the same encoder-decoder pair is used to do the same for the target face. Finally, during generation, the decoder is switched such that the original face is subjected to the encoder of the original face, but the decoder of the target face is used, so that it generates the original face with the features of the target face [22]. The pipeline of VAE-based deepfake generation is well-illustrated in Fig. 7 by Masood et al., 2023 [22]. Usually, GAN-based generations are way more powerful and the results are more realistic than VAE-based generation, but it requires more resources to train GANs than VAEs [39]. DeepFaceLab [19], DFaker [40], and DeepFaketf [41] represent some of the best-known VAE-based deepfake generation methods [34].

2.1.2.3 Recurrent Neural Networks (RNNs) for Deepfake Audio

While video-based (or visual) deepfakes dominate deepfake misinformation campaigns, we can't underestimate the deepfakes that occur with audio-based methods. Methods such as Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks can sequence audio patterns to generate syntactically plausible audio (i.e., a speech-like audio) that mimics real audio by capturing temporal dif-

ferences in audio [35]. Lip-sync is an example of the usage of RNN for audio-visual synthetic media creation, which is used to learn the mapping between audio features for every frame and synchronize mouth movements [22], [42]. The processing mostly applies in the lower part of the face, as mouth movement along with chin, cheeks, and forehead is the most critical area in audio-visual deepfake to make it more natural and realistic [22]. GAN-based methods can also be used to improve the accuracy of the generated media during the post-processing stage [42]. In addition, methods such as WaveNet [43] use a combination of Convolutional and Recurrent Neural Networks (CNN and RNN) for audio synthesis [43].

2.1.3 Deepfake Detection Techniques

As deepfake generation techniques continue to advance rapidly, the development of deepfake detection techniques is necessary to identify deepfakes promptly, thereby preventing the spread of misinformation, for instance. As deepfakes become increasingly realistic with the support of state-of-the-art technologies, it may often be difficult or even impossible to detect such synthetic media with the naked eye. It is essential to have a good level of expertise in both deepfake generation and detection to identify subtle anomalies in a given media content. Therefore, it is better to have several approaches, such as using machine learning algorithms, forensic methods, and content authentication, rather than focusing solely on manual detection by the human eye [33].

Therefore, this chapter provides a detailed description of the techniques available for deepfake detection.

2.1.3.1 Machine Learning-Based Detection

Machine learning-based detection is considered the most common type of deepfake detection method, as deepfakes are often generated using machine or deep learn-

ing models. Machine learning-based detection largely relies on supervised learning that trains detection models using a large dataset of both real and fake media content. There are numerous machine learning models and methods available for deepfake detection based on different technologies such as Convolutional Neural Network (CNN)-based, Recurrent Neural Network (RNN)-based, and transformer-based models (i.e., Vision Transformers (ViT)).

Reiss et al. [44] proposed a model called FACTOR [45], which is a fact-checking method to detect zero-day (i.e., previously unseen) deepfakes [44]. Fake media is often accompanied by a false fact to make it more reliable in most deepfake attacks; fake video of Obama giving a speech, the attacker claims that fake media shows Obama, for instance [44]. In their work, they focus on three key areas of deepfakes, including face swap, audiovisual, and Text-to-Image (TTI) deepfake detection [44]. In the detection, it assigns a "truth score" for each claimed fact, and based on that score, the given media can be classified as real or fake [44].

Moreover, Bonettini et al. [46] proposed a CNN-based video face manipulation detection method which is an ensemble (model grouping) method for deepfake detection of videos with added attention mechanism (to identify which portion of the video is more informative for classification) and how Siamese training can be included to enhance the learning process [46]. They choose XceptionNet and EfficientNetB4 as a baseline model and four variations of it (i.e., with end-to-end training: EfficientNetB4, EfficientNetB4Att, and with siamese training: EfficientNetB4ST, EfficientNetB4AttST) in their implementation [46]. According to the results, the newly integrated attention mechanism can highlight the most detailed portions of the faces (i.e., eyes, mouth, nose, and ears), while the Siamese training approach results in clearer clustering of the outputs compared to the classical end-to-end training [46]. Therefore, combining models improves the accuracy and quality of deepfake detection rather than using them individually [46].

Furthermore, Coccomini et al. [34] proposed a novel method for video deepfake detection by using a combination model of EfficientNet and ViT by analyzing the faces extracted from the source videos [34]. They solve the problem as a binary classification problem (i.e., in layman's terms, for a given input, the model must decide between two possible outcomes: real (media content is genuine) or fake (media content is altered or synthetic)). They proposed two models called Efficient ViT and Convolutional Cross ViT [34]. Efficient ViT contains a convolutional module (i.e., EfficientNet B0) and a Transformer Encoder for feature extraction, while Convolutional Cross ViT (S-branch and L-branch) module uses a widespread (local and global) scale [34]. While Efficient ViT only focuses on small patches (i.e., pixel ranges) of face images, Convolutional Cross ViT works on a wide spread scale of patches (i.e., S-branch for small patches and L-branch for large patches) [34]. After the experiments, they concluded that the models with Cross Vision Transformer combining EfficientNet B0 as a patch extractor show a particularly marked improvement over other models [34].

Rössler et al. [29] generated a large-scale manipulated facial image dataset of 1.8 million images called FaceForensics++ [47] from 1000 videos from authentic sources and crafted 4000 fake videos [29]. The dataset was constructed using four distinct techniques: two computer graphics-based approaches (Face2Face and FaceSwap) and two learning-based approaches (DeepFakes and NeuralTextures) [29]. Moreover, they have used YouTube as the primary video source during this work [29]. Furthermore, they proposed a CNN-based automated benchmark for facial manipulation detection and evaluated the state-of-the-art detection methods using the proposed benchmark [29]. In addition, they have conducted a user study using human observation sessions to detect deepfakes of those four types, in which the average forgery detection accuracy decreased as the image quality decreased [29].

Cai et al. [48] proposed and generated a novel large scale audio-visual dataset

called Localized Audio Visual DeepFake (LAV-DF) and proposed a new multimodel method called Boundary Aware Temporal Forgery Detection (BA-TFD) for temporal forgery localization detection (i.e., to detect when tiny segment of audio/video is manipulated which is hard to detect but crucial as it can change the meaning of the entire audio/video) and compared its performance with BMN [49], AGT [50], MDS [49], and AVFusion [51] methods. LAV-DF which contains three variations of deepfake data including fake audio and fake video, fake audio and real video, and real audio and fake video, generation involve three steps including replacement of words in the transcript, use of pre-trained SV2TTS for audio generation, and use of generated audio for fake video generation using pre-trained Wav2Lip model [48]. In addition, as an extension to the previous model, BA-TFD [48], Cai et al. [52] proposed an improved multi-model called BA-TFD+ specifically for deepfake detection on audio and content-driven modifications with few modifications to the original model, including a Boundary Localization Module (instead of the Boundary Matching Layer on BA-TFD) [52]. With this replaced module, they have achieved much better results in temporal forgery detection on audio/visual deepfakes than the previously introduced model [52].

Tariq et al. [53] proposed a model called Convolutional LSTM-based Residual Network (CLRNet) [54] for detecting Deepfake-in-the-wild (DFW) videos, which are generated using unknown generation methods to the detection model's trained methods [53]. According to the training and experiment results, their method has been proven to perform well for DFW and exhibits generalization capabilities [53]. Moreover, rather than focusing on models or methods that detect deepfakes of one specific type of attack (i.e., from a single generation method in a particular dataset), their study adopted a generalized approach to detect multiple types of deepfakes in videos, especially DFW [53]. For the DFW videos, three categories of high-quality deepfakes were considered, chosen specifically to avoid the easily detectable artifacts

often present in low-quality forgeries. These categories include: deepfakes where celebrities' faces are blended into unrelated online video footage, instances where a film character is replaced with a different celebrity, and political speech videos that have been synthetically altered using deepfake techniques [53]. Furthermore, CLRNet concentrates on analyzing multiple consecutive frames to utilize spatio-temporal information rather than detecting facial manipulation on a single frame in a video, as it's vital to detect the exact spot (time frame) where the attack happened and the attacked area of that frame [53].

In addition they considered three threat models of deepfake generation including in-domain (i.e., attack set and training set contains deepfakes generated using same method for the detector), out-domain (i.e., attack set and training set contains deepfakes generated using different techniques (attack set is absent in training set) for the detector), and open-domain (i.e., train using known dataset while attack set is utterly unknown to the detector) deepfake attackers [53]. To get the generalization capabilities and effective results, they have used three approaches for training the CLRNet model including single domain learning (i.e., train using only one dataset), merge learning (i.e., merge all dataset into one large dataset and train using it), and transfer learning (i.e., first fully train using only one dataset (single domain) and then transfer learning to learn targets by taking few samples from each remaining training dataset as target dataset and training) [53]. They stated that CLRNet has well-acquired generalization capabilities when compared to other state-of-the-art methods [53].

Yan et al. [30] proposed a novel comprehensive benchmark for deepfake detection called DeepfakeBench [55]. DeepfakeBench has three modules including a data processing module ((data processing and data arrangement), a training module (has 15 state-of-the-art models in all three categories; naïve, spatial, and frequency detectors), and an evaluation and analyzing module (for evaluation; accuracy (ACC), the

Area Under the ROC (Receiver Operating Characteristics) Curve (AUC), average precision (AP), and Equal Error Rate (EER) and for analysis; Grad-CAM, t-SNE visualization)) [30]. After conducting extensive experiments, they performed two types of evaluations: within-domain and cross-domain evaluations using the AUC metric, as well as cross-manipulation evaluations [30].

Yan et al. [56] tried to answer three main research questions. The first one was that "with the notable success of image-level blending in deepfake detection, what about video-level blending? Can this method also be applied to learn general forgery artifacts in the video domain?" [56] (to understand this, they have done Facial Feature Drift (FFD) for the realistic forgery videos to identify unrealistic drifts of the faces of those video frames) and then explained how this drifting happened during the fake video generation by face swapping and then integrating the fake face into the source video to make it realistic, but eventually it made inconsistencies of fake videos [56]. They also introduced a novel video-level blending data (VB) to reproduce FFD and to create an effective detection model for FFD detection in the given deepfakes [56]. The second question was, "How do we jointly learn temporal and spatial features without heavily relying on one?" [56]. The final question was, "As videos are naturally more complex than images with multiple frames to be computed, how do we address the efficiency issue?" [56]. Therefore, in their approach, instead of developing a deepfake detection model for videos from scratch, they convert a pre-trained image model into a video model. For this purpose, they developed a lightweight Spatiotemporal Adapter (StA) that has the capability of Spatiotemporal reasoning and only trained this adapter during their training process [56].

The study by Coccomini et al. [39] performed a comprehensive analysis of the generalization capabilities of different deepfake detection methods in videos to identify the most efficient in deepfake detection in videos [39]. They considered three types of network architectures for testing, including CNN (i.e., EfficientNetV2-M)

and two ViTs (i.e., ViT-Base and Swin Transformer (i.e., Swin-Small)) [39]. All models were pre-trained on ImageNet-21K and fine-tuned on a subset from ForgeryNet [57]. During the experiments, authors followed two training approaches, namely, single-method training and multi-method training (i.e., training the models on real frames and frames manipulated using a group of methods belonging to the same category (ID Replaced or ID-Remained)), to create training sets [39]. According to the findings, under single-method training conditions, the EfficientNet-V2 convolutional network surpassed the ViT in effectively learning from less diverse and limited datasets, while the Swin Transformer yielded competitive performance [39]. In contrast, under multi-method training, the ViT exhibited enhanced generalization ability, outperforming the convolutional model in identifying frames manipulated with previous unseen techniques, but at the cost of increased data and computational requirements [39]. Additionally, cross-dataset evaluation was performed by training the three architectures on videos manipulated using either ID-remained or ID-replaced techniques from the ForgeryNet dataset, and subsequently testing them on the widely recognized DFDC Preview test set [39]. The results showed that all models were incapable of generalization based on video-level AUC results, while only Swin Transformers (i.e., Swin-Small) showed a slightly higher AUC value for cross-dataset evaluation [39].

Lai et al. [58] introduced the Generalized Multi-Scenario Deepfake Detection Framework (GM-DF) to address the significant decline in detection performance observed when models are trained directly on merged datasets. This performance drop is primarily attributed to inconsistencies in data collection scenarios and manipulation techniques [58]. The GM-DF framework comprises three core components: the Dataset-Embedding Generator (DEG), which captures dataset-specific information; the Multi-Dataset Representation (MDP), designed to learn fine-grained and localized relational features of forged patterns; and the Meta-Domain-Embedding

Optimizer (MDEO), which enhances generalization by modeling the interaction between universal features and dataset-specific embeddings [58]. For the multi-dataset training and cross-dataset testing, the authors used five datasets, including FF++ [29], Celeb-DF (V2) [59], WDF [60], DFDC [61] (only for testing), and DFF [62] [58]. Each dataset is considered as an individual domain, and all five datasets are merged into one single set, which is then further divided into training and testing sets [58]. Three evaluation metrics are used, including ACC, AUC, and EER [58]. Based on the findings of the main study, it was observed that directly merging datasets for training does not lead to improved accuracy. The pre-trained model demonstrates strong performance within its original domain, while incorporating knowledge fading in unseen samples, alongside training on Stable Diffusion and GAN-generated face forgeries may further amplify domain discrepancies and hinder learning. The effectiveness of the proposed method is higher than other baseline methods that they have done experiments with and compared [58]. Moreover, the proposed model demonstrated better detection results and a superior generalization capacity [58].

2.1.3.2 Physiological and Behavioral Analysis

When detecting synthetic media, it is most often limited to identifying anomalies in skin tone, lip-sync patterns, and the displacement of facial artifacts, such as the eyes, nose, and ears. However, when it comes to physiological analysis to detect deepfakes, there are often areas that are difficult for machines or deep learning algorithms to replicate. For instance, Li et al. [63] found that deepfake videos frequently fail to replicate natural eye blinking, which is "the rapid closing and opening movement of the eyelid" [63] rates. During their work, they have used a Long-term Recurrent Convolutional Networks (LRCN) method to detect the temporal relationship of eye blinking of consecutive frames as natural eye blinking (i.e., opening and closing of the eyelid) is considered a temporal action [63]. The developed method utilizes the

absence of eye blinking in consecutive frames to detect deepfakes [63]. However, they have proposed that it is better to consider the dynamic pattern of eye blinking, such as too fast or frequent blinking, which could be a sign of unreal content [63].

Moreover, Ciftci et al. [64] introduced a novel approach for deepfake detection, which is based on analyzing biological signals such as photoplethysmography (PPG) signals from the given content to identify whether the given media is fake or real [64]. It analyzes skin color variations caused by blood flow to detect deepfakes, according to the time of the research, no deepfake generative model can generate deepfakes with consistent PPG signals [64]. The heartbeat pattern of fake video does not exhibit the actual heartbeat pattern of a real video, which is a unique feature difference that paves the way to identify deepfake [64]. Their work not only can differentiate between fake and real videos, but also can identify deepfake generation methods (i.e., deepfakes generated using Face2Face or FaceSwap, etc.) [64].

2.1.3.3 Frequency-Based Analysis

Deepfake content contains high-frequency artifacts that deviate from real content. Therefore, frequency-based analysis can be used to identify those anomalies and detect deepfake media. Fourier Transform, specifically, Discrete Fourier Transform (DFT), is one of the methods that can serve this purpose. Convertini et al. [65] introduced a novel method using DFT to detect manipulated images that were generated using GANs (specifically using StyleGAN and StyleGAN2) [65]. By analyzing power spectrum degradation, they were able to achieve above 99% accuracy to detect deepfake images [65]. The proposed process include convert real and fake images into the frequency domain using DFT which can reveals hidden spectral artifacts of counterfeit images, extract power spectrum features to identify key differences between authentic and manipulated images, and finally feed the extracted frequency features into Support Vector Machines (SVMs) and Random Forest classifiers to

distinguish deepfake images with highest accuracy as they stated [65].

Lanzino et al. [66] proposed a novel method based on Binary Neural Network (BNN) for real-time deepfake detection of images [66]. It has a three-layered architecture with BNext as the backbone [66]. Moreover, this model included FFT and Local Binary Pattern (LBP) as additional modules to detect the traces of manipulation in the frequency and texture domain [66].

In addition, a method based on multi-level Discrete Wavelet Transform (DWT) combined with Vision Transformers (ViT) is proposed by Uddin et al. [67]. Unlike Fourier Analysis, which provides global frequency analysis, DWT provides localized frequency analysis, which makes it easier and efficient to detect artifacts in deepfakes [67]. This approach has the steps including applying multi-level DWT to decompose input images into various frequency sub-bands, capturing both global and local frequency features, and then feeding those decomposed components into ViT, which enables capturing inconsistencies across different facial regions [67]. By combining those two methods, they were able to achieve an accuracy of over 99% for two different datasets [67].

2.1.3.4 Blockchain and Digital Watermarking

Blockchain technology [68] provides a decentralized and tamper-resistant mechanism for verifying the authenticity of digital content. Each transaction (i.e., media files) can be cryptographically hashed and stored on a shared blockchain ledger, making it impossible to tamper unnoticed [68]. Choi and Kim [69], proposed a novel method called Deepfake Detection System (DDS) to detect deepfakes by make use of blockchain with smart contract, digital watermarking, and collective voting system [69]. As per their proposal, the DDS makes use of Hyperledger Fabric as a private blockchain to record deepfake detection results, preventing manipulation. At the same time, smart contracts automate decentralized detection decisions, incor-

porating a reputation-based voting mechanism where multiple sources (i.e., Internet Content Providers (ICPs)) validate deepfake content individually to set the collective voting score [69]. Additionally, digital watermarking embeds metadata into media files, enabling traceability and tamper resistance of media files [69]. Moreover, Qureshi et al. [70]. Lai et al. [71], and Alattar et al. [72] proposed different methods and systems to detect deepfakes using digital watermarking.

2.1.3.5 AI-Powered Deepfake Detection Tools

As misinformation comes in many forms, it's essential and more or less valuable to make use of AI-powered tools in deepfake detection [73]. With the recent rapid development and rise of AI-powered tools, deepfake generation as well as deepfake detection was no exception. Not only for regular deepfake detection, which requires considerable effort, computing resources, and time, but also for real-time deepfake detection, several open-source and commercial AI-powered tools have been developed in recent years. The Microsoft Video Authenticator [73] is one of such tools. It analyzes the given video or still photo and assigns a confidence score as a percentage in each frame (i.e., for videos) in real-time to identify whether the provided content is manipulated or not [73]. According to the published article [73], it detects what might be undetectable to the naked human eye, such as blending boundaries of synthetic media and subtle fading or greyscale elements [73].

Furthermore, the commercialized tool called Sensity AI [74] was recently introduced, featuring numerous AI-powered real-time deepfake detection capabilities. This tool can detect face manipulation techniques such as face swap, lip-sync, and face reenactment at the pixel level with a higher confidence score as a percentage [74]. Not only that, but it also enables users to download a comprehensive report after the analysis, which includes many more details. In addition, Sensity AI provides forensic analysis of the uploaded file to check whether it has been tampered

with and, if so, how. It also features another detection tool to identify AI-generated images [74]. Moreover, there are tools such as Deepware Scanner, Google SynthID, Intel FakeCatcher, etc. [75].

2.2 Misinformation Campaigns

Misinformation or disinformation campaigns can be used to spread fake news, manipulate public opinion, fabricate narratives, statements, or events to deliver false messages to the listeners or audience that pave the way to confusion, reputation damages, or financial fraud, etc. [76]. The rise of social media and powerful AI tools makes it easy to generate and spread such misinformation on a large scale in a short period.

This section examines the motivations behind misinformation campaigns and their effects, and several case studies illustrate the consequences of such campaigns. Additionally, it discusses existing detection methods used to combat misinformation.

2.2.1 Motives Behind and Impacts

Misinformation campaigns can be motivated by several aspects, including, but not limited to, political, financial, social, ideological, and personal factors. It can have far-reaching consequences, including political instability, economic downturns, social disintegration, and damage to personal reputation.

2.2.1.1 Political Manipulations

Governments or political parties often utilize misinformation to divert public opinion, discredit their opposition, and thereby gain political benefits, such as favorable electoral outcomes. Moreover, on a large scale, these campaigns can be used to turn citizens against a ruling government. For instance, during the 2016 US election,

it was reported that large-scale fake news campaigns were deployed through social media and fake news web pages, which may have influenced the electoral outcome [2].

