UNIVERSITÉ DU LUXEMBOURG

PhD-FSTM-2026-039
The Faculty of Science, Technology and Medicine

DISSERTATION

Defence held on 26 March 2026 in Luxembourg

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN INFORMATIQUE

by

VĂN ĐẠT NGUYỄN
Born on 29 December 1994 in HÀ NỘI (Viet Nam)

MODELING AND EXPLOITING VULNERABILITIES FOR DEEPFAKE DETECTION

Dissertation defence committee

Dr. Djamila AOUADA, Dissertation Supervisor
Associate Professor, SnT, Université du Luxembourg

Dr. Gilbert FRIDGEN, Chairman
Professor, SnT, Université du Luxembourg

Dr. Wassim HAMIDOUCHE
Principal Research Scientist, Microsoft AI for Good Lab

Dr. Cu NGUYEN
Lead R&D in Telco Security and Frauds, POST Luxembourg

Dr. Luca GUARNERA
Assistant Professor, University of Catania

Affidavit / Statement of originality

I declare that this thesis:

- is the result of my own work. Any contribution from any other party, and any use of generative artificial intelligence technologies have been duly cited and acknowledged;
- is not substantially the same as any other that I have submitted, and;
- is not being concurrently submitted for a degree, diploma or other qualification at the University of Luxembourg or any other University or similar institution except as specified in the text.

With my approval I furthermore confirm the following:

- I have adhered to the rules set out in the University of Luxembourg's Code of Conduct and the Doctoral Education Agreement (DEA)[1], in particular with regard to Research Integrity.
- I have documented all methods, data, and processes truthfully and fully.
- I have mentioned all the significant contributors to the work.
- I am aware that the work may be screened electronically for originality.

I acknowledge that if any issues are raised regarding good research practices based on the review of the thesis, the examination may be postponed pending the outcome of any investigation of such issues. If a degree was conferred, any such subsequently discovered issues may result in the cancellation of the degree.

Approved on 2026-02-18

[1] If applicable (DEA is compulsory since August 2020)

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, **Prof. Djamila Aouada**, for her continuous guidance, trust, and support throughout my PhD journey. I am also deeply thankful to my mentor, **Prof. Enjie Ghorbel**, for her invaluable advice, constructive feedback, and constant availability, as well as to **Dr. Anis Kacem** for his guidance and many helpful discussions that greatly influenced the direction and quality of this work.

I would also like to thank my colleagues at **CVI^2-SnT** for creating a friendly and stimulating working environment. In particular, I am grateful to my office mates **Mohamed Adel**, **Nassim**, and **Peyman**, as well as **Elona**, **Astrid**, and **other members** of the group for their support, collaboration, and everyday moments we shared.

I am further thankful to my friends at **SnT**, including **anh Vu Ha**, **anh Thang Vu**, **em Hung Kha**, and **anh Dung Tran**, who have supported me and my small family during my PhD and our life in Luxembourg. Their kindness and practical help made many difficult periods much easier to overcome.

I also gratefully acknowledge the financial and computational support that made this work possible. This thesis is supported by the **Luxembourg National Research Fund (FNR)** under the project **BRIDGES2021/IS/16353350/FaKeDeTeR**, and by **POST Luxembourg**. A large part of the experiments was performed on the Luxembourg national supercomputer **MeluXina**; I sincerely thank the **LuxProvide** teams for their expert technical support.

I also wish to express my gratitude to the lab **HMI** at **UET-VNU**, where I had the opportunity to grow academically during my Master's studies. I am deeply thankful to **Prof. Ha Le Thanh**, **Dr. Duyen Ngo**, and **Dr. Chau Ma** for their dedicated supervision, guidance, and encouragement during my time there, which laid an essential foundation for my subsequent research and for undertaking this PhD.

I am deeply grateful to my first company, **Sun Asterisk**, the place that first accepted me and gave me the opportunity to learn and grow. I would like to thank my leaders there, especially **Prof. Thanh Minh Ta**, who was both my leader and my first research mentor, as well as **chi Mo**, **anh Quang**, **chi Hang**, **anh Thang**, and **anh Toan**, for their mentorship, trust, and for providing an environment where I could build my foundational skills.

I further acknowledge **VinAI Research** (now Qualcomm AI Research), my previous company, and my leaders **Prof. Minh Hoai** and **Dr. Tri Dzung**, for enabling me to make significant progress in my professional expertise and for entrusting me with challenging and impactful projects that strongly contributed to my development before starting this PhD.

Finally, my deepest appreciation goes to **my family**. I am profoundly thankful to **my parents** for their unconditional love and support, and especially to my wife **Phuong Linh** and my daughter **Cam** (Orange) for always standing by my side during the most difficult moments. Their patience, sacrifice, and encouragement have been the greatest source of strength for me to complete this thesis.