2.2.1.2 Financial Gains

Besides deepfake-based financial frauds, such as voice phishing (vishing) [35], misinformation campaigns can also be involved in economic manipulations, including stock markets and cryptocurrency markets, or promote fraudulent products. For instance, the "pump-and-dump" scheme in cryptocurrency markets, which is used to artificially inflate cryptocurrency stock prices (pumping) just before selling at a high profit (dumping) [77]. Since prices are artificially inflated using fake signals, the prices usually drop, and buyers are at a loss [77].

2.2.1.3 Social and Psychological Influence

Regarding social and psychological influences, misinformation campaigns can be used to initiate fear, erode trust in reputable institutions, or create division among them. For instance, during the latest COVID-19 global outbreak, numerous fake news campaigns were disseminated, particularly on social media, promoting COVID-19 conspiracy theories, counterfeit cures, inefficiencies, and side effects of several vaccines, among other claims, which had a significant social and psychological impact [78]. In addition, misinformation campaigns can lead to public panic and even critical chaos by spreading fake emergencies or fabricated disaster footage, for instance.

2.2.1.4 Personal Attacks and Reputational Damages

Civilians or high-profile individuals can be targeted using deepfake-based technologies to damage their reputation or even for blackmail. The creation of non-

consensual intimate images/videos which are often refereed to as "revenge porn" becoming a rising form of personal attacks and reputations damages [3] with the increased accessibility of the deepfake tools such as so called "AI undressing tools" [79]. The victims of those cases, especially women, have devastating effects such as having to vanish from the Internet entirely, mental issues, and constant fear of retraumatization, etc. [80].

2.2.2 Case Studies Analysis

This chapter examines several case studies involving deepfake misinformation campaigns.

2.2.2.1 The Slovak Case (2023)

The so-called "Slovak Case" is an outstanding incident that occurred during the 2023 Slovak parliamentary elections, involving AI-driven misinformation in the democratic process. Just two days prior the election, during the "silent time" of the election in which it is prohibited to any media discussion related to the election, an audio clip went viral on Facebook capturing a conversation between Michal Šimečka, leader of the liberal Progressive Slovakia party (Pro-European), and journalist Monika Tódová discussing about orchestrating electoral fraud, including buying votes from marginalized communities such as "Roma" minority [81]. Both Šimečka and Tódová announced that the audio was fabricated, and fact-checkers, including the Agence France-Presse (AFP), identified the signs of AI manipulations of the audio clip [81]. Additionally, since the misinformation was in audio format, it bypassed Meta's manipulated-media policy, which only applies to fabricated videos, further demonstrating a policy loophole in the Meta platform [81].

The release of this deepfake audio clip during a critical period illustrated potential regulatory gaps and the rapid spreading capabilities of social media. The inci-

dent not only cast doubt on the integrity of the election process but also highlighted the unique challenges that existing deepfake detection or fact-checking systems face in countering deepfake misinformation. Furthermore, the victory of Pro-Russian SMER over the Pro-European Progressive Slovakia highlighted the potential inference of deepfake misinformation campaigns on election outcomes, although it was not the only reason for their defeat [81]. This case often called as "dawn of a new era of disinformation" and a "test case" of how vulnerable can be the current democratic processes in the era of AI-driven interference and emphasized the importance of strong regulatory frameworks in elections as well as proper utilization of deepfake detection tools specially integrated in social media platforms [82].

2.2.2.2 AI-Generated Deepfake Voice in Polish Election Advertisement (2023)

In August 2023, the largest opposition party in Poland, the centrist Civic Platform (PO), released an AI-generated deepfake voice of the Prime Minister (PM) Mateusz Morawiecki as part of PO's election campaign advertisement, in which Morawiecki reads alleged leaked emails of the PM's former chief of staff, Michał Dworczyk [83]. Only after a wide range of criticisms from commentators and experts, PO urged clarification that only some parts of the video were genuine, while other parts were fabricated using an AI-generated voice of the PM [83]. This advertisement aimed to criticize the contradiction of the PM's policies [83].

However, failure to disclaim AI-generated content led to a discussion on the ethical aspects of the political campaigns, which demonstrates a lack of transparency and contributes to public confusion. On the other hand, it is a prime example of using deepfake for the reputational damage of an individual. Additionally, it is emphasized that the importance of clearly labeling or marking AI-generated content when releasing it on social media is crucial, as mandated by the European Union's

AI Act [83]. Furthermore, it highlights the continued lack of deepfake detection tools and their limited usage on social media platforms, such as X (formerly known as Twitter), in this instance.

2.2.2.3 Deepfakes in Warfare: The Russian-Ukraine Conflict (2022)

In February 2022, a deepfake video of the Ukrainian President Volodymyr Zelensky was deployed online and simultaneously broadcast via a hacked Ukrainian news website's feed, Zelensky stating that the war with Russia was over and ordering the Ukrainian armed forces to surrender [84]. Despite the unsophisticated nature of the video [85], the strategic and psychological impact was significant, with Zelensky's face slightly appearing out of sync with his head. A similar sort of fabricated video of Russian president Putin [86] was also circulated online announcing the defeat of Russia [87].

These deepfake misinformation campaigns illustrate a new era of cyber warfare, which could lead to confusion, destabilization of societies, weakening of morale, and manipulation of war fronts. Moreover, it eroded trust in reputable media and news sources, which had previously delivered reliable and timely important news to the public. Although, some videos created on educational or entertainment purposes by the Ukrainian government to educate people about the conflict such as placing Putin into famous films like Downfall and The Great Dictator [84], it make it harder to the public to differentiate between factual and fabricated media which eventually ended up in more significant conflicts.

2.2.2.4 The UK 2025 General Election Fear of Deepfakes

Although no involvement of deepfakes in the UK's general election 2025 has been reported yet [88], many reports and surveys predicted the fear of deepfake misinformation campaigns would take place, and the UK suspected that they are not

quite ready for such a situation [89]. According to the Institute for Government, 70% of MPs were worried about deepfake involvement in the 2025 general election in the UK [90]. Since the 2019 general election, deepfake technologies have advanced rapidly. The Center for Policy Studies labelled this election as the UK's "first deepfake election" [90].

Furthermore, according to the survey conducted and the report published by the Alan Turin Institute, 87.4% of people in the UK are worried about deepfake's involvement of effecting the election results while 91.8% of people concerned about deepfake's involvement in child sexual abuse material which is also considered an alarming concern [91]. Despite the available technology and regulatory frameworks, the researchers recommended that it is better to urgently tackle the threats posed by AI and deepfakes, especially concerning the UK's general election ahead [91].

2.3 Existing Works and Their Limitations

The rapid evolution of deepfake generation techniques has led to extensive research into automated detection methods. Numerous deep learning-based approaches have been proposed, leveraging CNNs, attention methods, and spatio-temporal modeling as discussed above. Despite achieving notable performance on existing benchmarks, these models exhibit several limitations that affect their effectiveness in real-world applications. This section examines existing works and highlights their key shortcomings in terms of generalization, robustness, compression sensitivity, and computational efficiency.

2.3.1 Generalization Limitations Across Datasets

A fundamental limitation of many deepfake detection models is their inability to generalize effectively across different datasets. Models such as XceptionNet [92] and

MesoNet [93], etc., achieve high accuracy when evaluated on the dataset used for training but suffer from significant performance drops when tested on open-domain datasets [29]. For instance, Rossler et al. [29] reported high accuracy using XceptionNet on the FaceForensics++ dataset. However, when the model was evaluated on Celeb-DF [59], performance declined sharply due to differing visual artifacts and synthetic techniques. Similarly, Verdoliva [94] and Tolosana et al. [95] emphasized the domain dependency of detection models, noting that many models overfit to the visual characteristics of specific datasets that they trained on. As a result, these models lack robustness in open-domain settings where the source of the manipulated content is unknown.

However, works such as Tariq et al. [53], Li et al. [96], Yan et al. [30], etc. introduced models and benchmarks that can have the generalization ability in deepfake detection. For example, one of those introduced models is called CLRNet [53], which will also be tested in this study. However, in that study, a known limitation was specified. The authors of CLRNet only considered high-quality videos in their experiments; therefore, this study will evaluate that model using three different levels of video compression.

However, cross-dataset evaluation remains an underexplored area, despite being crucial to building robust and trustworthy deepfake detection systems.

2.3.2 Vulnerability to Compression and Resolution Variations of the Data

Another major challenge is the sensitivity of existing models to video compression and degradation in resolution. Most real-world video content, especially on social media platforms, undergoes multiple rounds of lossy compression before being published to the audience. These compression artifacts often erase subtle deepfake traces, such as blending artifacts, unnatural transitions, or any inconsistencies in

facial warping.

Moreover, works such as [29] and [59] considered how the accuracy and performance of the selected models were affected by the compression levels of the specific datasets introduced by the authors. It showed that detection accuracy declined substantially at higher compression levels, such as C40, in many cases, while with few exceptions [29] [59]. However, as a significant limitation, cross-dataset evaluation of different compression levels of various datasets was not considered in a single study.

2.3.3 Underreporting of Inference Time and Resource Efficiency

In addition to the detection accuracy, inference time and computing resource usage are critical for identifying the real-time deployability, especially on edge devices or streaming platforms where computing resources are constrained. However, many published models do not report inference time and/or CPU/GPU memory usage when evaluating the deepfake detection models. Lightweight models such as DefakeHop++ [97], MobileNetV3 [98], and Liu et al. [99] have been proposed in deepfake detection focused on both images and videos, but only inference time is recorded and in order to have a clear understanding of deepfake detection model deployability on edge devices, it is crucial to measure computing resource usage as an additional metric as well.

Even when models are tested on GPUs/CPUs, studies rarely benchmark inference time or memory usage under varying batch sizes or video quality (or compression) levels. This lack of attention to practical deployment metrics creates a gap between research outcomes and industry needs.

2.3.4 Addressing the Gaps: Contributions of This Work

To overcome the above limitations, this study evaluates three state-of-the-art models, BA-TFD+, Convolutional Cross Efficient ViT, and CLRNet, on five datasets, namely, Celeb-DF (v2), DeeperForensics-1.0, DFD, DFW, and LAV-DF, and under three standard video compression levels (C0, C23, and C40). The following key innovations distinguish this work.

1. **Unified and multi-dataset testing for measuring the generalization ability and compression-level performance:** Five diverse datasets and three H.264 quality tiers give a 15-cell test grid per pre-trained deepfake detection model that exposes hidden failure modes and the ability of generalization.

2. **Inference and resource profiling:** GPU/CPU memory usage and inference time are measured to assess model efficiency and deployment feasibility.

3. **Visualization of evaluation results:** Performance is visualized using various graphs, heat maps, and plots, helping identify misclassification patterns across datasets and compression settings.

4. **Holistic perspective:** Technical findings of the study feed directly into deployment architectures and policy recommendations, bridging the usual gap between laboratory metrics and societal impact.

3 Evaluation of the Detection Models

This chapter describes the selected pre-trained deepfake detection models and datasets. Furthermore, the experimental setup and testing process are presented. This experiment evaluates the effectiveness and robustness of pre-trained deepfake detection models under real-world misinformation scenarios. The evaluation focuses on:

RQ1: Measuring how well existing models detect deepfakes in real-world misinformation settings.

RQ2: Assessing model performance under different video compression conditions (RAW, medium compressed, and highly compressed).

3.1 Pre-trained Model Selection, Datasets Selection, and Preprocessing

This section discusses the criteria for pre-trained deepfake detection model selection, dataset selection, and how the data preprocessing is done. Both dataset selection and preprocessing (in most cases) are done locally using Ubuntu 24.04.2 LTS in a conda environment with the notebook laptop of Intel Core i7-1165G7 @ 2.80GHz, Intel Iris Xe graphics, and 8GB RAM. Inside the conda environment, packages such

as random, cv2, and FFmpeg are being used. The model evaluations and some data preprocessing were conducted utilizing the university-allocated server and container, which runs on JupyterLab notebook as a front-end application. It contains two NVIDIA GeForce RTX 4090 GPUs, each with 24564 MB memory capacity. It has NVIDIA-SMI 560.35.05, Driver Version: 560.35.05, and CUDA Version: 12.6. Moreover, within the university container's JupyterLab environment, packages such as Tensorflow, PyTorch, CUDA, OpenCV, seaborn, and Pandas, etc. were used both for frame extractions, model evaluations, and performing the calculations.

3.1.1 Overview of the Selected Pre-trained Models

This subsection discusses the selected pre-trained deepfake detection models by highlighting the reasons for selection as well as the architecture, strengths, weaknesses, and limitations of each model. The following table 3.1 outlines the selected pre-trained deepfake detection models known for their high accuracy and efficiency.

Table 3.1: Selected pre-trained deepfake detection models.

Model Name	**Architecture**	**Pre-trained**	**Strengths**	**Repository**
BA-TFD+	3D CNN	LAV-DF	Detection using temporal forgery localization with combination of ViT	BA-TFD+
Convolutional Cross Efficient ViT	CNN with cross ViT	DFDC, FF++	Better performance when combining and able to capture a wide range of details	Cross ViT
CLRNet	LSTM	DF, FS, F2F, NT, DFD	Better generalization	CLRNet

3.1.1.1 BA-TFD+ Model

Boundary Aware Temporal Forgery Detection Plus (BA-TFD+) [52] is an improved version of the original BA-TFD model introduced by Cai et al. [48]. Unlike most deepfake detection models, which are purely frame-based detectors, BA-TFD+ is specifically designed to detect deepfakes by capturing the video frames' boundary

artifacts and temporal inconsistencies [52]. It focused on learning both spatial artifacts and temporal dynamics, which often leave subtle traces of manipulations over a sequence of frames [52]. The model emphasizes the boundaries of faces and regions where manipulations are likely to occur and captures inconsistencies between consecutive frames to enhance the accuracy and performance of the model in detecting synthetic media segments [52].

BA-TFD baseline model has a 3D Convolutional Neural Network (3D CNN) based architecture, and the improved version, BA-TFD+, has the same architecture but with the replacement of the backbone with Multiscale Vision Transformers [52]. Moreover, frame extraction is utilized by a lightweight CNN backbone (frame classification module) while specialized convolutional layers are used to emphasize boundary (or edge) inconsistencies (boundary localization module) [52]. Furthermore, the BA-TFD+ model comprises visual and audio encoders to process raw video frames and audio as inputs and a multimodal fusion module to fuse multimodal latent features in a broad spectrum [52].

The model was initially trained and tested on their proposed Localized Audio Visual DeepFake (LAV-DF) dataset. In addition, it was tested on different standard datasets such as ForgeryNet and Deepfake Detection Challenge Dataset (DFDC) and evaluate the performance and accuracy of the model using Area Under the Curve (AUC), Average Precision (AP), Average Recall (AR) in different thresholds such as 0.5, 0.75, and 0.95 for AP and 100, 50, 20, and 10 for AR [52]. As per the results obtained, it was claimed that the novel method had outstanding performance in each case compared to the previous techniques [52].

The BA-TFD+ model pre-trained on the LAV-DF dataset was used in this study because it represents a unique, temporal-aware architecture, focusing on boundary inconsistencies and consecutive frame relationships in deepfake detection. Its design best aligns with real-world situations where temporal localization often occurs in

synthetic media. In addition, it can serve as a case study to experiment with generalization ability as it is only trained on one dataset, and it allows for evaluating how well it will perform in unseen datasets as well.

3.1.1.2 Convolutional Cross Efficient ViT Model

Cross-Efficient ViT is a deepfake detection model based on a combination of EfficientNet B0 as a feature extractor and different types of Vision Transformer (ViT) architectures to learn subtle manipulations distributed across multiple frames [34]. By combining those two architectures, this model aims to balance the computational efficiency and the model's performance at its best. The Convolutional Cross ViT architecture allows for identifying the artifacts introduced by deepfake generation on both local and global scales using two branches called S-branch for small patches and L-branch for large patches of synthetic media [34]. In this model, EfficientNet (i.e., EfficientNet B0) acts as the backbone, and the classification is treated as a binary classification problem. Within the EfficientNet B0, frame extraction is performed by using the MTCNN method [34].

The model was trained and tested using DFDC and FaceForensics++ (FF++) datasets, and AUC and F1-Score are used as the evaluation metrics for the performance evaluation of each model proposed by the authors [34]. Based on the experiments and results, one of the proposed models achieved the best results, while Convolutional Cross ViT combined with EfficientNet B0 as a backbone model recorded impressive results as well in each of the two datasets.

This model was selected for this study due to its hybrid architecture, the ability to identify a wide range of deepfake manipulations in synthetic media using local and global patch classification, and the combined strengths of both CNN and ViT architectures. Moreover, it serves as a great example of intermediate architecture of lightweight CNN-based architecture and heavy-weight Vision Transformers-based

architecture, providing a better trade-off between computational efficiency and detection performance.

3.1.1.3 CLRNet Model

Convolutional LSTM-based Residual Network (CLRNet) was proposed as a method focused on the generalization ability of deepfake detection, which claimed to perform well not only in deepfakes generated using known generation methods, but also deepfakes generated using unknown generation methods such as Deepfake-in-the-Wild (DFW) [54]. Instead of combining LSTM with a CNN architecture, which can be unstable in transfer learning, the authors opted to use Convolutional LSTM cells in their architecture to achieve more stable training and improved performance [54].

During the training of this model, three unique training techniques including single domain learning (i.e., train the model only on one dataset), merge learning (i.e., train on a combined dataset of all selected datasets), and transfer learning (i.e., first fully train the model on one dataset and then train the model using small subsets of remaining datasets to transfer (or extend) the leanings) were used to defend against in-domain (i.e., training and attacking datasets are subsets of the same known dataset to the detector), out-of-domain (i.e., training and attacking datasets are from different datasets but the detector knows both datasets), and open-domain attacks (i.e., training set is a known dataset to the detector while the test set is an unknown dataset) [54]. The model trained on DeepFake (DF), FaceSwap (FS), Face2Face (F2F), Neural Textures (NT), Deepfake Detection (DFD), and Deepfake Detection Challenge (DFDC) datasets and deepfake generation methods while tested on all those datasets as well as DeepFake-in-the-Wild (DFW) dataset [54]. Based on the experiments and results presented, it is claimed that the CLRNet model has the best performance against the state-of-the-art models and better generalization ability based on F1-Score as the evaluation metric [54].

In this evaluation, the CLRNet was selected for its state-of-the-art generalization ability to evaluate the model under different sets of datasets and compression levels of data. Moreover, the pre-trained CLRNet model trained on all datasets was used in this study for the evaluation. It is also worth noting that the DFW dataset used in the CLRNet study for testing and the DFW dataset used in this study are different.

Unlike the other two selected pre-trained models, the CLRNet model had compatibility issues as it was trained on earlier versions of TensorFlow and Keras with an earlier version of the Python environment, and when it comes to extracting values and variables of the pre-trained model, it generates errors continuously. Therefore, the CLRNet model first had to be converted to be compatible with the updated versions of TensorFlow, Keras, and the Python environment. A Python script used to export the original pre-trained model into a compatible version using SaveModel is outlined in Appendix Listing 4.

3.1.2 Overview of the Selected Dataset

In this work, five mainstream datasets were selected for pre-trained model testing. Table 3.2 outlines the dataset chosen for evaluation of the selected pre-trained models, with respective statistics of total, authentic, and fake videos included in each dataset. Some of the datasets have been released to the public archives, while other datasets had to be requested by the respective owners for permission to download and use.

Table 3.2: Selected datasets for evaluation of pre-trained models.

Dataset	**Total videos**	**Real**	**Fake**	**Selected real**	**Selected fake**
DeeperForensics-1.0 (2020)	59475	48475	11000	500	500
Celeb-DF (v2) (2020)	6229	590	5639	500	500
LAV-DF (2023)	26100	6906	19194	500	500
DFD (2019)	3432	364	3068	250	250
DFW (2021)	1869	0	1869	0	500
Total	**97105**	**56335**	**40770**	**1750**	**2250**

DeeperForensics-1.0 [100] was the largest among the selected datasets, consisting

of a total of 59475 videos, with 48475 real and 11000 manipulated videos. It is well-known for "good quality, large scale, and high diversity" [100]. They used 100 paid actors from 26 countries to make the source videos and then used a newly invented many-to-many end-to-end method for manipulated face-forgery video creation [100]. Furthermore, Celeb-DF (v2) [59] dataset consists of a total of 6229 videos, which include 590 real and 5639 fake videos. The source videos of Celeb-DF (v2) were collected from YouTube with celebrities of different ages, ethic groups, and genders [59]. The authors have used an improved DeepFake synthesis algorithm to generate high-quality synthesis media out of the source videos compared to other available datasets at the time of generation and release [59]. In addition, Localized Audio Visual DeepFake Dataset (LAV-DF) [52] consists of 26100 videos in its "test" dataset. However, they have not clearly divided it into a folder structure, such as real and fake, but a metadata file was included. Therefore, a separate Python script outlined in Appendix Listing 5 separates fake and real videos into two folders. However, the LAV-DF complete dataset contains a total of 136,304 videos, including 36,431 real and 99,873 fake videos, and the full pipeline of LAV-DF dataset creation is explained in the paper [52].

Moreover, Google and Jigsaw Deep Fake Detection (DFD) [101] dataset is a collection of a total of 3432 videos, which includes 364 real videos using 28 paid actors in 16 different scenes and 3068 synthesis videos generated using deepfakes. Finally, DeepFake in the Wild (DFW) [102] dataset, which was collected from two online video streaming and hosting services (YouTube and Bilibili), and the generation method is unknown. It consists of 1869 deepfake videos and no real videos, although the author claimed that DFW has both real and fake videos; there was no separate folder structure or metadata file for separation. Therefore, the DFW dataset is considered "all fakes", generated using unknown generation methods. Therefore, this particular dataset is important to check how the selected models react when

there are only data from one class is present (i.e., only "real" or only "fake") as most of the models make the predictions by treating it as a binary classification problem (i.e., classify given data as either "real" or "fake"). The initial plan was to also include the FaceForensics++ (FF++) dataset in the evaluation, as it is a diverse dataset with an extensive amount of real and fake data. However, due to specific permissions and storage restrictions, it was excluded.

To ensure fair and unbiased evaluation across datasets of varying sizes, a fixed number of real and fake samples (500 or 250 per class) in a 1:1 ratio were selected from each dataset as illustrated in Table 3.2. This balanced sampling approach avoids dataset dominance, supports cross-dataset comparability, and ensures computational efficiency during evaluation. DeeperForensics-1.0, Celeb-DF (v2), DFD, and LAV-DF had clear folder structure for real and fake videos and based on the size of the dataset, 500 real and 500 fake videos selected randomly for DeeperForensics-1.0, Celeb-DF (v2), and LAV-DF while 250 real and 250 fake videos were chosen from DFD dataset. Furthermore, DFW only consists of fake videos with unknown generation methods, and to match the ratio, 500 fake videos were selected from DFW. The script for this random video selection is attached in Appendix Listing 6.

3.1.3 Preprocessing of Dataset

This section describes the process of dataset preprocessing before feeding it into the respective pre-trained models.

3.1.3.1 Data Preprocessing for the BA-TFD+ Model

As per the first pre-trained model to be tested, the BA-TFD+ model mainly requires three inputs to detect the deepfakes. It requires pre-trained weights, video files (i.e., ".mp4" format in this model), and a metadata file (in JSON format), which includes details about the data in each dataset as inputs. Since there are five datasets

available, metadata for each dataset had to be generated, except for the LAV-DF dataset, as the authors had already provided it. Additionally, the metadata file for the DFW dataset was generated separately, as it only contains fake videos collected from the Internet without referencing the original videos. It is worth reminding that the BA-TFD+ model was pre-trained using a subset of the LAV-DF dataset, and a subset of the test dataset of the LAV-DF dataset is used for this experiment. The Python script used to generate a metadata file for each dataset (except LAV-DF and DFW) is outlined in Appendix Listings 7 and 8. Moreover, after generating metadata files, all video files for a respective dataset should be placed in one folder, as the metadata file contains information on which video data is real or manipulated. This will make the process less complicated for the model to be tested.

Furthermore, by using the FFmpeg library supported by the libx264 (i.e., H.264) video encoder, the three varieties of videos for each dataset based on compression levels were generated to perform the experiments on each pre-trained model to determine how well each can perform when the quality level of the data is degraded. The compression levels, namely C0, which is considered RAW (or almost 0% compressed) with high quality videos, C23, which is regarded as moderate or lightly compressed (around 30% compressed) with moderate quality videos, and C40, which is considered highly compressed (around 50% compressed) with low quality videos. The Python script compressing the videos into three levels is included in the Appendix Listing 9. The general folder structure is illustrated in Listing 1 for all datasets.

3.1.3.2 Data Preprocessing for the Convolutional Cross-Efficient ViT Model

As per the Cross Efficient ViT model's requirements, the dataset preprocessing is done in advance before feeding it into the model during the testing phase. As per the previous steps for data preprocessing of the BA-TFD+ model, the video

Listing 1 General folder structure of extracted frames in BA-TFD+ model evaluation.

```
Datasets/
    C0_video/
        DATASET_NAME/
            test/
                video_000.mp4
                video_001.mp4
                ...
            (same structure for each dataset)
    C23_video/
        ...
    C40_video/
        ...
```

datasets containing three compression levels are already prepared and hosted in the university's server allocated for this work. Therefore, preprocessing for this model was done using those video datasets. Unlike the BA-TFD+ model, the Cross Efficient ViT model does not require the videos as input, but the extracted crops of faces from each video frame in .png format. First, a CSV file containing the labels of the videos of each dataset is required for testing. Therefore, the Python script outlined in Listing 10 was used to generate relevant CSV files based on each dataset. As the next step, the model's preprocessing unit detects the faces of each video frame and then extracts the resized crops, which are then ready to be fed into the model for classification. The extracted crops of detected faces is in the size of 225 by 225 and if the module is unable to detect faces or extract crops that match this standards, it either resize or completely discards the relevant frames of that suspected video, so that model only fed with clear and a standardized dataset make the decision. The extracted frames were structured inside the project folder as the folder structure mentioned in 2.

3.1.3.3 Data Preprocessing for the CLRNet Model

First of all, all the datasets had to be compressed based on three compression levels, C0 (raw), C23, and C40, as mentioned in 3.1.3.1. However, the compressed versions

Listing 2 General folder structure of extracted frames in Cross Efficient ViT model evaluation.

```
datasets/
    c0/
        dataset/
            test_set/
                DATASET_NAME
                    video_name
                      0_0.png
                      1_0.png
                ...
            (same structure for each dataset)
    c23/
        ...
    c40/
        ...
```

of datasets had to be in two separate folders for frame extraction and then for feeding to the CLRNet model. Therefore, the compression had to be performed again based on new requirements using the Python script outlined in Listings 11 and 12, and the compression was performed locally. As per the requirement of the CLRNet pre-trained model, input is required in ".png" format as frames (videos are a series of consecutive collections of still images called frames), not as a video format. Therefore, the videos of the selected datasets had to be preprocessed before feeding them into the detection model. The Python script is outlined in Appendix Listings 13 and 14, which is used to extract frames from each video of a selected subset of datasets, which was performed on the university server. Each video is preprocessed and extracts frames with 10 frames per video and resized to 128 by 128, and the general folder structure is illustrated in Listing 3, except for the DFW dataset due to the unavailability of real videos in it. Therefore, DFW has only a "fake" video subfolder with the same folder structure as the other datasets.

Listing 3 General folder structure of extracted frames in CLRNet model evaluation.

```
extracted_frames/
    C0/
        DATASET_NAME/
            fake/
                video_name
                  video_name_frame_name0.png
                  video_name_frame_name1.png
                ...
            real/
                video_name
                  video_name_frame_name0.png
                  video_name_frame_name1.png
                ...
            (same two folders for each dataset and a separate folder for each
            video name)
    C23/
        ...
    C40/
        ...
```

3.2 Testing and Benchmarking

This section describes how the testing of each pre-trained model is done to evaluate the model's performance, generalization ability, and computational efficiency based on different sets of metrics.

3.2.1 Metrics for Evaluation

This section explains the varieties of metrics used in the evaluation for the pre-trained models to assess the model's performance, generalization ability, and computational efficiency.

As the first step, it is worth clearly understanding the four basic terms, especially in binary classifications. When performing a binary classification, the prediction by the model (i.e., positive or negative) is compared against the ground truth value. Therefore, every sample of a considering dataset exactly falls under one of four brackets: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). TP means the count of samples, the model correctly labeled positive

(i.e., a truly "fake" video, the model correctly flagged as "fake") while TN means the count of samples, the model correctly labeled negative (i.e., a truly "real" video, the model correctly leaves unflagged). On the other hand, FP is the count of samples the model incorrectly labeled positive (i.e., a truly "real" video, the model incorrectly flagged as "fake") while FN is the count of samples the model incorrectly labeled negative (i.e., a truly "fake" video, the model incorrectly flagged as "real"). While TP and TN do not negatively impact the model's accuracy, FP raises a false alarm, and FN is considered a missed detection, which negatively affects the model's accuracy and performance when measuring.

3.2.1.1 Precision (Pr)

Of all the samples the model predicted positive, how many were actually positive is meant by precision. Precision is used as a metric to measure the performance of a selected model. It can be demonstrated using the equation presented below. On one hand, high precision (close to one) means that when a model flags something as positive, the prediction is always almost correct. On the other hand, low precision (close to zero) means it has many false alarms (i.e., FP). It is possible to have a better precision value by raising the decision threshold, but then the model may suffer from a decrease in recall. At the industrial level, false alarms can be expensive to investigate, which consumes time, finances, and human resources. Therefore, it is one of the crucial factors in measuring a model's performance in deepfake detection.

$$\text{Precision} = \frac{\text{correctly classified actual positives}}{\text{everything classified as positive}} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

[103]

3.2.1.2 Recall (Re)

Of all the actual positives, recall means how many of the models were identified correctly. Recall is also known as sensitivity, as well as the TP rate, and is used as a metric to measure the performance of a given model. The formula to measure the recall is outlined below. On the positive node, high recall (close to one) means the model missed almost no positives. In other words, the model is very sensitive. On the other hand, a low recall (close to zero) means the model leaves many positives undetected. It is possible to have a better value for recall by decreasing the decision threshold, but then the precision may suffer. In a critical task such as deepfake detection, it is necessary to have a high recall rate, and therefore, recall can be considered as another critical factor when measuring the performance of deepfake detection models.

$$\text{Recall} = \frac{\text{correctly classified actual positives}}{\text{all actual positives}} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

[103]

3.2.1.3 Area Under the Receiver-Operating Characteristic (ROC) Curve (AUC)

In order to understand the AUC, it is crucial to have a better understanding of the ROC curve beforehand. ROC curve can be represented as a confusion metric with x-axis being the False Positive Rate (FPR) while the y-axis being the True Positive Rate (TPR) [104]. The ROC curve can be drawn between (0,0) and (1,1) by calculating TPR and FPR at every possible threshold [104]. If all the thresholds are ignored, a perfect model can be represented at the point of (0, 1) where FPR is 0 and TPR is 1 [104]. The formulas for TPR and FPR are as follows.

$$\text{True Positive Rate (TPR)} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \qquad \text{False Positive Rate (FPR)} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

[105]

"AUC represents the probability that the model, if given a randomly chosen positive and negative example, will rank the positive higher than the negative" [104]. It measures the Area under the ROC curve [106]. A higher AUC (close to one) means a better performance of a model with the ability to better distinguish between classes, while an AUC of 0.5 means random guessing [106]. Furthermore, if AUC is less than 0.5, the model's performance is even worse than that of random guessing. Moreover, unlike precision, recall, and F1-score, AUC is threshold independent, which summarizes performance across all operational points. AUC is an important metric in cases where no specific decision threshold is chosen for model comparison, and also when the dataset is balanced and needs to evaluate the performance over all the threshold values [106].

3.2.1.4 F1-Score (F1)

F1-score means the harmonic mean of precision and recall, which can be demonstrated using the following formula. The key distinction between the arithmetic mean and the harmonic mean lies in the way they handle imbalanced values: unlike the arithmetic mean, the harmonic mean significantly penalizes situations where either precision or recall is low (close to zero). Therefore, the model only returns high F1-score, only if both precision and recall returns higher values. Higher F1-scores (close to one) are considered best while lower scores (close to zero) are considered worst. Moreover, it can be considered as a single metric calculated based on precision and recall. F1-score can be considered a great tool to measure a model's performance in cases where raw accuracy itself can be misleading.

$$F_1 = 2\,\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2\,\text{TP}}{2\,\text{TP} + \text{FP} + \text{FN}}$$

[107]

3.2.1.5 Average Inference Time (AIT)

Average Inference Time measures how long the model takes to process one sample of a given dataset (i.e., one video or one frame). It usually reports in milliseconds per frame (ms/frame) or seconds per video (s/video). This work reports AIT as a s/sample (i.e., video or frame). Therefore, this metric can be considered as a measurement to detect the selected model's computational efficiency and evaluate the model's real-time applicability. It is a crucial metric when deciding whether the chosen model is suitable for deployment in live systems such as video surveillance or social media platforms, for instance. If the value is lower, it implies faster inference and better performance for real-world applications.

3.2.1.6 GPU/CPU Memory Usage (Me)

While the model is running the inference, the peak memory consumption (RAM for CPU or VRAM for GPU) is tracked. This helps to identify how resource hungry the considering model is. In this work, CPU memory is recorded in GB. Therefore, this metric is used to identify the computational efficiency of the selected models. It also helps ensure that the selected model fits the targeted hardware availability (i.e., can this model run on resource-constrained environments such as mobile devices, or does it require more computing power, such as a high-power data centre). For instance, if it has lower memory usage, it can be considered a lightweight model with greater scalability.

4 Results and Discussion

This section comparatively analyzes the results recorded during the experiments and represents them in Table 4.1 based on three critical aspects: model performance, generalization ability, and computational efficiency of the tested models.

4.1 Evaluations and Results

As per the requirements of each model, datasets, dataset preprocessing, and after carefully troubleshooting each model's errors, such as compatibility issues, the model evaluation was conducted on all three pre-trained models in the university-provided server container using five selected datasets, each with three compression levels. The specifications of the university server container were explained at the beginning of chapter 3 under section 3.1. For each model, 15 evaluations were conducted, and a total of 45 assessments were conducted across three pre-trained models. In each case, values returned for six evaluation metrics were calculated as shown in Table 4.1 and are analyzed in the subsequent section.

Table 4.1: Performance of pre-trained deepfake detection models on different datasets and compression levels

Dataset	Compression	BA-TFD+					Cross Efficient ViT					CLRNet				
		Metrics					Metrics					Metrics				
		Pr	Re	AUC	F1	AIT	Me	Pr	Re	AUC	F1	AIT	Me	Pr	Re	AUC	F1	AIT	Me
DeeperForensics	C0	0.9965	0.0280	0.8899	0.0545	1.7100	5.67	0.7479	0.8780	**0.8438**	**0.8077**	0.0084	5.10	0.8599	0.9940	**0.9344**	**0.9221**	0.7656	0.71
	C23	0.9968	0.0280	0.8897	0.0545	1.6942	6.18	0.5946	0.8420	0.7133	0.6970	0.0088	5.10	0.8619	0.9920	**0.9345**	**0.9224**	0.7515	0.71
	C40	0.9967	0.0300	0.8905	0.0582	1.6847	**1.94**	0.6625	0.7460	0.7553	0.7018	0.0088	5.06	0.8590	0.9930	0.9323	0.9212	0.7396	0.71
Celeb-DF (v2)	C0	0.9911	0.0300	0.8569	0.0582	0.6343	6.03	0.5051	0.9840	0.4899	0.6676	0.0081	4.89	0.5654	0.1732	0.4991	0.2651	0.3028	**0.70**
	C23	0.9898	0.0280	0.8565	0.0545	0.6342	4.74	0.5000	0.9960	0.4934	0.6658	0.0081	4.89	0.5654	0.1732	0.4989	0.2651	0.3098	0.71
	C40	0.9870	0.0280	0.8599	0.0545	0.6307	4.75	0.5098	0.9840	0.5129	0.6717	0.0085	4.85	0.5682	0.1752	0.4980	0.2678	0.3059	0.71
LAV-DF	C0	0.9989	0.9861	**0.9993**	**0.9924**	**0.1382**	3.50	0.5035	0.9960	0.5218	0.6689	0.0080	4.81	0.5400	0.1890	0.5047	0.2800	**0.1475**	0.71
	C23	0.9989	0.9861	**0.9993**	**0.9924**	0.1401	3.43	0.5036	0.9880	0.5059	0.6671	0.0077	4.81	0.5455	0.1800	0.5038	0.2707	0.1537	0.72
	C40	0.9988	0.8299	0.9620	0.9065	0.1394	2.68	0.5010	0.9980	0.5010	0.6671	**0.0074**	4.77	0.5467	0.1990	0.5060	0.2918	0.1523	0.71
DFD	C0	0.9932	0.0320	0.8406	0.0620	6.9267	3.24	0.5062	0.9760	0.5503	0.6667	0.0094	3.40	0.5057	0.4420	0.4804	0.4717	1.2460	**0.70**
	C23	0.9926	0.0320	0.8414	0.0620	6.9272	2.69	0.5051	0.9898	0.5546	0.6685	0.0092	3.39	0.5036	0.4220	0.4798	0.4592	1.2291	0.71
	C40	0.9901	0.0240	0.8451	0.0469	6.5375	2.90	0.5084	0.9680	0.5063	0.6667	0.0089	**3.28**	0.5074	0.4820	0.4778	0.4944	1.1734	0.71
DFW	C0	0.9961	0.1491	0.9599	0.2594	2.0963	4.35	1.0000	1.0000	-	1.0000	0.0093	4.16	-	-	-	-	0.4552	0.73
	C23	0.9960	0.1491	0.9599	0.2594	2.1120	3.33	1.0000	1.0000	-	1.0000	0.0094	4.15	-	-	-	-	0.4657	0.73
	C40	0.9958	0.1491	0.9600	0.2593	2.0904	2.11	1.0000	1.0000	-	1.0000	0.0089	4.02	-	-	-	-	0.4558	0.72

4.2 Comparative Analysis of the Model Performance and Accuracy

During the experiments, four metrics including Precision (Pr), Recall (Re), F1-score (F1), and Area Under the ROC Curve (AUC) were calculated and recorded regarding the model performance and accuracy in each test case. Among those four metrics, two metrics (F1-score and AUC) were selected as primary performance metrics in this analysis. These two metrics were chosen because they effectively capture both discrimination power and class balance sensitivity of deepfake detection models, which are crucial for such deepfake detection tasks, significantly when diverse conditions such as class distribution, compression, and quality of the input media are involved, which can affect the model's performance and accuracy.

Moreover, as the harmonic mean of precision and recall, the F1-score is a perfectly balanced combined representation of both metrics. It shows the model's capability to minimize false positives and false negatives, which is a crucial factor in deepfake detection since misclassifying a deepfake as real can be dangerous, while misclassifying a real as deepfake can be costly. Furthermore, AUC measures the model's discrimination ability, which means how well the model separates classes across all thresholds. Therefore, AUC is threshold independent, which provides a broader view of the model's performance.

In addition, precision and recall are excluded from this analysis because, if considered separately in isolation, they can be misleading. For instance, a model with high precision but low recall may miss most fakes, which is undesirable in detecting deepfake misinformation.

Based on the presented results in Table 4.1, heat maps for F1-score and AUC of all models across datasets and compression levels were generated as shown in Fig. 4.1 and Fig. 4.2, respectively. Moreover, a scatter plot between threshold

independent AUC and threshold dependent F1-score is generated as shown in Fig. 4.3. In the scatter plot on Fig. 4.3, the calibrator line is the diagonal straight black line. The points on the line or at least near the line represents the better models as those has a well balanced AUC and F1 due to better threshold calibration (i.e., capacity of the model is matching with the operating point of the model), while points far below the diagonal shows models with good AUC but poorly calibrated threshold.

4.2.1 Analysis on BA-TFD+ Detector

On the AUC heat map on Fig. 4.2, BA-TFD+ is almost always bright green and yellow, even for the longest and highest resolution DFD dataset, and continuously varies between approximately 0.84 and 1.00, indicating that the model's pipeline has learned features that separate real from fake frames. However, in the F1 map on Fig. 4.1, the same high performing cell on AUC Fig. 4.2 are near dark blue showing lower F1 values which varies between approximately 0.05 to 0.06 on DeeperForensics, Celeb-DF (v2) and DFD regardless of compression level. Moreover, the DFW dataset only shows a slightly better performance on F1 than the other three datasets.