Contents

Acronyms

AI Artificial Intelligence. 19, 101

AP Average Precision. 13, 57, 120

AUC Area Under the Receiver Operating Characteristic (ROC) Curve. 10, 13, 25, 26, 46, 50, 57, 65, 75, 94, 96, 101, 120, 148, 153, 154, 156

CNN Convolutional Neural Network. 26, 36, 37, 47, 63–66, 119, 120, 144, 146

CNNs Convolutional Neural Networks. 20, 23, 38, 39, 44, 64, 66, 67, 69, 81, 144

DL Deep Learning. 48

DNN Deep Neural Network. 9, 21, 51

DNNs Deep Neural Networks. 19, 20, 50

E-FPN Enhanced Feature Pyramid Network. 13, 50, 56, 59, 139

FLOPs Floating Point Operations Per Second. 10, 65, 68, 69, 75, 143

FPN Feature Pyramid Network. 26, 38, 50, 51, 56, 139

FPNs Feature Pyramid Networks. 38

FPR False Positive Rate. 105

GenAI Generative Artificial Intelligence. 19, 20

HQ High-Quality. 20–23, 25, 26, 28, 67, 69, 79, 85

Mask-SSIM Mask Structural SIMilarity. 26, 50, 57, 68

MHSA Multi-Head Self-Attention. 41, 43, 44, 143

MLP Multi-Layer Perceptron. 42, 143, 144

NLP Natural Language Processing. 39

SA Self-Attention. 9, 39–43, 45, 98, 152

SOTA State-of-the-Art. 27, 28, 51, 57, 68, 74, 76, 84, 93, 94, 96, 99, 108

TPR True Positive Rate. 105

ViT Vision Transformer. 39, 42–45, 64–66, 68, 70, 71, 73, 144

ViTs Vision Transformers. 20, 23, 27, 41, 43, 64–67, 69, 71, 79, 80, 144

VLMs Vision-Language Models. 20

List of Figures

List of Tables

Abstract

The rapid progress of generative artificial intelligence has enabled the creation of highly realistic facial manipulations, commonly referred to as deepfakes. While such technologies offer significant benefits for media production and human-computer interaction, they also pose serious risks to privacy, security, and societal trust. Ensuring the authenticity of visual content has therefore become a critical challenge. Although recent deep learning-based detectors have demonstrated strong performance under controlled benchmarks, their reliability remains insufficient for real-world deployment, where manipulations are diverse, continuously evolving, and often imperceptible to human observers. Current deepfake detectors typically rely on deep neural networks that are trained in a supervised manner as binary classifiers that differentiate fake and real data.

Hence, they suffer from two main limitations. First, they are inevitably subject to overfitting issues and tend, therefore, to achieve poor generalization to unseen data variations. Specifically, their performance degrades significantly under unseen domains, high-quality forgeries, or real-world distortions. Second, deep architectures, including CNN and transformers, fail to capture by definition localized artifact-prone features, which typically characterize deepfakes, especially high-quality ones. Addressing these limitations requires moving beyond conventional binary classification toward models capable of modeling localized areas where generative processes are more likely to introduce subtle artifacts.

This thesis introduces a unified vulnerability-aware framework for deepfake detection across both images and videos. The central idea is to explicitly guide the learning process toward vulnerable regions that are defined as the areas that are more likely to incorporate blending artifacts. Since blending artifacts are common to most generation methods, this strategy enables detectors to focus on generic artifacts rather than patterns specific to a given dataset. At the frame level, a CNN-based framework, called LAA-Net, aims at modeling fine-grained vulnerabilities at the pixel level in deepfake images. This is achieved through multi-task learning strategies that jointly perform detection and artifact localization while integrating complementary multi-scale representations. This concept is further extended to

transformer architectures through LAA-Former, which transfers vulnerability modeling from pixels to patches, unifying explicit local supervision and global relational reasoning within a single framework. As a result, it contributes to improving cross-domain generalization and robustness to noise while maintaining computational efficiency.

Extending this idea to video analysis, a vulnerability-aware spatio-temporal framework, namely FakeSTormer, is developed to capture intertwined spatial and temporal inconsistencies introduced during video synthesis. By disentangling the learning of temporal and spatial vulnerabilities, the proposed approach improves robustness against diverse high-quality and unseen deepfake videos.

Beyond model design, this thesis revisits the evaluation of deepfake detectors under realistic conditions. In fact, current evaluation protocols rely on AUC solely estimated on separate datasets. This cannot guarantee the maturity of deepfake detectors for real-world deployment, since strong AUC on each dataset in isolation can mask instability when data are mixed with strong distribution shifts. To address this issue, a new evaluation metric, termed Cross-AUC, is introduced to assess generalization stability across uncontrolled and mixed-domain scenarios.