However, there is a striking exception. LAV-DF dataset shows both an impressive F1-score and AUC value on the BA-TFD+ model. In both heat maps, 4.1 and 4.2, it highlighted with brighter yellow on C0 and C23 LAV-DF datasets and only a slight decrease on the C40 dataset. On the scatter plot in Fig. 4.3, in most cases, points for BA-TFD+ lie far below the diagonal line, representing the high AUC and lower F1-score. This is because the model BA-TFD+ is specifically trained on a subset of the LAV-DF dataset and specifically fine-tuned to detect data and deepfakes similar to LAV-DF, with a threshold that perfectly matches the model's capacity. Furthermore, since the DFW dataset had random deepfake videos collected from

the Internet, it might contain videos similar to the LAV-DF dataset. Hence, showed slightly better AUC and F1 than the other three datasets.

Additionally, the model BA-TFD+ is compression independent in almost every test case except moving from C0 to C40 on the LAV-DF dataset, which showed a slight decrease in performance regarding both AUC and F1, as shown in Fig. 4.2 and Fig. 4.1 respectively. However, even in the LAV-DF dataset, there is no effect on the C23 compression level on the BA-TFD+ model.

Therefore, it can be concluded that BA-TFD+ can be a top-tier detector as it has a vast capacity to distinguish between real and fake. Still, due to severe threshold bias, it performed worst in open-domain datasets. Furthermore, it had only a minor negative impact on performance for highly compressed data on the LAV-DF dataset and no effect elsewhere. In addition, if a small validation set from each dataset is used to calibrate the score threshold rather than the default threshold already calibrated for the LAV-DF dataset, BA-TFD+ might score higher F1-score elsewhere.

4.2.2 Analysis on Convolutional Cross Efficient ViT Detector

Every point in the scatter plot on Fig. 4.3 either sits on or near the diagonal calibrator or slightly above it, meaning the ViT's default 0.5 threshold extracts as many features as the ROC curve promises. In the heat maps, ViT shows a good F1-score on almost all cases with lime green cells and light green sections on the DeeperForensics C0 dataset in Fig. 4.1. However, the AUC for those test cases was merely low with light to dark blue cells of Fig. 4.2, while DeeperForensics, especially with the C0 dataset, shows a slightly better AUC. This means the detector performed well at the default threshold but struggled at class separation in most cases, probably due to bias or over-fitting to particular dataset(s).

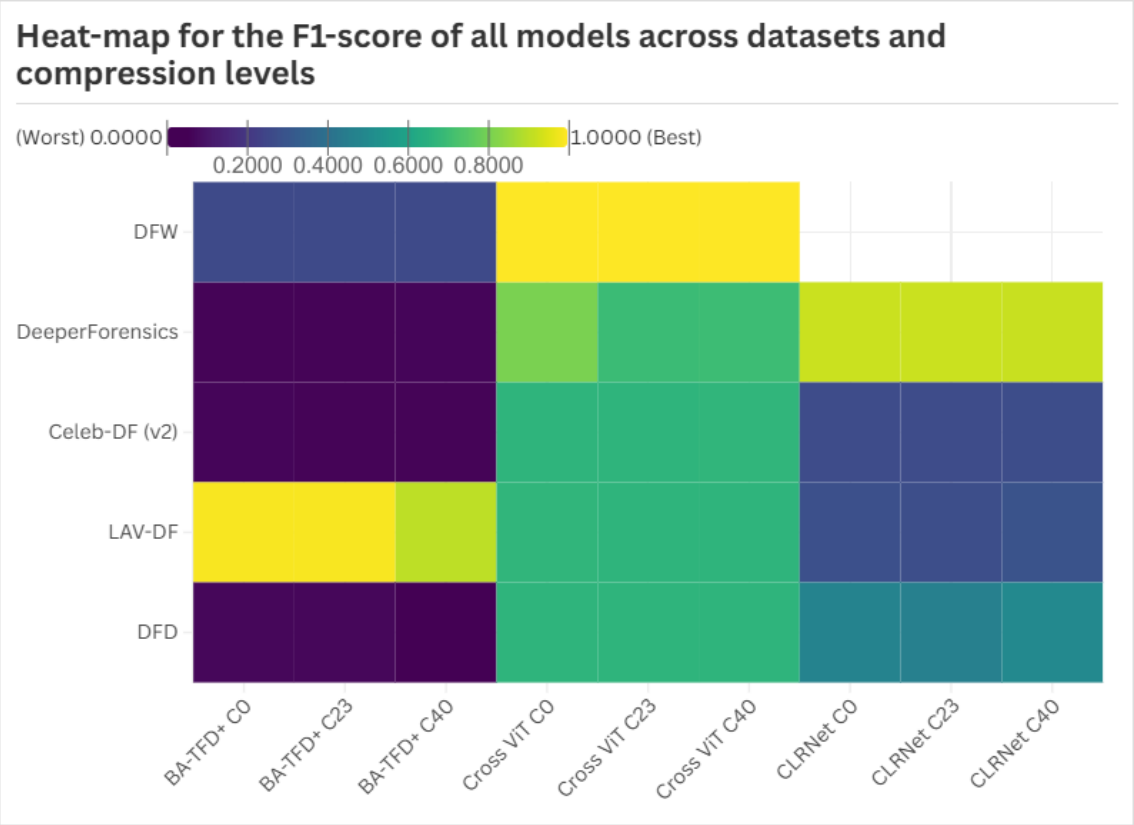

Figure 4.1: Heat-map for the F1-score of all models across datasets and compression levels.

The Cross Efficient ViT model's performance is mid-range as F1 floats approximately in a narrow 0.67-0.81 band for almost all cases except DFW, which climbs up to 1.0000, but no value returned for AUC on the DFW dataset. The reason behind the perfect F1 on the DFW dataset is that, as per the model's pipeline, the model predicted only one class (i.e., all fakes in this case), and the ground truth matches, as DFW had only fake videos (i.e., imbalanced classes on the dataset). Moreover, all predictions were correct as precision and recall also have a perfect value of 1.0000 as shown in table 4.1. Furthermore, AUC requires both positive and negative samples to compute true/false positive/negative rates across thresholds. Therefore, since only one class is presented on the DFW dataset and due to the model's perfect pre-

diction on precision and recall (i.e., the model predicts correctly, but only one class is predicted), the ROC curve cannot be constructed as it returned a "nan" value during the experiments.

Although it had a few good, moderate, and reasonable values for both F1-score and AUC everywhere, it never reached the perfect scores of both F1 and AUC of BA-TFD+ presented under the LAV-DF dataset. Moreover, compared to the original results from [34], which have an AUC above 0.9200 and an F1 above 0.8400 across all test cases, this study showed a slight decrease in both metrics. However, the chosen datasets in the original work and this work are different, which could be the reason for the decline in accuracy metrics. In addition, the compression level of the dataset had a low to no effect on the detector's performance and accuracy, except for the DeeperForensics dataset moving from C0 to C40.

Therefore, it can be concluded that Convolutional Cross Efficient ViT is the safest choice when no validation dataset is available for tuning, as it has a moderate F1-score everywhere, excluding the outlier of the DFW dataset, and calibration is robust. Furthermore, it had only a minor negative impact on performance for highly compressed data on the DeeperForensics dataset and no effect elsewhere.

4.2.3 Analysis on CLRNet Detector

On the scatter plot in Fig. 4.3, CLRNet's points split into three clearly separate groups. On the DeeperForensics dataset under every compression level, they sit at approximately AUC = 0.93, F1 = 0.92, right on the diagonal line of the scatter plot. The detector is both powerful and perfectly calibrated when the input features of the given datasets match those seen during the training. This can also be seen clearly on the heat maps of F1 4.1 and AUC 4.2. On every other dataset, it falls to approximately AUC = 0.50, F1 = 0.28, except DFD (approximately AUC = 0.48, F1 = 0.46). Furthermore, unlike the BA-TFD+ detector, the F1 decline mirrors

the AUC decline on the CLRNet detector as shown in both heat maps 4.1 and 4.2. Therefore, it is not a threshold error, and the convolutional encoder simply fails to generalize to unseen artifacts. Hence, it can not be fixed with threshold tweaking.

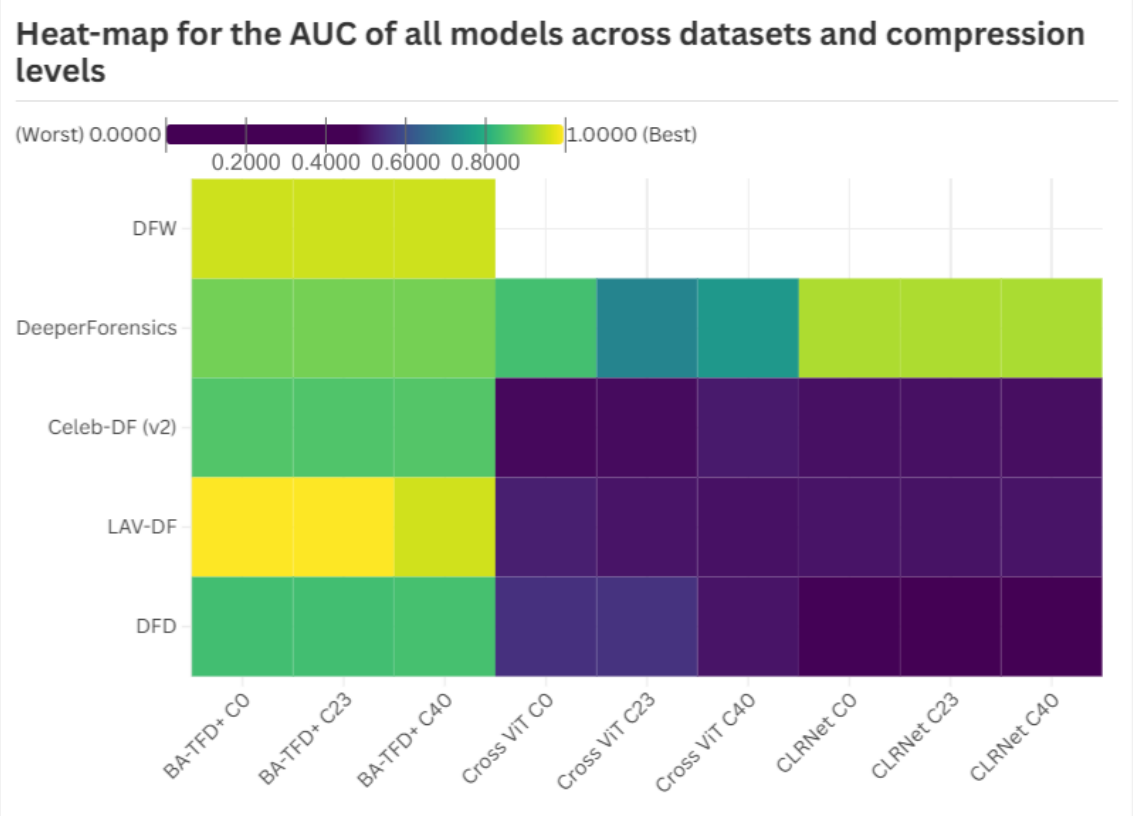

Figure 4.2: Heat-map for the AUC of all models across datasets and compression levels.

Although it was claimed that the CLRNet detector trained on a subset of the DFD dataset, among other datasets, and had a better AUC value by the authors [53], it still showed no impressive results in these experiments, as expected. The most probable reason might be that the publicly available pre-trained model [54], which was used in these test cases, might not have been trained with a subset of the DFD dataset. Moreover, the provided dataset's compression level did not considerably impact the model's performance and accuracy in either test case.

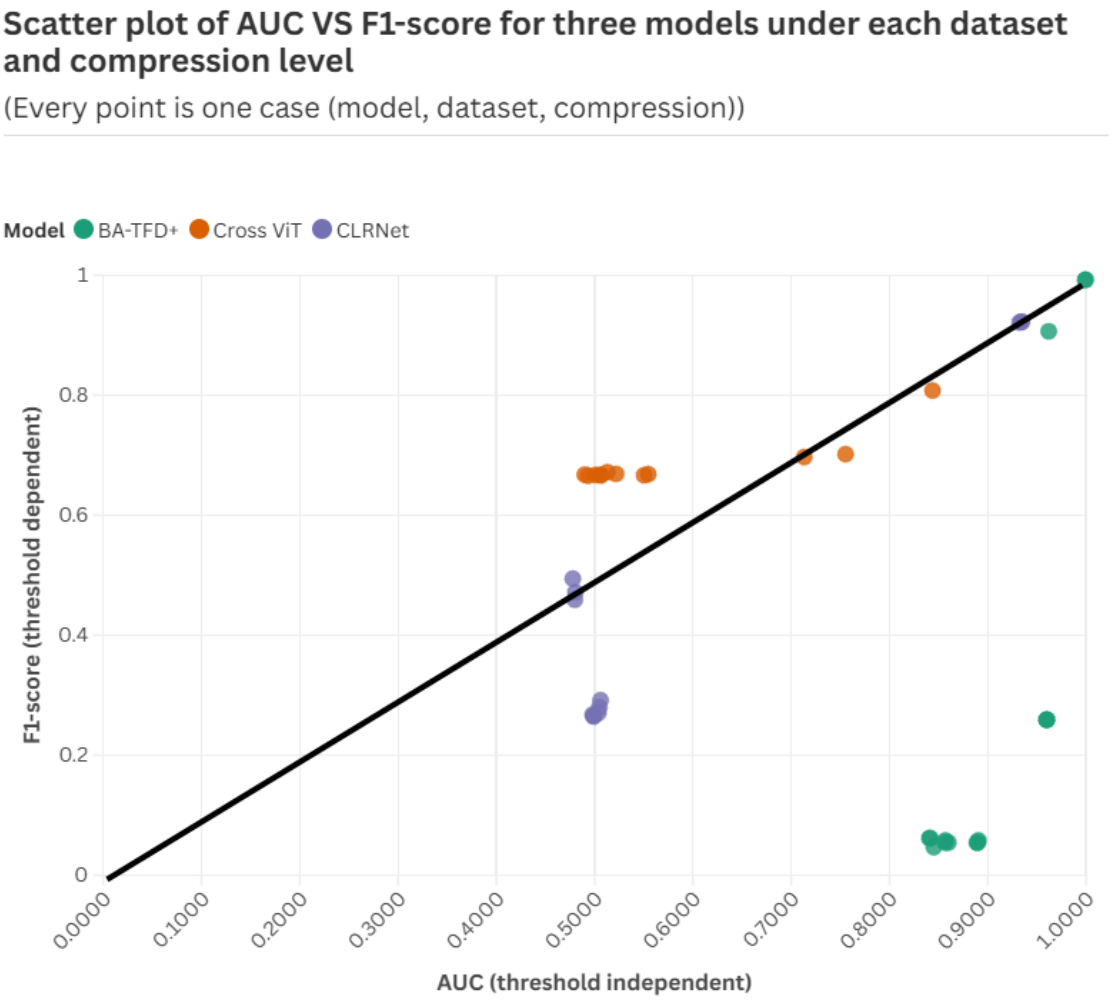

Figure 4.3: Scatter plot of AUC VS F1-score for three models under each dataset and compression level. Each point represents one case (model, dataset, compression). The diagonal line represents the perfect calibration line.

Additionally, for the DFW dataset, it returned "nan" for all cases. As mentioned above in the Cross ViT section, it is due to the fact that only one class is presented (i.e., only fakes). Moreover, both precision and recall are undefined as the CLRNet model predicts only the wrong class apart from the ground truth value (i.e., if CLRNet predicted all real (0) and the ground truth is all fake (1), then no true

positives, no positive predictions). Hence, F1 is undefined as it is based on precision and recall. Furthermore, AUC also fails because there's only one class in the ground truth, and no predicted positives are presented.

However, unlike the CLRNet detector, Cross Efficient ViT predicted perfect scores for precision, recall, and F1-score. It might be due to the fact that ViT predicted all fake (1) and the ground truth is also all fake (1), leading to TP = all, FP = 0, TN = FN = 0, as no real video was presented. Hence, both precision and recall (see section 3.2.1 for definitions) become 1, leading to AUC to 1 as well.

Conclusively, CLRNet is robust in datasets resembling DeeperForensics, although robustness was also expected for the DFD dataset. Moreover, it must be fine-tuned in open-domain conditions, and calibration alone might not fix the issue. In addition, compression has no significant impact on the model's accuracy and performance based on the considered metrics.

4.3 Generalization Ability

The detector's practical values hinge on how well it performs on out-of-domain and open-domain datasets, which partially or entirely differ from the data it was trained on. Therefore, in this study, two-folded shifts of datasets were used. One is the cross-dataset shift, which uses five different datasets with different generation methods and resolutions. The other shift is the bit-rate shift, which uses three H.264 quality tiers (C0, C23, and C40) that systematically and progressively remove some visual features when compressing.

The behavior between those two factors is clearly visible in the F1 heat map in Fig. 4.1, AUC heat map in Fig. 4.2, and scatter plot of AUC VS F1 in Fig. 4.3. As per the observed results in table 4.1 and the above figures, Convolutional Cross Efficient ViT detector is a clear generalist. In the F1 heat map on Fig. 4.1, it always stays in the green to yellow band for every row, and the scatter plot shows two

stable spots near approximately (0.49, 0.67) and (0.84, 0.81) points. Although AUC drops under the domain shift, the operating threshold tracks that fall, so the F1 remains the same. Furthermore, generalization is not only about how much a model knows (i.e., measured in AUC), but also how wisely it converts that knowledge into decisions (i.e., measured in F1). Since Cross ViT has both high F1 and moderate AUC across all varied test cases, it can be considered a well-generalized deepfake detector.

Moreover, CLRNet detector has slightly lower F1 score elsewhere except DeeperForensics test cases which exhibit an excellent decision making with higher F1 as well as higher AUC which can be seen clearly on both scatter plot on Fig. 4.3 with violet dots on top-right along the diagonal line as well as in two heat maps. Moreover, as mentioned in subsection 4.2.3, the fall in F1 mirrors the fall in AUC on CLRNet, so no threshold adjustment would fix it, which means that the feature extractor itself is over-fitting to high-resolution, well-fitted training data. Therefore, it will not be well-fit for the generalization.

On the other hand, BA-TFD+ exhibits a complete opposite trend than both Cross ViT and CLRNet models, as in most cases BA-TFD+ has a higher AUC but a very poor F1-score, except for the LAV-DF dataset, which initially was trained on. AUC stays mid to higher range as an average of approximately 0.89 even on open-domain sets, yet the corresponding F1 falls as low as approximately 0.07, which can be clearly seen on the AUC heat map in Fig. 4.2 with dark purple color, which is close to 0.0000 from the map's legend. Moreover, on the scatter plot in Fig. 4.3, this pattern is represented by the green dots close to the bottom-right corner, which have F1 close to 0, way below the diagonal calibrator line. The only perfect scores it returned are the LAV-DF dataset that perfectly lay on the calibrator line in the top-right corner of the scatter plot. Therefore, as mentioned in subsection 4.2.1, it has a great potential to be a generalized model, only if the threshold is re-learned

with a small validation split of each dataset. Therefore, as of now, BA-TFD+ is not a suitable candidate for generalization purposes.

In addition, because every dataset/compression triplet creates three dots of identical color in the scatter plot, compression effects are easy to tackle. Dots would spread horizontally if capacity changes (AUC axis) or vertically if calibration shifted (F1 axis). However, as shown in the scatter plot in Fig. 4.3, dots overlap almost perfectly in every case except a few exceptions. That complete horizontal and vertical collapse means all three detectors under each dataset are compression-invariant. Heat maps represent the same scenario. Only BA-TFD+ under LAV-DF dataset from C0 to C40 and Cross ViT under DeeperForensics from C0 to C40 are exceptions as seen in both heat maps in Fig. 4.2 and Fig. 4.1.

Conclusively, Convolutional Cross Efficient ViT shows a great generalization ability, while BA-TFD+ has the potential to be a generalized detector under certain conditions. Although authors [53] claimed that the CLRNet detector is a more generalized model as per their extensive experiments, this study showed relatively low results with few exceptions and is unsuitable for generalization according to the received results and selected metrics.

4.4 Computational Efficiency and Limitations

Evaluating the computational efficiency of deepfake detection models is crucial for assessing their real-world deployments, where resource constraints, response time, and scalability become critical factors.