Together, the proposed contributions advance the principled design, analysis, and evaluation of deepfake detection systems toward more generalizable, robust, and interpretable solutions, bringing automated deepfake detection closer to practical real-world applicability.

Chapter 1

Introduction

We are living in an era where Artificial Intelligence (AI) profoundly transforms human life and reshapes the way we interact with the world. Generative AI (GenAI) [1, 2, 3, 4], a prominent branch of AI, refers to models capable of *learning patterns from existing data and generating entirely new content that resembles it.* It can automatically produce realistic content across diverse modalities from text and images to audio and video. Popular examples include ChatGPT [5], Gemini [6], and Grok [7], which demonstrate the versatility of GenAI in providing personalized, interactive digital experiences. As these tools become deeply integrated into daily life, GenAI is getting more and more present and intertwining into our modern societal life.

Among all forms of generative content, face synthesis [12, 13, 14] has drawn particular attention, largely because *human faces are central to identity, communication, and trust*, making them both highly engaging and extremely dangerous. Powered by Deep Neural Networks (DNNs), GenAI systems now can create hyper-realistic artificial faces, commonly known as **Deepfakes**, which are often indistinguishable from real ones to the naked eye. Taxonomies of deepfakes are shown in Figure 1.1.

Despite their potential for positive applications, such as remote education [15] (creating synthetic instructors) or digital entertainment [16] (virtual actors in films), deepfakes also pose serious ethical and security risks. When misused, they can facilitate political misinformation [17, 18], financial fraud [19], or non-consensual explicit content [20]. The widespread

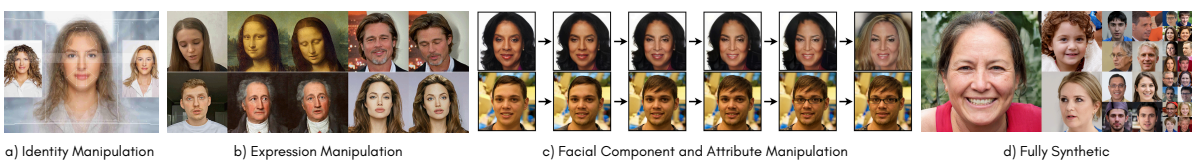


Figure 1.1 Taxonomies of Deepfakes: a) Identity Manipulation [8], b) Expression Manipulation [9], c) Facial Components and Attributes Manipulation [10] and d) Fully Synthetic [11].

and malicious use of deepfakes threatens to erode public trust and blur the boundary between truth and deception.

There is, therefore, an urgent need to develop automatic and reliable deepfake detection systems, especially those capable of identifying high-quality (HQ) and realistic forgeries.

1.1 Motivation and Scope

Deepfake detection has become an increasingly important and active research area in computer vision, especially in the era of Generative AI (GenAI).

In recent years, this topic has been widely explored by the research community, evolving from simple machine learning-based detectors [21, 22] to increasingly complex and large-scale deep learning architectures [23, 24, 25, 26, 27, 28, 29, 30]. Early approaches relied on handcrafted features combined with traditional machine learning algorithms, which were effective for detecting early low-quality forgeries containing visual artifacts.

With the remarkable progress of Deep Neural Networks (DNNs), most recent works leverage powerful architectures such as Convolutional Neural Networks (CNNs) [31, 26, 32, 23, 33], Vision Transformers (ViTs) [34, 35, 36, 37], and Vision-Language Models (VLMs) [38] to improve detection accuracy and generalization. Despite these advances, the deployment of deepfake detection systems in industry remains limited. This thesis aims to advance the field of deepfake detection by proposing methods and solutions that move research closer to real-world standards. The focus is on developing detection frameworks that are generalizable, robust, interpretable, and practically applicable across diverse scenarios, contributing to a more trustworthy and secure use of generative technologies.

1.2 Challenges

As discussed earlier, despite remarkable progress in recent years, current deepfake detectors are not mature enough for practical deployment. There remains a clear gap between research performance and real-world reliability. This gap is caused by several persistent challenges (Figure 1.2), including: *1) Limited generalization to unseen forgeries*; *2) Lack of robustness to HQ deepfakes and common perturbations such as noise, compression, etc.*; and *3) An incomplete evaluation of practicality*, such as efficiency, scalability, and interpretability. Moreover, existing evaluation protocols do not fully reflect real-world conditions, making it difficult to assess how well current systems would perform once deployed. Addressing these challenges is essential for advancing deepfake detection toward practical and trustworthy

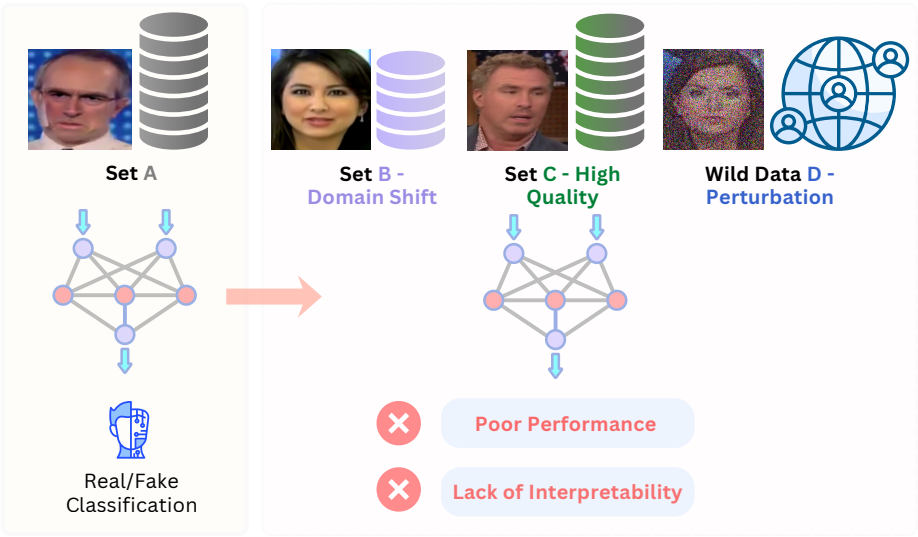


Figure 1.2 Challenges in deepfake detection: *Left)* Training Phase: a vanilla single-branch DNN is trained on a fixed dataset under a supervised manner, leading to overfitting to specific artifacts; *Right)* Testing Phase: the trained model, which is evaluated on several unseen datasets, exhibits limited generalization performance, lack of robustness to HQ and distorted deepfakes and limited interpretability.

applications. In this section, we present the current progress of research works and discuss their limitations and the key challenges to be addressed.

1.2.1 Generalization yet Robustness to Unseen High-Quality Deepfake Images

Deepfake generation techniques are advancing rapidly. New methods appear every day, continuously producing new and increasingly hyper-realistic synthetic content. This progress makes generated faces dramatically varied, not only in terms of appearance but also in their underlying data distributions. As a result, a common phenomenon known as distribution shift (or domain shift) has become one of the most critical challenges in deepfake detection [25]. A detector can quickly become obsolete if it fails to identify novel types of deepfakes. Therefore, a reliable detector is expected to generalize effectively across a wide range of unseen manipulations that are not available during training. Since it is impossible to access all possible types of deepfakes beforehand, designing a generalizable deepfake detector remains a central research question. In addition to generalization, detecting HQ deepfakes presents another significant difficulty [39]. Such realistic forgeries often contain subtle and localized artifacts that are imperceptible to the human eye (Figure 1.3), making them extremely challenging to capture even with advanced neural models.

Previous works [23, 24, 40, 41, 42] mostly frame deepfake detection as a supervised binary classification task. This straightforward strategy causes overfitting issues, hindering

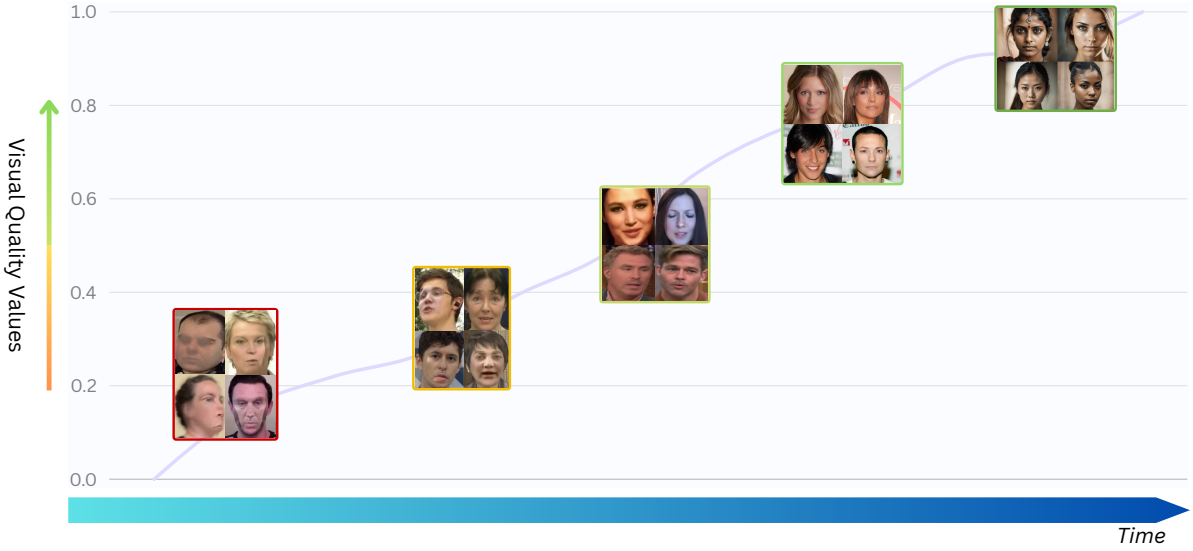


Figure 1.3 Evolution in deepfake visual quality. The quality of deepfakes is advancing surprisingly, characterizing very subtle and localized artifacts.