Therefore, this section presents a comparative analysis of the selected deepfake detection model's computational efficiency based on two factors considered: Average Inference Time (AIT) measured in seconds per video, frame, or batch, and CPU Memory Usage (Me) measured in GB. Those two metrics were recorded for the three deepfake detection models across five datasets and under three compression

levels as shown in Table 4.1.

4.4.1 Average Inference Time

As per the observed results, the Cross Efficient ViT model is the fastest model in every dataset and compression level, while the BA-TFD+ is the slowest. On the other hand, the CLRNet model shows a balanced approach in terms of AIT.

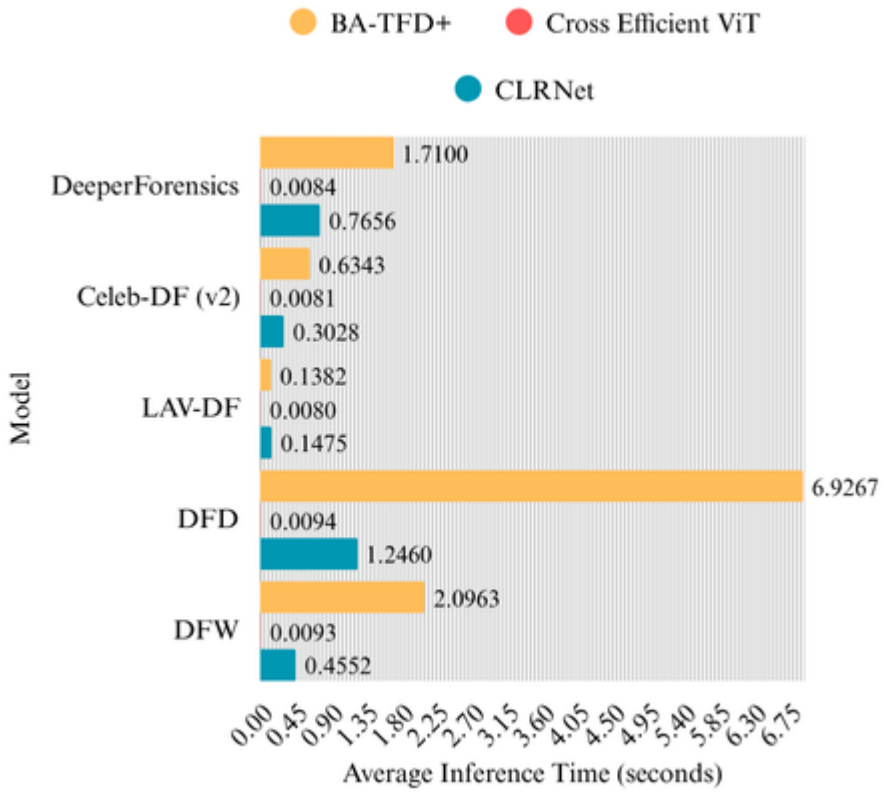

Figure 4.4: Average inference time variation across models and datasets for C0 compression level.

Figure 4.4, 4.5, and 4.6 show the row bar charts on how the AIT varies for each dataset and model across different compression levels. As per those row bar charts, there is no significant reduction of AIT when moving from C0 to C23 in each case. Moreover, the firm compression has no considerable impact on significantly

reducing AIT, except for the BA-TFD+ and CLRNet models on the DFD dataset, which have approximately a 6% reduction of AIT moving from C0 to C40. For instance, if considering the dataset-level behaviour, DeeperForensics over Cross ViT has a slight increase in AIT moving from C0 to C40 (0.0084 s to 0.0088 s) while approximately a 3% decrease of AIT in CLRNet and a 1.5% decrease of AIT in BA-TFD+. Furthermore, both Cross ViT and CLRNet show a slight increase in AIT on the Celeb-DF (v2) dataset.

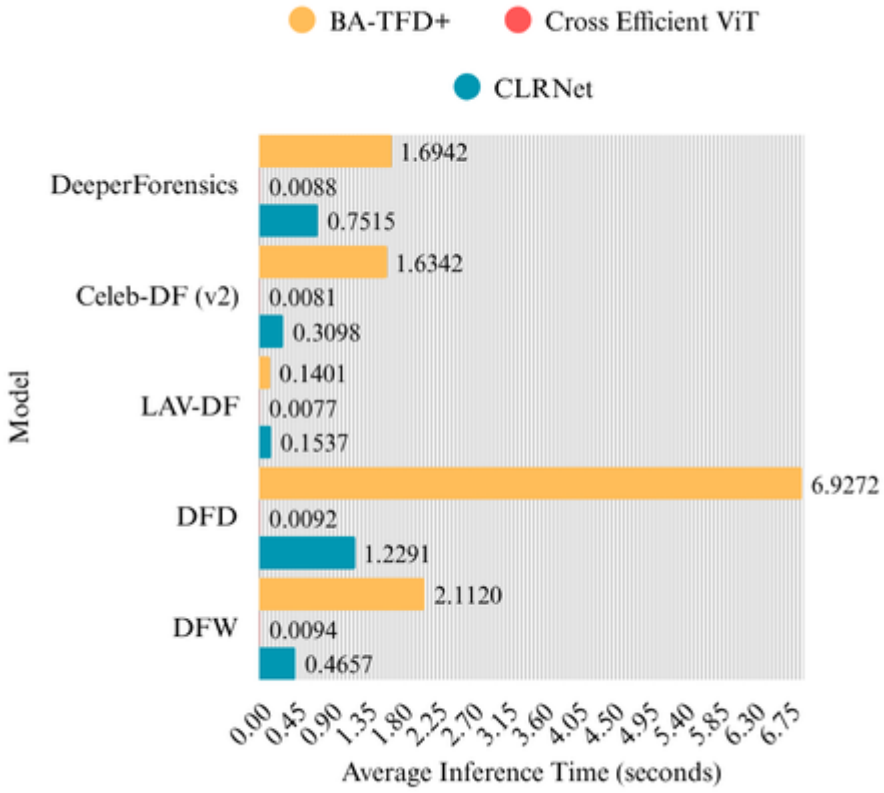

Figure 4.5: Average inference time variation across models and datasets for C23 compression level.

Moreover, table 4.2 presents the mean values of AIT in seconds over five datasets in each compression level, along with the difference percentage from C0 to C40. As per the table 4.2, BA-TFD+ recorded the highest deduction of the mean of the AIT

moving from C0 to C40 compression of 3.68%, followed by the CLRNet with 3.09%. Cross Efficient ViT achieved the lowest decrease difference of 1.16%. Based on these results, it can be concluded that the BA-TFD+ model is the best-performing model for highly compressed data (i.e., when the compression level increases, the AIT has a significant decrease).

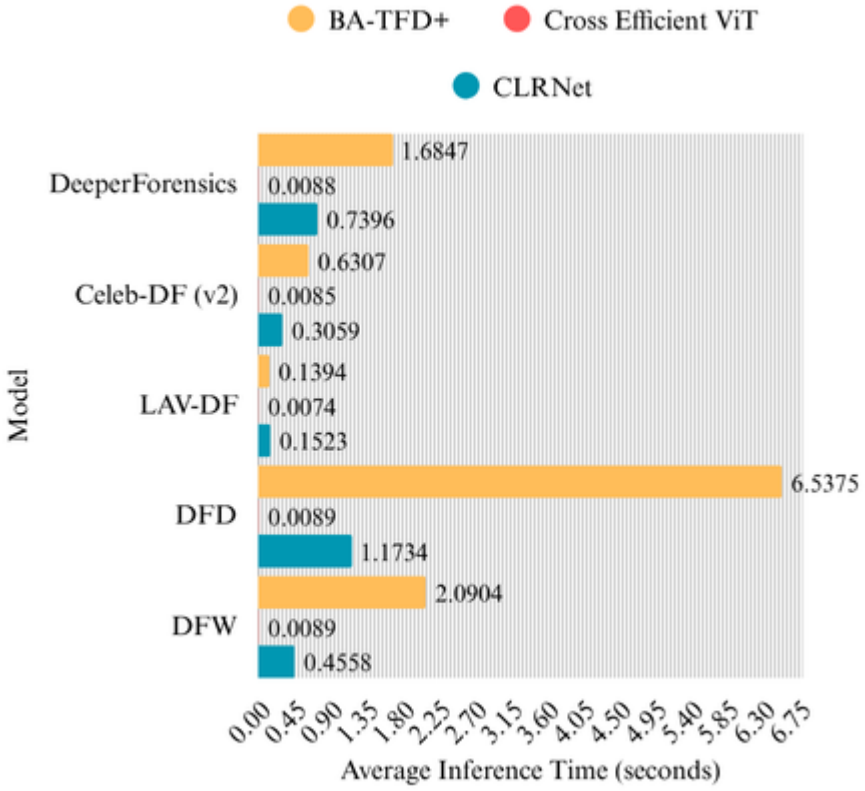

Figure 4.6: Average inference time variation across models and datasets for C40 compression level.

Table 4.2: Mean Average Inference Time (in seconds) over five datasets in each compression level and difference percentage from C0 to C40.

Model	**Mean AIT over 5 datasets (C0 / C23 / C40)**	**Difference percentage from C0 to C40**
Cross Efficient ViT	0.0086 / 0.0086 / 0.0085	-1.16%
CLRNet	0.5834 / 0.5820 / 0.5654	-3.09%
BA-TFD+	2.3011 / 2.3015 / 2.2165	-3.68%

In addition, in each case, BA-TFD+ has a higher AIT than both Cross Efficient ViT and the CLRNet model. It is due to the fact that the BA-TFD+ model's workflow is based on video processing, while the evaluation is ongoing by accepting video files as input. Therefore, all the preprocessing, including detecting faces, cropping, and resizing detected frames of the video inputs, is done by the model itself while the evaluation is conducted. On the other hand, for both Cross Efficient ViT and CLRNet models, the preprocessing had to be done beforehand and separately from the primary evaluation workflow of the models, as those models accept preprocessed frames in ".png" format as input. The frames from videos were cropped to capture faces and resized before being fed into those two models. Therefore, the AIT of those two models was significantly reduced.

Furthermore, the DFD dataset, especially with BA-TFD+, showed a heavy outlier. The DFD dataset has some of the longest videos and is heavy in file size. Due to this reason, DFD consumed the most inference time in each case, among all other datasets.

Conclusively, when considering AIT, Cross Efficient ViT outperformed both CLRNet and BA-TFD+ in each test case. However, BA-TFD+ is the most efficient model among others for highly compressed data, as AIT showed a significant reduction difference in BA-TFD+ cases. Moreover, the compression level C23 showed low to no AIT improvement in most cases. In addition, a heavy dataset DFD showed a heavy outlier, especially with the BA-TFD+ model.

4.4.2 Memory Usage

As per the observed results of the experiments, the CLRNet was the most memory-efficient model in every test case, while BA-TFD+ consumed the most memory in most test cases, except a few.

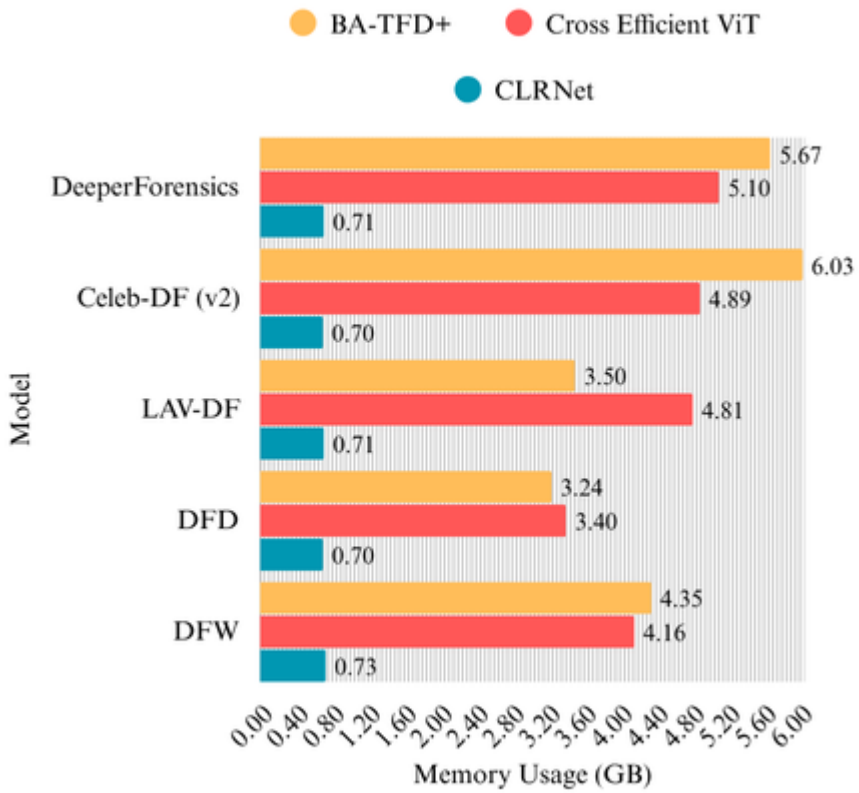

Figure 4.7: Compression level-wise memory usage variation across models and datasets for C0 compression level.

As per the row bar chart presented in Figure 4.7, 4.8, and 4.9 the CLRNet not only the most efficient but also it showed no sensitivity to the compression level as in each of 15 test cases, the CLRNet's memory consumption is ranged between 0.70-0.73 GB with only 0.03 GB difference. Moreover, Cross Efficient ViT followed the same pattern as CLRNet and showed only a minor sensitivity to compression level change, which can be negligible. On the other hand, memory consumption of BA-TFD+ ranged between 1.94 and 6.18 GB, with a 4.24 GB difference. Unlike test cases for AIT, for memory usage, in most cases, except a few, Cross Efficient ViT+ consumed the most memory of the system. In most test cases, there is no significant impact of C23 compression level on memory usage reduction. However,

in some cases, there is a considerable impact when move from C0 to C40 specially BA-TFD+ model on DeeperForensics dataset showed memory usage decrease from 5.67 GB to 1.94 GB of 3.73 GB which is approximately 65% decrease in memory usage as well as BA-TFD+ model on DFW dataset showed reduced memory usage from 4.35 GB to 2.11 GB when moving from C0 to C40 which is approximately 51% decrease in memory usage.

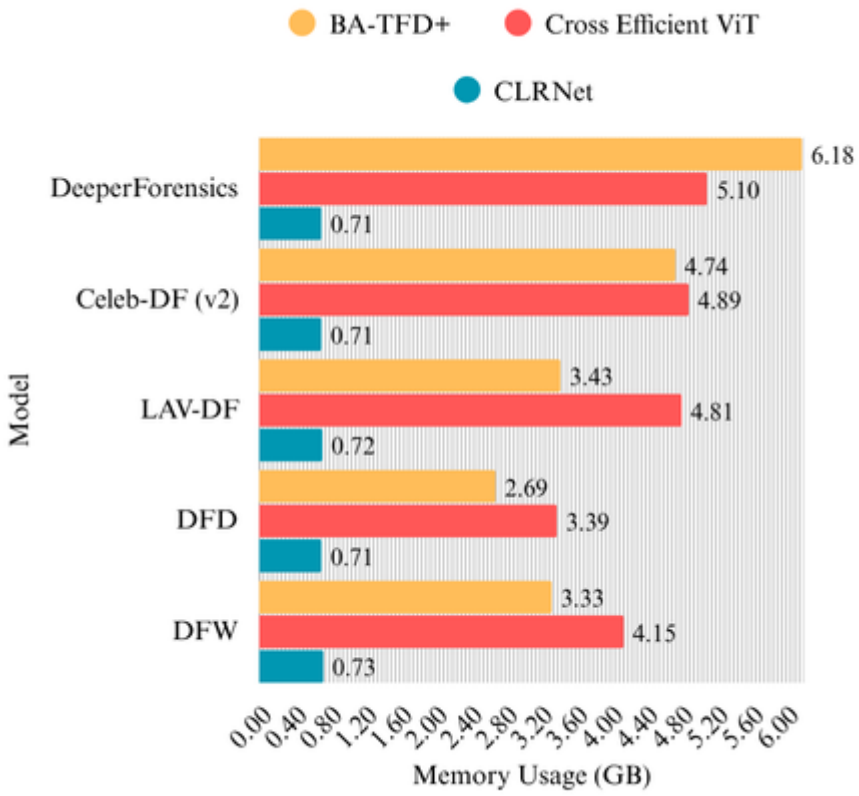

Figure 4.8: Compression level-wise memory usage variation across models and datasets for C23 compression level.

In addition, unlike AIT, BA-TFD+ outperformed the Cross Efficient ViT in memory consumption. Under every compression level except for a few cases (i.e., DeeperForensics under C0 and C40, Celeb-DF (v2) under C0, and DFW under C0), Cross Efficient ViT consumed more memory than the BA-TFD+ (see Fig. 4.7, 4.8,

and 4.9).

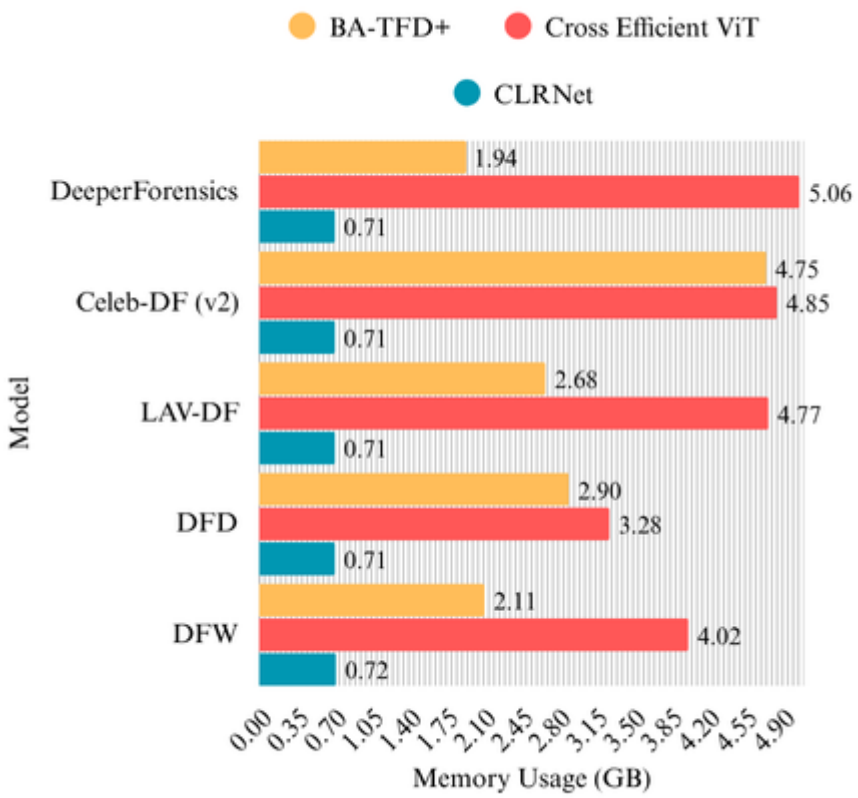

Figure 4.9: Compression level-wise memory usage variation across models and datasets for C40 compression level.

As per the mean memory usage presented over five datasets in each compression level in Table 4.3, the fact presented using Fig. 4.7, 4.8, and 4.9 can be further confirmed. CLRNet showed no sensitivity for compression, while Cross Efficient ViT had only a minor effect, with only 1.57% variation. On the other hand, BA-TFD+ had a significant impact on compression level change.

Conclusively, CLRNet is the most memory-efficient model over each dataset under every compression level, while Cross Efficient ViT is the worst under this metric. Furthermore, BA-TFD+ is the most elastic model with the most sensitivity for compression level change. It can be heavier or lighter based on the bit rate

Table 4.3: Mean Memory Usage (in GB) over five datasets in each compression level and difference percentage from C0 to C40.

Model	**Mean Memory Usage over 5 datasets (C0 / C23 / C40)**	**Difference percentage from C0 to C40**
CLRNet	0.71 / 0.71 / 0.71	0.0%
Cross Efficient ViT	4.47 / 4.47 / 4.40	-1.57%
BA-TFD+	4.56 / 4.07 / 2.88	-36.84%

and resolution of the provided data. By providing heavily compressed data for the BA-TFD+ model, the computational cost can be significantly reduced.

4.5 Suggestions and Strategies for Combating Deepfake Misinformation

The quantitative study discussed in depth in this Chapter 4 confirms that no single detector among the selected detectors in the deepfake detection domain is both infallible and universally efficient. Cross-domain error rates remain high, thresholds drastically drift, and hardware requirements vary on a large scale based on the architecture and pipeline of the concerning detector.