generalizability to novel manipulations. To mitigate generalization issues, various strategies have been explored in the research community, such as data synthesis [26, 25, 31, 43] and/or multi-task learning [27, 32, 44]. While these approaches have demonstrated improved generalization, they still lack robustness against HQ deepfakes. In fact, the majority rely on standard backbone architectures such as EfficientNet [45], XceptionNet [46], or ResNet [47], where successive convolutional operations tend to dilute fine-grained features that are crucial for identifying localized artifacts.

To address the challenge of HQ deepfakes, a few research studies have incorporated attention mechanisms to better capture subtle inconsistencies through low-level representations [39, 48]. However, these attention modules are often implicitly driven, without any guarantee of extracting true localized artifacts at a fine-grained level. Moreover, they do not incorporate complementary strategies to improve generalization while solely making use of single binary classifiers trained on real and fake data; consequently, they exhibit limited generalizability to unseen forgeries. Striving for both strong generalization and robustness to HQ forgeries simultaneously thus remains an open problem.

1.2.2 Generalization yet Robustness to Unseen High-Quality Deepfake Videos

The challenge becomes even more severe when detecting deepfake videos due to the complexity of modeling spatio-temporal artifacts, since spatial and temporal inconsistencies are often intertwined [3, 4, 49]. Most previous studies [50, 51, 52, 53, 54, 55, 56] adopt a single-branch deep neural network trained on both real and fake sequences. Such a strategy

tends to make the model overfit not only to the types of deepfake used in training but also to one dominant type of artifact, either spatial or temporal [52, 56].

Moreover, the quality of deepfake videos continues to improve, producing subtle spatio-temporal artifacts that are difficult to capture. As a result, standard single-branch architectures trained solely with binary supervision often fail to detect these fine-grained inconsistencies, highlighting the need for more effective attention mechanisms.

To mitigate the generalization issue, some studies have explored video-level data synthesis approaches [57, 29, 58] to encourage models to learn more generic representations. However, these methods often simulate exaggerated temporal variations that differ substantially from the subtle artifacts inherent in realistic deepfake videos. On the other hand, to model localized spatio-temporal inconsistencies, a few recent works [53, 59] have proposed dedicated architectures that integrate implicit attention mechanisms. Nevertheless, these models still rely on binary classifiers and provide no guarantee of effectively capturing fine-grained, artifact-prone traces.

In summary, detecting HQ deepfake videos remains a complex and unsolved problem due to the intricate nature of spatio-temporal artifacts. Beyond these technical challenges, detectors are also required to address broader aspects of practical reliability, including robustness and interpretability, to move closer to real-world standards.

1.2.3 Robustness to Unseen Common Perturbations, Flexibility, Scalability, Efficiency, and Interpretability

Even though deepfake detection has achieved great progress in research, current detectors are still far from being practical and trustworthy for real-world use. In practice, several important aspects need to be considered to ensure both reliability and usability of deployed systems.

Robustness to Unseen Common Perturbations: Deepfakes are widely shared across social media platforms, where they can be easily modified by common processing techniques such as compression, noise (*e.g.*, Gaussian or Block-wise), and Blur [54, 60]. Malicious users may intentionally apply these perturbations to bypass detectors. Therefore, a reliable system should remain stable and robust against unseen perturbations, ensuring consistent performance under different real-world conditions.

Flexibility, Scalability, and Efficiency: Most previous works [27, 54, 61, 56, 26, 33, 31, 62, 63, 64, 44, 65, 66, 67] have been built upon Convolutional Neural Networks (CNNs), which are effective in extracting local features but limited in capturing global relationships. Inspired by the success of Vision Transformers (ViTs) [68, 69] in various vision tasks [68, 70, 71, 72,