Therefore, combating deepfake misinformation demands not only technical best practices, but a blended approach with technical best practices and policy interventions. This chapter aims to describe such strategies.

This chapter focuses on answering the following research question.

RQ3: What actionable deployment strategies and policy recommendations can be derived from model performance insights to enhance detection and curb deepfake-based misinformation campaigns?

4.5.1 Recommendations Regarding Deepfake Model Deployment

From a technical point of view, deepfake model deployment requires multiple aspects of strategies based on the model's accuracy, performance, and computing hardware efficiency. The study showed the strengths and limitations of each such detector. Cross Efficient ViT excels at generalization tasks, BA-TFD+ is good at domain-specific detection (i.e., for previously seen data), and CLRNet has bandwidth constraints, for instance. From the hardware requirements point of view, CLRNet was the most efficient, Cross ViT had moderate consumption, while BA-TFD+ was the most power-hungry detector. Therefore, instead of using one detector for deepfake detection tasks, it would be more efficient to run two models in parallel, which would have a perfect balance of accuracy of detection and computational efficiency. Furthermore, if the pipelines of the models allow it, it would make more accurate predictions when hybrid detectors are used as well.

Moreover, for the BA-TFD+ model, threshold error was the principal failure, not the capacity. It had the capacity to almost perfectly distinguish between real and fake across all datasets, but due to the severe threshold error, it failed to achieve a higher F1-score on open-domain sets. Therefore, it would be better to calibrate the threshold using a small validation split before the testing phase.

Another observed result is that none of the detectors loses accuracy significantly at the C40 compression level. Therefore, live pipelines of the detectors can downsample or re-encode incoming video to cut the memory usage for a considerable amount (i.e., BA-TFD+ model under DeeperForensics dataset moving from C0 to C40 memory usage reduced by approximately 65% without reducing the accuracy scores significantly) and save the computational power without negatively affecting the detection accuracy.

In addition, lightweight detectors such as CLRNet can be used on resource-

constrained edge devices to flag suspicious frames and forward the flagged segments to heavy models hosted on clouds such as BA-TFD+ (after the identified threshold tuning) for further detection. This two-stage deployment strategy allows high throughput while keeping false negatives low, which eventually increases both the accuracy of the detection system and reduces the requirements of high computational power and reduces the latency of detection.

These sorts of deployment and model improvement strategies not only improve the accuracy and performance of the deepfake detection models, but also make them less computationally resource hungry and well-optimized and ready for deployment on computing power-constrained edge devices, such as mobile phones, for instance. Then these approaches allow for the combat against rising misinformation campaigns in a timely manner.

4.5.2 Policy Recommendations

It is better to have strong policy frameworks and regulations to combat misinformation campaigns powered by deepfakes. Especially in social media platforms, it will not only demotivate individuals who are about to share synthetic media claimed as real media, which can have a considerable negative impact on society, individually, and financially, but also enable authorities to act legally to protect the privacy of those who are affected.

Having proper legislation to declare AI-generated content on social media at the time of posting enables synthetic media transparency. For instance, Meta has a voluntary requirement for "AI label" when posting content, especially images, on their platforms, including Instagram, Facebook, and Thread [108]. The EU made a bold move to counter the rising threat of AI and also deepfakes with the newly introduced AI Act [32]. Labeling AI-generated content by the generators themselves is mandatory to distinguish human-generated and machine-generated content, which

improves the transparency of the content posted on digital media [109]. Moreover, it is obligatory to include cryptographically verifiable signatures as a watermark for the AI-generated content by the AI-generated content providers themselves [109]. Furthermore, there are significant fines for non-compliance. However, these mandates are currently rolling in and will fully come into effect by May 2026 [109]. Apart from fines for the generating platforms themselves, it is also better to have civil penalties for platforms and fines for platforms that fail to remove malicious deepfakes within a "trusted-flagged" notice window of 24 hours, for instance. It will significantly reduce the damage that such deepfakes could do. Moreover, making it mandatory to adopt standards such as C2PA standards [110] or similar provenance standards for news-wire, stock photos, or any sort of digital media worldwide will guarantee the authenticity of digital media content.

Furthermore, make it a requirement or a popular nice-to-have feature to have embedded real-time, lightweight deepfake detection APIs and fact-checkers, especially on Content Management System (CMS) pipelines, to flag suspected materials to editors of newsrooms, which will have a positive impact to avoid sharing false information by reputed media networks worldwide. In addition, educating the general public to detect deepfakes might be one of the effective ways to combat misinformation campaigns. Therefore, it is worth investing in educating citizens through modules such as media forensics in the school curriculum to spot fake media and facts. For instance, Finland has integrated media education since 2013 into the national school curriculum and trains the students to spot fake news and media [111], and this approach should be adopted worldwide.

Additionally, it is nice to prioritize grants in the research and development field to seek cross-domain, low-resource detectors and to introduce robust and tamper-resistant methods, such as watermarking, that survive common modifications. Furthermore, united gatherings such as the Paris Call for Cybersecurity [112] should be

organized more often to raise awareness of cyber threats and improve the security of cyberspace. Moreover, all international regulatory bodies should harmonise and introduce a set of common standards to avoid jurisdictional loopholes to protect the privacy of all.

Conclusively, combating deepfakes requires not only a technical solution as discussed, but also a set of robust and solid legislation and standards as obligatory requirements.

4.6 Discussion

After the findings in section 3 and suggested strategies and deployment methods in section 4.5, this chapter considers them all together.

One of the most striking discoveries is the relation between discrimination capacity (AUC) and realized accuracy (F1). The BA-TFD+ hovers near 0.90 on most datasets as a best performer on that metric, yet its F1-score collapses below 0.06 because the default threshold is misaligned with class priors in those domains. Conversely, Cross ViT only has moderate AUC on the hardest dataset but converts almost all latent capacity into F1 owing to robust calibration. Moreover, the two models, except the BA-TFD+, were failing when they were fed only one class of data (i.e., fake-only data from the DFW dataset), demonstrating the importance of having a different pipeline to handle such cases as well.

A single metric, such as AUC, masks half the truth, and so does the thresholded figure, such as F1. Effective deployment demands simultaneous optimization of both axes, or at least a calibration layer that can be modified as distribution shifts. The unified mismatch AUC-F1 proposed in these results therefore deserves a broader adoption as a continuous monitoring metric.

Remarkably, none of the three networks showed more than about 1% variation in either AUC or F1 across the H.264 quality sweep. Yet, some test cases showed a

relatively low requirement for computing power for highly compressed data. Therefore, it is possible to use down-sampled data to the detectors without a significant accuracy loss when implementing and testing such detectors, and thus, such models can be deployed into edge devices or a two-stage edge/cloud pipeline as proposed in section 4.5.

When considering the domain shift, none of the detectors satisfy the generalization ability at a perfect or at least a reasonable level that can have the title "universal model". Cross ViT generalizes best among the other two models, but still loses more than half of its separability on Celeb-DF (v2), for instance. CLRNet excels only on a few instances, as does BA-TFD+, which only excels on the LAV-DF dataset, as it was initially trained using a subset of it. Therefore, these models need calibration and fine-tuning to adapt to the diverse conditions, so they will have more generalization ability.

Latency and memory usage, as the reported metrics regarding computational efficiency, showed that the BA-TFD+ model was the most resource-hungry model, while the other two had low to moderate latency and memory usage. However, those two models also required pre-processed data to make the predictions, rather than the raw video files, which have additional costs and were not recorded in this study. Therefore, based on each considered model's architectural differences and working pipelines, the CLRNet is most suitable as a lightweight model to deploy into edge devices, only if the initial data pre-processing cost is discarded. However, it is also essential to introduce a wide range of metrics to report the carbon footprint and energy consumption throughout the entire life cycle of such detectors.

The policy recommendations introduced in section 4.5 section 4.5.2 were motivated by the technical observations but raise more profound ethical questions. Whoever controls the watermark issuing platforms can use it for malicious purposes, such as watermarking actual content as AI-generated or AI-generated content that can

be flagged as real. If detectors mislabeled specific demographics, it will not counter fake, instead support spread unfair doubts. Therefore, any such rollout requires clear legislation, rules, independent audits, a workable and fair plan, and a means of appeals.

Furthermore, the evolution of synthetic media generation methods is way faster than the academic researchers can be peer-reviewed and propose workable and efficient counter detectors. The solution is not a faster science but providing adaptive infrastructures, data pipelines that ingest novel attacks as they occur, detectors that can learn easily, and watermarking schemes that evolve in the adversarial sector. For instance, signatures alone become obsolete in antivirus software, but behavioural and cloud-aggregated approaches will dominate. More or less, the same applies in the domain of deepfake detection.

To sum up, combating deepfakes is not only about finding a smarter detector. It's about calibration for better performance and accuracy, smart and efficient resource use, concern for ethical aspects, and rapid updates. Combining all these together will lead to a successful effort to combat deepfake-based misinformation campaigns in a broader range.

5 Conclusion and Future Works

5.1 Limitations and Future Works

Despite the variety of datasets, metrics, and hardware requirements explored in this study, it represents a snapshot of a rapidly moving landscape. All five benchmark datasets were released between 2019 and 2023 and focused on face-centric manipulations generated by GAN-based pipelines, and also with unknown methods. Recent diffusion-based forgery tools such as Stable Diffusion generate subtler artifacts and more diverse subject matters, not only faces but also bodies, scenes, and perfect lip syncing patterns, than those captured here. Consequently, the conclusions regarding the generalization ability for the considered models in this study may overstate real-world robustness when confronted with next-generation synthetic media generation methods and tools. An essential next step is constructing a rolling, more solid, all-in-one, and community-maintained dataset that is robust enough to counter next-generation deepfake attacks.

The second constraint arises from compression handling. In this study, standard compression levels of C0, C23, and C40 from H.264 were adopted to compress the collected original datasets. Yet social platforms deploy far more diverse, aggressive, content-adaptive transcoding on various channels such as WhatsApp, Facebook, Instagram, and TikTok, etc. which might apply different sorts of compression and quality conversion methods and tools integrated into those platforms. Therefore,

the compression standard can no longer be limited to the levels considered in this study. Future work should extend the compression axis to match modern codecs and multiple ways of quality-degrading chains, mirroring how user-generated content is repeatedly saved and re-encoded before releasing into the public timeline.

The third constraint is that the temporal context was underutilized. While the BA-TFD+ model considered more of the temporal artifacts of the facial features, the other two detectors operated only on single frames, treating videos as a batch of frames and neglecting the temporal artifacts. However, it is essential not to ignore higher-order temporal artifacts such as eye-blinking synchronization, lip syncing, areas closely related to facial modifications, and normal and abnormal breathing patterns of persons, etc., that can be highly efficient in detecting deepfakes. Choosing a current state-of-the-art, robust, and efficient detector and integrating such a lightweight temporal attention module into such a detector can improve the accuracy of that detector while maintaining the computational efficiency.

The fourth is the computational requirements of the current models. From an operational point of view, energy consumption and carbon footprint should be important factors that were not profiled in this study. Making the detector lightweight, fine-tuned, and optimized, and reducing the latency of the predictions by the models will significantly reduce both energy consumption and carbon footprint, as those metrics are also valuable concerns in today's world. Therefore, rather than only recording inference time and computational memory usage in experiments, a complete life-cycle assessment for training, calibration, and inference at a large scale remains future work.

The fifth constraint identified is that the lack of demographic clarity of collected datasets as the public benchmarked datasets provide incomplete labels for instance age, skin-tone, race, and gender presentation, etc. of subjects although datasets such as DeeperForensics-1.0 contained a diverse set of video data, but not all those sort

of metadata was not included. Moreover, the detectors themselves excluded such metadata during training, validation, and testing. Therefore, collecting a balanced dataset with such a diverse set of metadata and introducing fairness to that data into the next-generation detectors remains future work.

The final constraint is that those concerned models were not designed to detect the latest temper resistance signatures, such as watermarks or C2PA signatures, which can improve the accuracy and efficiency of such models and maximize the latest technology to harness the advantages of such methods. Therefore, exploring this area would be a perfect research frontier.

In summary, while this work offers a solid and multi-metric audit of three selected detectors, the rapid evolution of generative models, codecs, and adversarial tactics ensures that up-to-date best practices will become tomorrow's baseline. As a combination, future works should focus on continuous, solid, and diverse dataset creation, temporal and robust architectures, energy-aware deployments, diverse detectors aware of a diverse range of metadata for fair auditing, and invest in creating more modest models. By addressing these limitations, the research community can hope to stay ahead in the ever-evolving area of deepfake-based misinformation.

5.2 Conclusion

This study was conducted to answer four initially identified research questions regarding combating the rising threat of deepfake-based misinformation campaigns. How effective are the existing models in detecting deepfakes under diverse and varying conditions, how can the simulated evaluation identify the strengths and limitations of those models, what sorts of actionable strategies can be derived from those results of assessments to improve detection and mitigate the deepfake-based misinformation campaigns, and finally how can the insights from model performance inform and introduce robust policies and standards to address deepfake-based mis-

information campaigns were the four research questions. By selecting three contrasting architectures, BA-TFD+, Convolutional Cross Efficient ViT, and CLRNet, across five datasets and three compression statuses, this study produced a multi-dimensional view that answers those research questions.

The experimental evidence confirms progress in raw discrimination power, as at least one model (i.e., BA-TFD+) achieves an F1 above approximately 0.80 on every test case. Equally important, all detectors proved compression invariant status. Yet the study also highlighted systematic vulnerabilities. BA-TFD+ wastes substantial capacity due to mis-calibration, while both BA-TFD+ and CLRNet collapse under domain shift. Even the most robust Cross ViT loses about one-third of its separability on unseen data. Moreover, the two detectors, except BA-TFD+, were failing when fed only one class of data (i.e., fake-only data from the DFW dataset), demonstrating the importance of having a different pipeline to handle such cases as well. These findings pointed out the unavailability of an all-around detector and instead point to the need for ensembles, continuous calibration, and meaningful integration of improved modules into the detectors.

Then in section 4.5, this study translated the technical insights into robust deployment and policy recommendations such as a combined approach of lightweight edge detectors with heavyweight cloud validators, mandate cryptographic watermarking, obligatory AI-label for synthetic media, and cultivate live datasets, etc. By doing so, this study reshaped the idea that combating deepfake detection is not only a technical effort but also a combined effort of legislation and standards as well.

The limitations acknowledged in section 5.1, dataset constraints, absence of diffusion-based forgeries, lack of robustness of detectors, and lack of fairness auditing outline future works for research and development in a broader scope. Moreover, expanding temporal resistance and reasoning, lowering energy consumption and carbon footprints in the deepfake detection life cycle, and aligning detection objectives

with societal values are also up for further investigation.

Conclusively, the main contribution of this study is a holistic evaluation backed up by multi-metric model analysis with operational cost profiling and policy and standards (i.e., governance) considerations. The ultimate takeaways are clear: detection accuracy is necessary but insufficient, and computing resources should be more and more optimized for real-time deployments. Only when capacity, calibration, efficiency, fairness, and policy move forward together can hope to preserve the integrity of digital media in an era of ever-evolving synthetic realities.

References

[1] A. S. George and A. H. George, *View of Deepfakes: The Evolution of Hyper realistic Media Manipulation*, Dec. 2023. [Online]. Available: https://www.puirp.com/index.php/research/article/view/19/15 (visited on 02/18/2025).

[2] H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election", *Journal of Economic Perspectives*, vol. 31, no. 2, pp. 211–36, May 2017. DOI: 10.1257/jep.31.2.211.

[3] E. Blancaflor, J. I. Garcia, F. D. Magno, and M. J. Vilar, "Deepfake blackmailing on the rise: The burgeoning posterity of revenge pornography in the philippines", in *Proceedings of the 2024 9th International Conference on Intelligent Information Technology*, ser. ICIIT '24, Ho Chi Minh City, Vietnam: Association for Computing Machinery, 2024, pp. 295–301, ISBN: 9798400716713. DOI: 10.1145/3654522.3654548.

[4] R. Gil, J. Virgili-Gomà, J.-M. López-Gil, and R. García, "Deepfakes: Evolution and trends", *Soft Computing*, vol. 27, no. 16, pp. 11 295–11 318, 2023. DOI: 10.1007/s00500-023-08605-y.

[5] S. Robinson, K. Yasar, and S. Lewis, *What is a Generative Adversarial Network (GAN)? | Definition from TechTarget*. [Online]. Available: https://www.techtarget.com/searchenterpriseai/definition/generative-adversarial-network-GAN (visited on 11/19/2024).

[6] O. Navarro Martínez, D. Fernández-García, N. Cuartero Monteagudo, and O. Forero-Rincón, "Possible health benefits and risks of deepfake videos: A qualitative study in nursing students", *Nursing Reports*, vol. 14, no. 4, pp. 2746–2757, 2024. DOI: 10.3390/nursrep14040203.

[7] D. Philmlee, *Practice Innovations: Seeing is no longer believing — the rise of deepfakes*, Jul. 2023. [Online]. Available: https://www.thomsonreuters.com/en-us/posts/technology/practice-innovations-deepfakes/ (visited on 02/15/2025).

[8] L. Scott, *Deepfake Video Impersonates VOA Russian Service Anchor, Underscoring AI Concerns*, Oct. 2023. [Online]. Available: https://www.voanews.com/a/deepfake-video-impersonates-voa-russian-service-anchor-underscoring-ai-concerns/7333990.html (visited on 02/16/2025).

[9] KnowledgeNile, *Applications of Deepfake Technology: Its Benefits and Threats*. [Online]. Available: https://www.knowledgenile.com/blogs/applications-of-deepfake-technology-positives-and-dangers (visited on 02/15/2025).

[10] J. V. Stone, *A Very Short History of Artificial Neural Networks*, Jan. 2025. [Online]. Available: https://jimstone-68634.medium.com/a-very-short-history-of-artificial-neural-networks-9820dfd6d903 (visited on 02/19/2025).

[11] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain", *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958. DOI: 10.1037/h0042519.

[12] G. Giacaglia, *The First AI Winter (1974–1980) — Making Things Think: How AI and Deep Learning Power the Products We Use*. [Online]. Available:

https://www.holloway.com/g/making-things-think/sections/the-first-ai-winter-19741980 (visited on 02/19/2025).

[13] Adobe, *What Is CGI Animation and How Does It Work? | Adobe*. [Online]. Available: https://www.adobe.com/uk/creativecloud/animation/discover/cgi-animation.html (visited on 02/19/2025).

[14] M. Turk and A. Pentland, "Face recognition using eigenfaces", Maui, HI, USA, 1991, pp. 586–591. DOI: 10.1109/cvpr.1991.139758.

[15] Aylin, *The Evolution of Face Swapping Technology: From Funny to Creepy - Postuby*, Nov. 2023. [Online]. Available: https://postuby.com/the-evolution-of-face-swapping-technology-from-funny-to-creepy/ (visited on 02/19/2025).

[16] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks", *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, Oct. 2020. DOI: 10.1145/3422622.

[17] 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)*, Las Vegas, NV, USA, 2016, pp. 2387–2395. DOI: 10.1109/CVPR.2016.262.

[18] AFP, *'DeepNude' app to 'undress' women shut down after furor*, Jun. 2019. [Online]. Available: https://www.france24.com/en/20190628-deepnude-app-undress-women-shut-down-after-furor (visited on 02/18/2025).

[19] Iperov, *Iperov/DeepFaceLab*, Feb. 2025. [Online]. Available: https://github.com/iperov/DeepFaceLab (visited on 02/18/2025).

[20] Malavida, *FakeApp 2.2 - Download for PC Free*. [Online]. Available: https://www.malavida.com/en/soft/fakeapp/ (visited on 02/18/2025).

[21] Torzdf, *Deepfakes/faceswap*, Dec. 2017. [Online]. Available: https://github.com/deepfakes/faceswap (visited on 02/18/2025).

[22] M. Masood, M. Nawaz, K. M. Malik, A. Javed, A. Irtaza, and H. Malik, "Deepfakes generation and detection: State-of-the-art, open challenges, countermeasures, and way forward", *Applied Intelligence*, vol. 53, no. 4, pp. 3974–4026, Feb. 2023. DOI: 10.1007/s10489-022-03766-z.

[23] J. Terning, *Deep Fake of Barack Obama*, Feb. 2021. [Online]. Available: https://video.ucdavis.edu/media/Deep+Fake+of+Barack+Obama/1_6zmvebuf (visited on 12/01/2024).

[24] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, "Synthesizing Obama: Learning lip sync from audio", *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1–13, Aug. 2017. DOI: 10.1145/3072959.3073640.

[25] T. Karras, S. Laine, and T. Aila, " A Style-Based Generator Architecture for Generative Adversarial Networks ", *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 43, no. 12, pp. 4217–4228, Dec. 2021. DOI: 10.1109/TPAMI.2020.2970919.

[26] NVlabs, *NVlabs/stylegan*, Feb. 2025. [Online]. Available: https://github.com/NVlabs/stylegan (visited on 02/19/2025).

[27] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis", *Commun. ACM*, vol. 65, no. 1, pp. 99–106, Dec. 2021. DOI: 10.1145/3503250.

[28] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. C. Ferrer, *Deepfake Detection Challenge Dataset*, 2020. [Online]. Available: https://ai.meta.com/datasets/dfdc (visited on 02/19/2025).

[29] 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)*, Seoul, Korea (South), Oct. 2019, pp. 1–11. DOI: 10.1109/ICCV.2019.00009.

[30] Z. Yan, Y. Zhang, X. Yuan, S. Lyu, and B. Wu, "Deepfakebench: A comprehensive benchmark of deepfake detection", ser. NIPS '23, New Orleans, LA, USA: Curran Associates Inc., 2023. DOI: 10.5555/3666122.3666323.

[31] Facebook, *Facebook*. [Online]. Available: https://www.facebook.com/legal/ai-terms (visited on 02/19/2025).

[32] EU, *Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence and amending Regulations (EC) No 300/2008, (EU) No 167/2013, (EU) No 168/2013, (EU) 2018/858, (EU) 2018/1139 and (EU) 2019/2144 and Directives 2014/90/EU, (EU) 2016/797 and (EU) 2020/1828 (Artificial Intelligence Act) (Text with EEA relevance)*, Jun. 2024. [Online]. Available: http://data.europa.eu/eli/reg/2024/1689/oj/eng (visited on 02/19/2025).

[33] B. U. Mahmud and A. Sharmin, "Deep insights of deepfake technology : A review", 2023. DOI: 10.48550/arXiv.2105.00192.

[34] D. A. Coccomini, N. Messina, C. Gennaro, and F. Falchi, "Combining EfficientNet and Vision Transformers for Video Deepfake Detection", in *Image Analysis and Processing – ICIAP 2022*, S. Sclaroff, C. Distante, M. Leo, G. M. Farinella, and F. Tombari, Eds., Lecce, Italy: Springer International Publishing, 2022, pp. 219–229, ISBN: 978-3-031-06433-3. DOI: 10.1007/978-3-031-06433-3_19.

[35] S. A. Shamo, "The DeepFake and its Impact on Trading Signals", *SSRN Electronic Journal*, 2025. DOI: 10.2139/ssrn.5070125.

[36] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha, "Stargan v2: Diverse image synthesis for multiple domains", Seattle, WA, USA, Jun. 2020, pp. 8185–8194. DOI: 10.1109/cvpr42600.2020.00821.

[37] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, *SKTBrain/DiscoGAN*, Mar. 2017. [Online]. Available: https://github.com/SKTBrain/DiscoGAN (visited on 02/23/2025).

[38] I. Korshunova, W. Shi, J. Dambre, and L. Theis, "Fast face-swap using convolutional neural networks", in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017, pp. 3697–3705. DOI: 10.1109/ICCV.2017.397.

[39] D. A. Coccomini, R. Caldelli, F. Falchi, and C. Gennaro, "On the Generalization of Deep Learning Models in Video Deepfake Detection", *Journal of Imaging*, vol. 9, no. 5, p. 89, May 2023. DOI: 10.3390/jimaging9050089.

[40] DepFA, *Dfaker/df*, Jan. 2018. [Online]. Available: https://github.com/dfaker/df (visited on 02/25/2025).

[41] StromWine, *StromWine/DeepFake_tf*, Jul. 2018. [Online]. Available: https://github.com/StromWine/DeepFake_tf (visited on 02/25/2025).

[42] O. A. Shaaban, R. Yildirim, and A. A. Alguttar, "Audio Deepfake Approaches", *IEEE Access*, vol. 11, pp. 132 652–132 682, 2023. DOI: 10.1109/ACCESS.2023.3333866.

[43] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio", 2016. DOI: 10.48550/arXiv.1609.03499.

[44] T. Reiss, B. Cavia, and Y. Hoshen, "Detecting Deepfakes Without Seeing Any", Nov. 2023. DOI: 10.48550/arXiv.2311.01458.

[45] Talreiss, *Talreiss/FACTOR*, Oct. 2023. [Online]. Available: https://github.com/talreiss/FACTOR (visited on 11/21/2024).

[46] N. Bonettini, E. D. Cannas, S. Mandelli, L. Bondi, P. Bestagini, and S. Tubaro, "Video Face Manipulation Detection Through Ensemble of CNNs", in *2020 25th International Conference on Pattern Recognition (ICPR)*, Milan, Italy, Jan. 2021, pp. 5012–5019. DOI: 10.1109/ICPR48806.2021.9412711.

[47] Ondyari, *Ondyari/FaceForensics*, Apr. 2018. [Online]. Available: https://github.com/ondyari/FaceForensics (visited on 11/21/2024).

[48] Z. Cai, K. Stefanov, A. Dhall, and M. Hayat, "Do You Really Mean That? Content Driven Audio-Visual Deepfake Dataset and Multimodal Method for Temporal Forgery Localization", in *2022 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, Sydney, Australia, Nov. 2022, pp. 1–10. DOI: 10.1109/DICTA56598.2022.10034605.

[49] T. Lin, X. Liu, X. Li, E. Ding, and S. Wen, "Bmn: Boundary-matching network for temporal action proposal generation", Seoul, Korea (South), Oct. 2019, pp. 3888–3897. DOI: 10.1109/iccv.2019.00399.

[50] M. Nawhal and G. Mori, "Activity graph transformer for temporal action localization", 2021. DOI: 10.48550/arXiv.2101.08540.

[51] A. Bagchi, J. Mahmood, D. Fernandes, and R. Sarvadevabhatla, "Hear me out: Fusional approaches for audio augmented temporal action localization", INSTICC, SciTePress, 2022, pp. 144–154, ISBN: 978-989-758-555-5. DOI: 10.5220/0010832700003124.

[52] Z. Cai, S. Ghosh, A. Dhall, T. Gedeon, K. Stefanov, and M. Hayat, "Glitch in the matrix: A large scale benchmark for content driven audio–visual forgery detection and localization", *Computer Vision and Image Understanding*, vol. 236, p. 103 818, Nov. 2023. DOI: 10.1016/j.cviu.2023.103818.

[53] S. Tariq, S. Lee, and S. S. Woo, "One Detector to Rule Them All: Towards a General Deepfake Attack Detection Framework", in *Proceedings of the Web Conference 2021*, Ljubljana, Slovenia: Association for Computing Machinery, Apr. 2021, pp. 3625–3637, ISBN: 9781450383127. DOI: 10.1145/3442381.3449809.

[54] S. Tariq, *Shahroztariq/CLRNet*, Feb. 2021. [Online]. Available: https://github.com/shahroztariq/CLRNet (visited on 11/21/2024).

[55] SCLBD, *SCLBD/DeepfakeBench*, Jun. 2023. [Online]. Available: https://github.com/SCLBD/DeepfakeBench (visited on 09/07/2024).

[56] Z. Yan, Y. Zhao, S. Chen, X. Fu, T. Yao, S. Ding, and L. Yuan, "Generalizing Deepfake Video Detection with Plug-and-Play: Video-Level Blending and Spatiotemporal Adapter Tuning", Aug. 2024. DOI: 10.48550/arXiv.2408.17065.

[57] Y. He, B. Gan, S. Chen, Y. Zhou, G. Yin, L. Song, L. Sheng, J. Shao, and Z. Liu, "Forgerynet: A versatile benchmark for comprehensive forgery analysis", Nashville, TN, USA, Jun. 2021, pp. 4358–4367. DOI: 10.1109/cvpr46437.2021.00434.

[58] Y. Lai, Z. Yu, J. Yang, B. Li, X. Kang, and L. Shen, "Gm-df: Generalized multi-scenario deepfake detection", 2024. DOI: 10.48550/arXiv.2406.20078.

[59] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-df: A large-scale challenging dataset for deepfake forensics", Seattle, WA, USA, Jun. 2020, pp. 3204–3213. DOI: 10.1109/cvpr42600.2020.00327.

[60] B. Zi, M. Chang, J. Chen, X. Ma, and Y.-G. Jiang, "Wilddeepfake", *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 2382–2390, Oct. 2020. DOI: 10.1145/3394171.3413769.

[61] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. C. Ferrer, "The deepfake detection challenge (dfdc) dataset", 2020. DOI: 10.48550/arXiv.2006.07397.

[62] H. Song, S. Huang, Y. Dong, and W.-W. Tu, "Robustness and generalizability of deepfake detection: A study with diffusion models", 2023. DOI: 10.48550/arXiv.2309.02218.

[63] Y. Li, M.-C. Chang, and S. Lyu, "In ictu oculi: Exposing ai created fake videos by detecting eye blinking", in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, Hong Kong, China, 2018, pp. 1–7. DOI: 10.1109/WIFS.2018.8630787.

[64] U. A. Ciftci, İ. Demir, and L. Yin, "How Do the Hearts of Deep Fakes Beat? Deep Fake Source Detection via Interpreting Residuals with Biological Signals", in *2020 IEEE International Joint Conference on Biometrics (IJCB)*, Houston, TX, USA, Sep. 2020, pp. 1–10. DOI: 10.1109/IJCB48548.2020.9304909.

[65] V. N. Convertini, D. Impedovo, U. Lopez, G. Pirlo, and G. Sterlicchio, "Discrete Fourier Transform in Unmasking Deepfake Images: A Comparative Study of StyleGAN Creations", *Information*, vol. 15, no. 11, p. 711, Nov. 2024. DOI: 10.3390/info15110711.

[66] R. Lanzino, F. Fontana, A. Diko, M. R. Marini, and L. Cinque, "Faster Than Lies: Real-time Deepfake Detection using Binary Neural Networks", in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Seattle, WA, USA: IEEE, Jun. 2024, pp. 3771–3780, ISBN: 979-8-3503-6547-4. DOI: 10.1109/CVPRW63382.2024.00381.

[67] M. Uddin, Z. Fu, and X. Zhang, "Deepfake face detection via multi-level discrete wavelet transform and vision transformer", *The Visual Computer*, pp. 1–13, Jan. 2025. DOI: 10.1007/s00371-024-03791-8.

[68] M. Di Pierro, "What Is the Blockchain?", *Computing in Science & Engineering*, vol. 19, no. 5, pp. 92–95, 2017. DOI: 10.1109/MCSE.2017.3421554.

[69] N. Choi and H. Kim, "DDS: Deepfake Detection System through Collective Intelligence and Deep-Learning Model in Blockchain Environment", *Applied Sciences*, vol. 13, no. 4, p. 2122, Jan. 2023. DOI: 10.3390/app13042122.

[70] A. Qureshi, D. Megías, and M. Kuribayashi, "Detecting Deepfake Videos using Digital Watermarking", in *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, Tokyo, Japan, Dec. 2021, pp. 1786–1793. [Online]. Available: https://ieeexplore.ieee.org/document/9689555.

[71] Z. Lai, S. Arif, C. Feng, G. Liao, and C. Wang, "Enhancing Deepfake Detection: Proactive Forensics Techniques Using Digital Watermarking", *Computers, Materials & Continua*, vol. 82, no. 1, pp. 73–102, 2025. DOI: 10.32604/cmc.2024.059370.

[72] A. Alattar, R. Sharma, J. Scriven, R. Sharma, and J. Scriven, "A System for Mitigating the Problem of Deepfake News Videos Using Watermarking", *Electronic Imaging*, vol. 32, pp. 1–10, Jan. 2020. DOI: 10.2352/ISSN.2470-1173.2020.4.MWSF-117.

[73] T. B. Horvitz Eric, *New Steps to Combat Disinformation*, Sep. 2020. [Online]. Available: https://blogs.microsoft.com/on-the-issues/2020/09/01/disinformation-deepfakes-newsguard-video-authenticator/ (visited on 02/26/2025).

[74] *Sensity AI: Best Deepfake Detection Software in 2025*, Mar. 2024. [Online]. Available: https://sensity.ai/ (visited on 02/26/2025).

[75] C. Harris and L. Iannini, *The Top 8 Deepfake Detection Solutions*, Feb. 2024. [Online]. Available: https://expertinsights.com/insights/the-top-deepfake-detection-solutions/ (visited on 02/26/2025).

[76] P. Singh and D. B. Dhiman, "Exploding ai-generated deepfakes and misinformation: A threat to global concern in the 21st century", *SSRN Electronic Journal*, 2023. DOI: 10.2139/ssrn.4651093.

[77] J. Kamps and B. Kleinberg, "To the moon: Defining and detecting cryptocurrency pump-and-dumps", *Crime Science*, vol. 7, no. 1, Nov. 2018. DOI: 10.1186/s40163-018-0093-5.

[78] S. Van Der Linden, J. Roozenbeek, and J. Compton, "Inoculating against fake news about covid-19", *Frontiers in Psychology*, vol. 11, Oct. 2020. DOI: 10.3389/fpsyg.2020.566790.

[79] J. Nelson, *AI Undressing: Deepfake Nude Services Skyrocket in Popularity*, Section: News, Dec. 2023. [Online]. Available: https://decrypt.co/209181/ai-undressing-deepfake-nude-services-skyrocket-in-popularity (visited on 11/19/2024).

[80] K. Hao, *Deepfake porn is ruining women's lives. Now the law may finally ban it.* [Online]. Available: https://www.technologyreview.com/2021/02/12/1018222/deepfake-revenge-porn-coming-ban/ (visited on 02/16/2025).

[81] M. Meaker, "Slovakia's Election Deepfakes Show AI Is a Danger to Democracy", *Wired*, Oct. 2023. [Online]. Available: https://www.wired.com/story/slovakias-election-deepfakes-show-ai-is-a-danger-to-democracy/ (visited on 02/17/2025).

[82] L. d. Nadal and P. Jančárik, "Beyond the deepfake hype: AI, democracy, and "the Slovak case"", *Harvard Kennedy School Misinformation Review*, Aug. 2024. DOI: 10.37016/mr-2020-153.

[83] D. Tilles, *Opposition criticised for using AI-generated deepfake voice of PM in Polish election ad*, Section: News, Aug. 2023. [Online]. Available: https://notesfrompoland.com/2023/08/25/opposition-criticised-for-using-ai-generated-deepfake-voice-of-pm-in-polish-election-ad/ (visited on 12/01/2024).

[84] C. Linehan, G. Murphy, and J. J. Twomey, *Deepfakes in warfare: New concerns emerge from their use around the Russian invasion of Ukraine*, Oct. 2023. [Online]. Available: http://theconversation.com/deepfakes-in-warfare-new-concerns-emerge-from-their-use-around-the-russian-invasion-of-ukraine-216393 (visited on 12/01/2024).

[85] The Telegraph, *Deepfake video of Volodymyr Zelensky surrendering surfaces on social media*, Mar. 2022. [Online]. Available: https://www.youtube.com/watch?v=X17yrEV5sl4 (visited on 12/01/2024).

[86] T. W. Machine, *Serhii Sternenko on Twitter: Https://t.co/5wWC3UlpYr" / Twitter*, Mar. 2022. [Online]. Available: https://web.archive.org/web/20220318073121/https://twitter.com/sternenko/status/1504090918994993160 (visited on 02/17/2025).

[87] R. Baig, *The deepfakes in the disinformation war – DW*, Mar. 2022. [Online]. Available: https://www.dw.com/en/fact-check-the-deepfakes-in-the-disinformation-war-between-russia-and-ukraine/a-61166433 (visited on 02/17/2025).

[88] F. Dennehy, *No evidence that AI disinformation or deepfakes impacted UK, French or European elections results*, Sep. 2024. [Online]. Available: https:

//www.turing.ac.uk/news/no-evidence-ai-disinformation-or-deepfakes-impacted-uk-french-or-european-elections-results (visited on 02/17/2025).

[89] B. Wheeler and G. Corera, "Fears UK not ready for deepfake general election", Dec. 2023. [Online]. Available: https://www.bbc.com/news/uk-politics-67518511 (visited on 12/01/2024).

[90] S. Intelligence, *How AI has affected the 2024 UK election*, Jun. 2024. [Online]. Available: https://www.verdict.co.uk/uk-election-deepfakes-ai/ (visited on 02/17/2025).

[91] F. Dennehy, *9 in 10 concerned about deepfakes affecting election results*, Jul. 2024. [Online]. Available: https://www.turing.ac.uk/news/9-10-concerned-about-deepfakes-affecting-election-results (visited on 02/17/2025).

[92] F. Chollet, "Xception: Deep learning with depthwise separable convolutions", Honolulu, HI, USA, Jul. 2017, pp. 1800–1807. DOI: 10.1109/cvpr.2017.195.

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

[94] L. Verdoliva, "Media forensics and deepfakes: An overview", *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 5, pp. 910–932, 2020. DOI: 10.1109/JSTSP.2020.3002101.

[95] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection", *Information Fusion*, vol. 64, pp. 131–148, 2020. DOI: https://doi.org/10.1016/j.inffus.2020.06.014.

[96] L. Li, J. Bao, T. Zhang, H. Yang, D. Chen, F. Wen, and B. Guo, "Face x-ray for more general face forgery detection", Seattle, WA, USA, Jun. 2020, pp. 5000–5009. DOI: 10.1109/cvpr42600.2020.00505.

[97] H.-S. Chen, S. Hu, S. You, and C.-C. J. Kuo, "Defakehop++: An enhanced lightweight deepfake detector", *APSIPA Transactions on Signal and Information Processing*, vol. 11, no. 2, 2022. DOI: 10.1561/116.00000126.

[98] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for mobilenetv3", Seoul, Korea (South), Oct. 2019, pp. 1314–1324. DOI: 10.1109/iccv.2019.00140.

[99] J. Liu, K. Zhu, W. Lu, X. Luo, and X. Zhao, "A lightweight 3d convolutional neural network for deepfake detection", *International Journal of Intelligent Systems*, vol. 36, no. 9, pp. 4990–5004, Jun. 2021. DOI: 10.1002/int.22499.

[100] EndlessSora, *DeeperForensics-1.0/dataset at master · EndlessSora/DeeperForensics-1.0 · GitHub*. [Online]. Available: https://github.com/EndlessSora/DeeperForensics-1.0/tree/master/dataset#overview (visited on 03/30/2025).

[101] N. Dufour and A. Gully, *Contributing Data to Deepfake Detection Research*, Sep. 2019. [Online]. Available: https://research.google/blog/contributing-data-to-deepfake-detection-research/ (visited on 03/30/2025).

[102] J. Pu, N. Mangaokar, L. Kelly, P. Bhattacharya, K. Sundaram, M. Javed, B. Wang, and B. Viswanath, "Deepfake videos in the wild: Analysis and detection", in *Proceedings of the Web Conference 2021*, ser. WWW '21, Ljubljana, Slovenia: Association for Computing Machinery, 2021, pp. 981–992, ISBN: 9781450383127. DOI: 10.1145/3442381.3449978.

[103] Google, *Classification: Accuracy, recall, precision, and related metrics | Machine Learning*. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall (visited on 04/05/2025).

[104] Google, *Classification: ROC and AUC | Machine Learning*. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc (visited on 04/20/2025).

[105] T. S. Team, *Understanding true positive rate in software testing*. [Online]. Available: https://www.statsig.com/perspectives/true-positive-rate-software-testing (visited on 04/29/2025).

[106] GeeksforGeeks, *Auc roc curve in machine learning*, Feb. 2025. [Online]. Available: https://www.geeksforgeeks.org/auc-roc-curve/ (visited on 04/20/2025).

[107] Scikit-learn, *F1-score*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html (visited on 04/20/2025).

[108] Isolomons, *Our Approach to Labeling AI-Generated Content and Manipulated Media*, Apr. 2024. [Online]. Available: https://about.fb.com/news/2024/04/metas-approach-to-labeling-ai-generated-content-and-manipulated-media/ (visited on 05/04/2025).