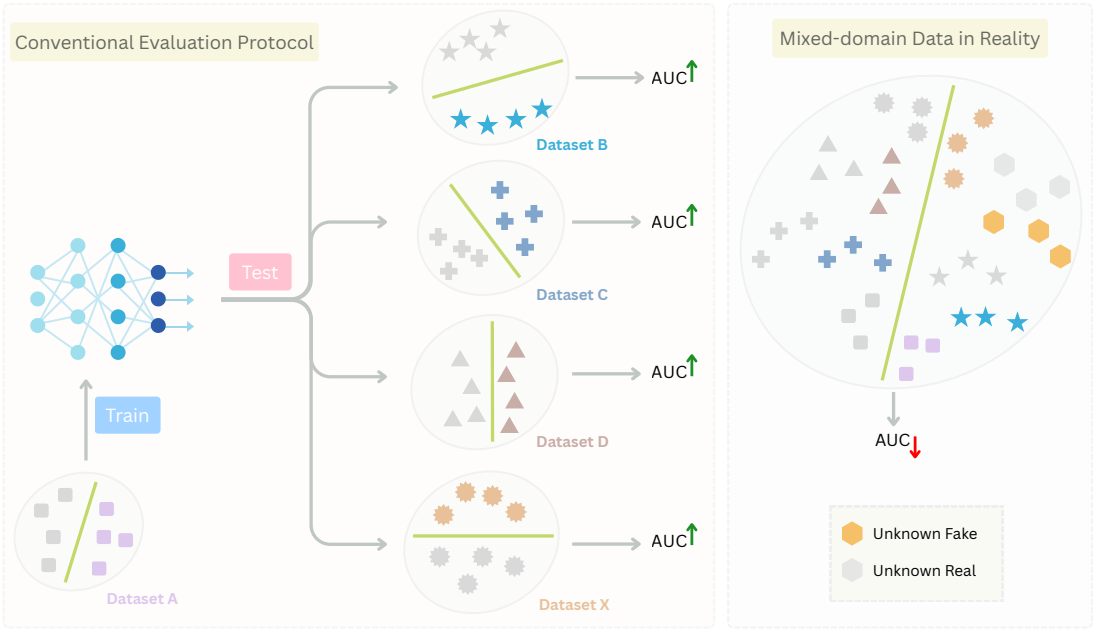


Figure 1.4 Conventional evaluation protocol evaluates on isolated benchmarks. This practice does not fully reflect the unknown mixed-domain data in a real-world scenario.

69], several transformer-based or hybrid CNN-ViT approaches [73, 35, 34, 36, 53, 28, 30, 38] have been introduced for deepfake detection. While these models are promising, they often require large network capacities [74, 75] and incur high computational costs and latency [76], which hinder real-world deployment. Designing an efficient and scalable architecture that can maintain strong performance while being lightweight and adaptable remains a key challenge.

Interpretability: Most existing deepfake detectors still provide only binary predictions (real or fake), offering little insight into why a particular decision is made [38, 77]. However, explainability is crucial to understanding how the network perceives and reacts to data, especially for improving model reliability and trust. Interpretability also enables post-analysis for examining failure cases, identifying weaknesses, and guiding model refinement. Enhancing explainability is, therefore, a fundamental step toward building trustworthy and transparent detection systems.

While improving these aspects is crucial for making deepfake detectors more practical and trustworthy, the way such systems are evaluated also plays an equally important role. In fact, existing evaluation protocols still fail to fully assess them under a real-world setting, which is discussed in the following section.

1.2.4 Realistic Evaluation

To assess the performance of deepfake detectors, two common evaluation protocols are typically used: **in-dataset evaluation** and **cross-dataset evaluation** [26, 31, 25, 27, 33]. The

former tests the learning capability of a trained detector on a test set that shares a similar distribution with the training data, while the latter evaluates its generalization capabilities across several unseen datasets that are not observed during training. Given its robustness to imbalanced data, the Area Under the Receiver Operating Characteristic (ROC) Curve (AUC) is then reported on each testing dataset separately. The average performance across these test sets is often used to approximate real-world performance and indicate how well a model might perform once deployed.

However, the conventional cross-dataset evaluation implicitly assumes isotropic distribution shifts, which do not fully reflect real-world conditions. In practice, data encountered in the wild often come from unknown heterogeneous and mixed sources, presenting anisotropic domain shifts that differ significantly from controlled test settings, as shown in Figure 1.4. As such, averaging performance across isolated datasets in the current protocol potentially obscures the relevance of a given deepfake detector in a practical scenario. The AUC measure, which estimates the class separability of a binary classification model, can be high when evaluated on each dataset individually. However, it can drop drastically when considering two datasets incorporating a domain shift. Therefore, establishing a more realistic evaluation protocol that better captures robustness, stability, and reliability under real-world conditions is both necessary and essential for advancing deepfake detection research.

In summary, these challenges emphasize the need for moving beyond laboratory settings toward methods that are both effective and reliable in real-world scenarios. Addressing them forms the core motivation of this thesis and guides the design of the proposed deepfake detection frameworks.