[109] EU, *European AI Act: Mandatory Labeling for AI-Generated Content*, Apr. 2024. [Online]. Available: https://www.imatag.com/blog/ai-act-legal-requirement-to-label-ai-generated-content (visited on 05/04/2025).

[110] C. Deschaseaux, *Integrating Watermarking into C2PA Standards: A Must for Online Content Authenticity*, Mar. 2024. [Online]. Available: https://www.

imatag.com/blog/enhancing-content-integrity-c2pa-invisible-watermarking (visited on 05/04/2025).

[111] Tara, *Finland trains six-year-olds to spot fake news – and they might be better at it than you*, Jan. 2025. [Online]. Available: https://www.helsinkitimes.fi/themes/themes/education/26048-finland-trains-six-year-olds-to-spot-fake-news-and-they-might-be-better-at-it-than-you.html (visited on 05/04/2025).

[112] ParisCall, *Paris Call for Trust and Security in Cyberspace — Paris Call*. [Online]. Available: https://pariscall.international/en/ (visited on 05/04/2025).

Appendix A Appendix

Appendix program code in Listing 4 outlined the Python script used to export the CLRNet pre-trained model from the TensorFlow 1.x environment to the TensorFlow 2.x environment to make it compatible before evaluating in the JupyterLab environment equipped with the latest TensorFlow, Keras, and Python packages. Appendix program code in Listing 5 describes the Python script used to separate real and fake videos from the LAV-DF test dataset split using the included metadata file. Appendix program code in Listing 6 describes the Python script used for randomly selecting samples from each dataset. Appendix program code in Listings 7 and 8 illustrates the Python script for generating the metadata file for each dataset (except the LAV-DF and the DFW datasets) for the BA-TFD+ model testing.

Appendix program code in Listing 9 describes the Python code used to generate compressed videos in three different levels for BA-TFD+ model testing. Appendix program code in Listing 10 describes the Python code to generate CSV files for each dataset for the Cross-Efficient ViT model. Appendix program code in Listings 11 and 12 describes the Python script used for video compression in three levels in each dataset in each video for the CLRNet model. Appendix program code in Listings 13 and 14 describes the Python script used for extracting frames from each selected video in the CLRNet model.

Listing 4 Python script to export CLRNet pre-trained model from TensorFlow 1.x environment to TensorFlow 2.x environment.

```
import h5py
import tensorflow as tf
from tensorflow.keras.models import model_from_json
import tensorflow_core.python.keras.saving.hdf5_format as h5f

HDF5_PATH = '/path/to/the/original/CLRNet/model/CLR_ALL_Ebest.hdf5'
SAVED_MODEL_DIR = '/path/to/the/output/model/saved_model_tf1/'

# Monkey-patch TF1's HDF5 weight-loader to catch any stray str.decode calls
_orig_load = h5f.load_weights_from_hdf5_group

def _patched_load(f, layers):
    try:
        return _orig_load(f, layers)
    except AttributeError as e:
        if "'str' object has no attribute 'decode'" in str(e):
            # fix keras_version or backend if still str
            for attr in ('keras_version', 'backend'):
                val = f.attrs.get(attr)
                if isinstance(val, str):
                    print(f"Re-encoding HDF5 attr '{attr}' to bytes on disk")
                    f.attrs[attr] = val.encode('utf-8')
            return _orig_load(f, layers)
        raise

h5f.load_weights_from_hdf5_group = _patched_load

# Re-encode model_config, keras_version, backend on disk as bytes
with h5py.File(HDF5_PATH, 'r+') as f:
    for attr in ('model_config', 'keras_version', 'backend'):
        val = f.attrs.get(attr)
        if isinstance(val, str):
            print(f"Re-encoding HDF5 attr '{attr}' to bytes on disk")
            f.attrs[attr] = val.encode('utf-8')

# Extract & parse the JSON architecture
with h5py.File(HDF5_PATH, 'r') as f:
    raw = f.attrs['model_config']
    arch = raw.decode('utf-8') if isinstance(raw, (bytes, bytearray)) else raw

# Reconstruct and load weights
model = model_from_json(arch)
print("Architecture reconstructed")

# Check if the weights can be loaded without issues
try:
    model.load_weights(HDF5_PATH)
    print("Weights loaded")
except Exception as e:
    print(f"Error loading weights: {e}")

# Export as SavedModel for TF-2.x
tf.saved_model.save(model, SAVED_MODEL_DIR)
print(f"SavedModel written to {SAVED_MODEL_DIR}")
```

Listing 5 Python script for separate real and fake videos of LAV-DF-test dataset using metadata file.

```
import os
import json
import shutil

# Paths
METADATA_FILE = "/path/to/metadata.json"  # Path to the metadata file
LAV_DF_TEST_DIR = "/path/to/dataset"  # Folder containing all videos
OUTPUT_REAL_DIR = "/path/to/output/real"
OUTPUT_FAKE_DIR = "/path/to/output/fake"

# Create output directories if they don't exist
os.makedirs(OUTPUT_REAL_DIR, exist_ok=True)
os.makedirs(OUTPUT_FAKE_DIR, exist_ok=True)

# Load metadata
with open(METADATA_FILE, "r") as f:
    metadata = json.load(f)

# Process videos
for entry in metadata:
    video_file = entry["file"].split("/")[-1]  # Extract filename
    video_path = os.path.join(LAV_DF_TEST_DIR, video_file)

    if not os.path.exists(video_path):
        print(f"Skipping missing file: {video_file}")
        continue

    if entry["n_fakes"] == 0:
        target_dir = OUTPUT_REAL_DIR  # Real video
    else:
        target_dir = OUTPUT_FAKE_DIR  # Fake video

    shutil.move(video_path, os.path.join(target_dir, video_file))
    print(f"Moved: {video_file} → {target_dir}")

print("Separation completed!")
```

Listing 6 Python script for random data selection from each dataset.

```
import os
import random
import shutil

# Define dataset directories
DATASET_DIR = "/path/to/the/dataset"  # Dataset path
OUTPUT_DIR = "/path/to/the/output"  # Save location for selected samples

# Selection criteria
SELECTION_CRITERIA = {
    "Celeb-real": (500, "Celeb DF (v2)/Celeb-real"),
    "Celeb-synthesis": (500, "Celeb DF (v2)/Celeb-synthesis"),
    "source_videos": (500, "DeeperForensics-1.0/source_videos"),
    "manipulated_videos": (500, "DeeperForensics-1.0/manipulated_videos"),
    "DFD_original_sequences": (250, "DFD/DFD_original_sequences"),
    "DFD_manipulated_sequences": (250, "DFD/DFD_manipulated_sequences"),
    "LAV-DF-test-real": (500, "LAV-DF/LAV-DF-test-real"),
    "LAV-DF-test-fake": (500, "LAV-DF/LAV-DF-test-fake"),
    "DFW": (500, "DFW")  # DFW has only fake videos
}

def select_and_copy_videos(category, num_samples, source_folder):
    source_path = os.path.join(DATASET_DIR, source_folder)
    if not os.path.exists(source_path):
        print(f"Skipping missing directory: {source_path}")
        return

    videos = [f for f in os.listdir(source_path) if f.endswith(('mp4', 'avi', 'mov', 'mkv'))]
    if len(videos) < num_samples:
        print(f"Warning: Only {len(videos)} videos found in {category}, selecting all available.")
        selected_videos = videos  # Use all available if not enough
    else:
        selected_videos = random.sample(videos, num_samples)  # Randomly select required number

    # Create output directory
    output_path = os.path.join(OUTPUT_DIR, category)
    os.makedirs(output_path, exist_ok=True)

    # Copy selected videos
    for video in selected_videos:
        shutil.copy2(os.path.join(source_path, video), os.path.join(output_path, video))
    print(f" Selected {len(selected_videos)} videos for {category}")

# Run selection process for each dataset category
for category, (num_samples, source_folder) in SELECTION_CRITERIA.items():
    select_and_copy_videos(category, num_samples, source_folder)

print("Video selection completed.")
```

Listing 7 Python script for generating metadata file for each dataset (except LAV-DF and DFW datasets) for BA-TFD+ model testing (Part 1).

```
import os, json, subprocess
from pathlib import Path
from tqdm import tqdm
from difflib import get_close_matches

def get_video_info(video_path):
    """Returns duration, frame count, and audio frame count using ffprobe."""
    cmd = [
        "ffprobe", "-v", "error",
        "-select_streams", "v:0", "-count_frames",
        "-show_entries", "stream=nb_read_frames,duration",
        "-of", "json", str(video_path)
    ]
    video_info = subprocess.run(cmd, capture_output=True, text=True)
    video_json = json.loads(video_info.stdout)

    cmd_audio = [
        "ffprobe", "-v", "error",
        "-select_streams", "a:0",
        "-show_entries", "stream=channels,nb_frames",
        "-of", "json", str(video_path)
    ]
    audio_info = subprocess.run(cmd_audio, capture_output=True, text=True)
    audio_json = json.loads(audio_info.stdout)

    duration = float(video_json["streams"][0].get("duration", 0))
    video_frames = int(video_json["streams"][0].get("nb_read_frames", 0))

    audio_channels = int(audio_json["streams"][0].get("channels", 1)) if audio_json["streams"] else 1
    audio_frames = int(audio_json["streams"][0].get("nb_frames", 0)) if audio_json["streams"] else 0

    return duration, video_frames, audio_channels, audio_frames

def find_best_original_match(fake_filename, real_filenames):
    """Tries to find the closest real filename to the fake one using fuzzy matching."""
    fake_basename = os.path.splitext(fake_filename)[0]
    candidates = [os.path.splitext(real)[0] for real in real_filenames]

    match = get_close_matches(fake_basename, candidates, n=1, cutoff=0.6)
    if match:
        return f"test/{match[0]}.mp4"
    return None

def generate_metadata(dataset_path, output_path, dataset_name="Dataset"):
    metadata = []
    dataset_path = Path(dataset_path)

    # List all real filenames for matching
    real_dir = dataset_path / "real"
    real_videos = [f.name for f in real_dir.glob("*.mp4")]
    real_stems = {os.path.splitext(f)[0]: f for f in real_videos}

    for split in ["real", "fake"]:
        files = list((dataset_path / split).glob("*.mp4"))

        for video_file in tqdm(files, desc=f"[{dataset_name}] Processing {split} videos"):
            try:
                duration, video_frames, audio_channels, audio_frames = get_video_info(video_file)

                if split == "real":
                    original = None
                    fake_periods = []
                    n_fakes = 0
                    modify_video = False
                    modify_audio = False
                else:
                    original = find_best_original_match(video_file.name, real_videos)
                    fake_periods = [[0.0, round(duration * 0.9, 3)]]
                    n_fakes = len(fake_periods)
                    modify_video = True
                    modify_audio = True
```

Listing 8 Python script for generating metadata file for each dataset (except LAV-DF and DFW datasets) for BA-TFD+ model testing (Part 2).

```
                entry = {
                    "file": f"test/{video_file.name}",
                    "n_fakes": n_fakes,
                    "fake_periods": fake_periods,
                    "duration": duration,
                    "original": original,
                    "modify_video": modify_video,
                    "modify_audio": modify_audio,
                    "split": "test",
                    "video_frames": video_frames,
                    "audio_channels": audio_channels,
                    "audio_frames": audio_frames
                }

                metadata.append(entry)
            except Exception as e:
                print(f"Failed processing {video_file.name}: {e}")

    os.makedirs(output_path, exist_ok=True)
    output_file = Path(output_path) / "metadata.min.json"
    with open(output_file, "w") as f:
        json.dump(metadata, f, indent=4)
    print(f"[{dataset_name}] Saved metadata to: {output_file}")

# Dataset paths
datasets = {
    "Celeb-DF-v2": "/path/to/the/Celeb/dataset",
    "DeeperForensics-1.0": "/path/to/the/DeeperForensics/dataset",
    "DFD": "/path/to/the/DFD/dataset"
}

output_base = "/base/path/to/the/output"

for name, path in datasets.items():
    generate_metadata(path, os.path.join(output_base, name), dataset_name=name)
```

Listing 9 Python script for generating videos in three compression levels for each dataset for BA-TFD+ model testing.

```
import os
import subprocess
from pathlib import Path
from tqdm import tqdm

# Define dataset directories
datasets = ["Celeb-DF-v2/test", "DeeperForensics-1.0/test", "LAV-DF/test", "DFD/test", "DFW/test"]

# Base directory where datasets are stored
BASE_DIR = "/base/path/to/the/datasets"
OUTPUT_DIR = "/output/path/to/the/compressed/data"

# Compression levels using CRF (lower CRF = better quality, higher CRF = more compression)
compression_levels = {
    "C0": 18,   # High quality, almost lossless
    "C23": 23,  # Medium quality (default for FFmpeg)
    "C40": 40   # Low quality, more compression
}

# Ensure output directory exists
Path(OUTPUT_DIR).mkdir(parents=True, exist_ok=True)

# Loop through each dataset
for dataset in datasets:
    dataset_path = os.path.join(BASE_DIR, dataset)
    if not os.path.exists(dataset_path):
        print(f"Dataset not found: {dataset_path}, skipping...")
        continue

    # Get all video files in the dataset
    video_files = [f for f in os.listdir(dataset_path) if f.endswith(('.mp4', '.avi', '.mov'))]

    for video in tqdm(video_files, desc=f"Compressing {dataset}"):
        input_video_path = os.path.join(dataset_path, video)

        # Process each compression level
        for level, crf in compression_levels.items():
            output_folder = os.path.join(OUTPUT_DIR, dataset, level)
            Path(output_folder).mkdir(parents=True, exist_ok=True)

            output_video_path = os.path.join(output_folder, video)

            if os.path.exists(output_video_path):
                print(f"Already exists: {output_video_path}, skipping...")
                continue

            # FFmpeg command for compression
            ffmpeg_cmd = [
                "ffmpeg", "-i", input_video_path, "-c:v", "libx264", "-preset", "slow",
                "-crf", str(crf), "-c:a", "aac", "-b:a", "128k", output_video_path
            ]

            # Run the FFmpeg command
            try:
                subprocess.run(ffmpeg_cmd, check=True)
                print(f"Compressed {video} -> {level} ({crf})")
            except subprocess.CalledProcessError:
                print(f"Error compressing {video} at {level}")

print("Compression process completed.")
```

Listing 10 Python script for generating CSV files for each dataset for Cross-Efficient ViT model.

```
import os
import csv

# Base folder:
BASE_DIR = "/path/to/the/base/directory/"

# Write out the CSV:
OUTPUT_CSV = "/path/to/the/outout/file/datasetname_test_labels.csv"

with open(OUTPUT_CSV, "w", newline="") as f:
    writer = csv.writer(f)
    # header
    writer.writerow(["filename", "label"])

    # walk exactly one level down from BASE_DIR
    for entry in sorted(os.listdir(BASE_DIR)):
        dir_path = os.path.join(BASE_DIR, entry)
        if not os.path.isdir(dir_path):
            continue

        # for DFD and others, entry=="real" and entry=="fake"
        # for DFW, entry=="DFW", which treat as "fake"
        label = "real" if entry.lower() == "real" else "fake"

        # now collect all .mp4 files in that dir
        for fname in sorted(os.listdir(dir_path)):
            if fname.lower().endswith(".mp4"):
                writer.writerow([fname, label])

print(f"Done! Wrote labels to {OUTPUT_CSV}")
```

Listing 11 Python script to compress each dataset's videos in three compression levels for CLRNet model testing (Part 1).

```
import os
import subprocess
from pathlib import Path
from tqdm import tqdm

# Define dataset directories (Modify these paths as per dataset location)
datasets = ["Celeb-DF-v2", "DeeperForensics-1.0", "LAV-DF", "DFD", "DFW"]

# Base directory where datasets are stored
BASE_DIR = "/path/to/the/base/directory/"
OUTPUT_DIR = "/path/to/the/output/directory/"

# Compression levels using CRF (lower CRF = better quality, higher CRF = more
compression)
compression_levels = {
    #"C0": 18,   # High quality, almost lossless
    "C23": 23,  # Medium quality (default for FFmpeg)
    "C40": 40   # Low quality, more compression
}

# Ensure output directory exists
Path(OUTPUT_DIR).mkdir(parents=True, exist_ok=True)

for dataset in datasets:
    ds_path = os.path.join(BASE_DIR, dataset)
    if not os.path.isdir(ds_path):
        print(f"Dataset not found, skipping: {ds_path}")
        continue

    # Expect inside each dataset folder either both 'real' and 'fake', or 'fake'
    for DFW.
    for subset in ("real", "fake"):
        in_folder = os.path.join(ds_path, subset)
        if not os.path.isdir(in_folder):
            # e.g. DFW has no 'real' folder
            continue

        # find all videos (*.mp4, *.avi, *.mov) in this subset folder
        video_files = sorted([
            f for f in os.listdir(in_folder)
            if f.lower().endswith((".mp4", ".avi", ".mov"))
        ])
        if not video_files:
            print(f"  No videos in {in_folder}, skipping.")
            continue

        pbar = tqdm(video_files, desc=f"{dataset}/{subset}", unit="vid")
        for video in pbar:
            input_path = os.path.join(in_folder, video)

            for level, crf in compression_levels.items():
                out_folder = os.path.join(OUTPUT_DIR, dataset, subset, level)
                Path(out_folder).mkdir(parents=True, exist_ok=True)
```

Listing 12 Python script to compress each dataset's videos in three compression levels for CLRNet model testing (Part 2).

```
                output_path = os.path.join(out_folder, video)
                if os.path.exists(output_path):
                    # already done
                    continue

                # build ffmpeg command
                cmd = [
                    "ffmpeg", "-y",  # overwrite if exists
                    "-i", input_path,
                    "-c:v", "libx264", "-preset", "slow",
                    "-crf", str(crf),
                    "-c:a", "aac", "-b:a", "128k",
                    output_path
                ]
                try:
                    subprocess.run(cmd, check=True, stdout=subprocess.DEVNULL,
                    stderr=subprocess.DEVNULL)
                except subprocess.CalledProcessError:
                    print(f"Error compressing {input_path} → {level}")
        pbar.close()

print("All datasets processed.")
```

Listing 13 Python script for extracting frames from each video for CLRNet model testing (Part 1).

```
#!/usr/bin/env python3
import os
import glob
import cv2
import argparse
from tqdm import tqdm

DATASETS = ["DEEPERFORENSICS", "LAV-DF", "CELEB-DF-V2", "DFD", "DFW"]

def extract_all(videos_root, extracted_root, fps_skip=1):
    for ds in DATASETS:
        for label in ("real", "fake"):
            # DFW has only 'fake'
            if ds == "DFW" and label == "real":
                continue

            in_dir = os.path.join(videos_root, ds, label)
            out_base = os.path.join(extracted_root, ds, label)
            os.makedirs(out_base, exist_ok=True)

            mp4s = sorted(glob.glob(os.path.join(in_dir, "*.mp4")))
            if not mp4s:
                print(f"[!] no videos found in {in_dir!r}, skipping")
                continue

            for vpath in tqdm(mp4s, desc=f"{ds}/{label}"):
                name = os.path.splitext(os.path.basename(vpath))[0]
                out_dir = os.path.join(out_base, name)
                os.makedirs(out_dir, exist_ok=True)

                cap = cv2.VideoCapture(vpath)
                frame_idx = 0
                saved_idx = 0

                while True:
                    ret, frame = cap.read()
                    if not ret:
                        break
                    if frame_idx % fps_skip == 0:
                        # zero-pad the frame number
                        fn = os.path.join(out_dir,
                        f"{name}_frame_{saved_idx:04d}.png")
                        cv2.imwrite(fn, frame)
                        saved_idx += 1
                    frame_idx += 1

                cap.release()
```

Listing 14 Python script for extracting frames from each video for CLRNet model testing (Part 2).

```
if __name__ == "__main__":
    parser = argparse.ArgumentParser(
        description="Extract frames from mp4s into per-video folders."
    )
    parser.add_argument(
        "--videos-root",
        type=str,
        required=True,
        help="Root folder containing your five datasets of mp4s"
    )
    parser.add_argument(
        "--extracted-root",
        type=str,
        required=True,
        help="Where to write extracted_frames/"
    )
    parser.add_argument(
        "--fps-skip",
        type=int,
        default=1,
        help="Extract every Nth frame (default=1 → every frame)"
    )
    args = parser.parse_args()

    extract_all(args.videos_root, args.extracted_root, args.fps_skip)
    print("DONE")
```