1.3 Objectives and Contributions

High-Quality (HQ) deepfake detection is an extremely challenging task. A successfully developed detector stems from the integration of several crucial elements. First, (**i**) **Domain Knowledge**: understanding how deepfakes are generated is essential for designing effective detectors that can target specific manipulation traces and recognize subtle synthetic patterns. (**ii**) **Dedicated Architectural Design**: The structure of a detector fundamentally determines what the model is able to perceive and learn. A well-conceived design enables the network to focus on informative cues while minimizing irrelevant patterns, thereby improving its ability to distinguish real from fake content. (**iii**) **Practical Applicability**: Beyond achieving high performance, practical applicability is a fundamental criterion to ensure that the developed methods are truly useful and reliable in real-world conditions. These criteria offer rich

directions for advancing the research topic.

Building upon these key elements, this thesis aims to address the main challenges identified in section 1.2 by proposing unified solutions for both image-level and video-level scenarios, as well as a more realistic evaluation protocol that better reflects practical deployment. Specifically, the objectives and corresponding contributions are the following:

1.3.1 A CNN-based Self-supervised Multi-task Learning Framework for Quality-Agnostic and Generalizable Deepfake Image Detection

Inspired by [39, 63, 78], our first framework, namely "LAA-Net", is built using a CNN architecture. Our goals are *threefold*: addressing the detection of HQ deepfakes, at the same time, improving the generalization performance and trustworthiness. Therefore, instead of employing a vanilla binary model that is trained to learn a single task only, we argue that these objectives can be achieved by designing an attention module compatible with generic deepfake detection strategies. Specifically, the solution would be to introduce an explicit fine-grained mechanism within a multi-task learning framework supported by an appropriate pseudo-fake synthesis technique. Moreover, in addition to such a learning strategy, we posit that an adequate architecture preserving low-level context features could implicitly contribute to better capturing localized artifacts.

More concretely, this work proposes a novel fine-grained approach called "Localized Artifact Attention Network (LAA-Net)" that relies on a self-supervised multi-task learning framework. First, a new fine-grained mechanism that aims at focusing on small artifact-prone regions centered at the vulnerable pixels is introduced. By vulnerable pixels, we mean the pixels that are more likely to showcase artifacts. This is achieved by considering two auxiliary branches, namely, a heatmap branch and a self-consistency branch. Second, the proposed architecture incorporates a novel, simple, yet effective Feature Pyramid Network (FPN) termed Enhanced FPN (E-FPN). It enables making use of multi-scale features while avoiding redundancy. The association of these two complementary components makes LAA-Net a suitable candidate for fine-grained and generic deepfake detection.

Our approach achieves better and more stable Area Under the Curve (AUC) performance as compared to existing methods [39, 26, 23, 32, 27], regardless of the quality of deepfakes, quantified using the Mask Structural SIMilarity (Mask-SSIM) [79]. For a more comprehensive evaluation, in addition to the standard AUC, other metrics are reported, namely, Average Precision (AP). We report experiments on several deepfake benchmarks [80, 81, 82, 83, 23] and show

that LAA-Net outperforms state-of-the-art (SOTA) methods. This work has been published at CVPR'2024 [62] and received the best paper award at the Research Day'2024 organized by the Faculty of Science, Technology, and Medicine at the University of Luxembourg.

1.3.2 An Efficient Transformer-based Self-supervised Learning Framework for Quality-Agnostic and Generalizable Deepfake Image Detection

In this work, our main goals are to enhance the performance of the previous work, LAA-Net, while strengthening its practicability for deployment. LAA-Net can effectively capture localized cues; nevertheless, its convolutional nature limits long-range reasoning across facial regions. Inspired by the flexibility and scalability of ViTs [68, 69], we extend our framework to Transformer architectures, introducing "LAA-Former", which combines global context modeling with explicit local attention. Transformers excel at capturing global dependencies but often overlook fine-grained artifacts due to their broad receptive fields and patch-level abstraction [84, 85, 86, 87]. Specifically, we design a lightweight "Learning-based Local Attention (L2-Att)" module that generalizes vulnerability modeling from pixels to patches, enabling Transformers to explicitly focus on artifact-sensitive regions while preserving their strength in global reasoning. In addition to the ViT-based LAA-Former, we further instantiate a Swin Transformer variant [85], termed "LAA-Swin", which incorporates the same vulnerability-aware attention mechanism within a hierarchical window-based architecture, demonstrating the flexibility and scalability of our design across different Transformer variants. Our experimental study shows that generalizing this approach to transformers contributes to: **1) Enhanced generalization performance**; **2) Friendly computational cost and reduced model size**; and **3) Mitigating the need for large-scale datasets**. This is confirmed by extensive experiments conducted on several challenging datasets [80, 23, 83, 82, 88, 81], showing that our method outperforms SOTA approaches, including both CNN-based and ViT-based methods, while relying on a significantly smaller training set [37]. This work is an extension of LAA-Net, namely LAA-X, and will be submitted to TPAMI.

1.3.3 A Self-supervised Multi-task Learning Framework for Generalizable and Interpretable Deepfake Video Detection

This work aims to extend the key challenges of image-based deepfake detection to the video-level, which requires more complex modeling of spatio-temporal features. Inspired by LAA-Net and LAA-Former for image-based deepfake detection, it has been demonstrated that

the use of a tailored multi-task learning framework for explicitly attending artifact-prone small regions coupled with a subtle data synthesis strategy can be a way to enhance generalization and, at the same time, robustness to HQ deepfakes. Nevertheless, its extension to the video level is not straightforward. In particular, it would necessitate the characterization of subtle temporal artifacts that are inherently different from spatial ones, within both the multi-task learning framework and the data synthesis.

In this work, we redefine deepfake video detection as a fine-grained detection task by proposing a multi-branch network that leverages synthesized data and incorporates specialized learning objectives specifically targeting both subtle spatial and temporal artifacts. Specifically, a novel multi-task learning framework, termed "FakeSTormer" is introduced. It is formed by two auxiliary parallel branches in addition to the standard classification head, namely: (1) a regression temporal branch incorporating an explicit attention that aims at locating the vulnerability-prone temporal locations; and (2) a spatial branch to ensure a balance between the spatial and the temporal domains. These proposed branches help the model focus on vulnerable regions and provide more valuable insights into how the network *sees* the data while offering more robustness to high-quality deepfakes.

To create hand-free ground truths for the proposed branches, we introduce a HQ video-level data synthesis algorithm, called "Self-Blended Video (SBV)", inspired by "Self-Blended Image (SBI) [26]", enforcing temporal coherence using two proposed modules on top of SBI. Our experiments demonstrate that simply training a baseline classification model on SBV enables achieving on par performance w.r.t. SOTA, highlighting the effectiveness of SBV. Finally, for enhancing spatial and temporal modeling, we revisit the TimeSformer [89] architecture that we use as our backbone. In particular, we leverage TimeSformer's decomposed temporal and spatial attention on embedded patches, appending classification tokens for each frame and for each patch across frames, rather than a single token for the entire video. These classification tokens are then used within the spatial and classification heads, while the embedded patches are used within the temporal head. Extensive experiments on several well-known deepfake detection benchmarks [23, 80, 82, 83, 88, 81, 14, 13] show that our method outperforms the existing SOTA approaches. This work has been published at ICCV'25 [90].

1.3.4 A Novel Metric for more Realistic Evaluation under Domain Shifts

This work aims to address the limitations of conventional evaluation protocols for more precisely evaluating the generalization performance of deepfake detectors under realistic

conditions.

Specifically, we propose a new metric, called *Cross-dataset AUC (Cross-AUC)*, that can better assess the generalization capabilities of deep detectors under domain shift. It relies on the mean AUC on diverse datasets while taking into account the polarization extent of the probability predictions. A stronger polarization of probability predictions (predictions closer to 0 and 1) suggests that the classes are more distinctly separated within the score space, and a fixed threshold is more likely to remain effective under varied conditions, contributing to enhanced generalization capabilities. The polarization extent is estimated using the Wasserstein Distance (WD) between the distribution of positive and negative probability predictions. Consequently, the proposed Cross-AUC captures both the relative ranking performance (via the mean AUC) and the absolute separability of predictions (via the polarization), offering a more adequate estimate of generalization performance in real-world scenarios. Finally, we empirically show that the Cross-AUC is very close to the AUC measured on a combination of all the considered datasets, making it a practical measure for evaluating cross-dataset generalization. This work is under review at ECCV'26.

1.4 Publications

JOURNAL

1. **Nguyen, D.**, Astrid, M., Kacem, A., Ghorbel, E., Aouada, D. (2025). "LAA-X: Unified Localized Artifact Attention for Quality-Agnostic and Generalizable Face Forgery Detection". The IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (Under Review).

CONFERENCES

1. **Nguyen, D.**, Mejri, N., Singh, I. P., Kuleshova, P., Astrid, M., Kacem, A., Ghorbel, E., Aouada, D. (2024). "Laa-net: Localized artifact attention network for quality-agnostic and generalizable deepfake detection". In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 17395-17405).

2. **Nguyen, D.**, Astrid, M., Kacem, A., Ghorbel, E., Aouada, D. (2025). "Vulnerability-Aware Spatio-Temporal Learning for Generalizable Deepfake Video Detection". In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (pp. 10786-10796).

3. **Nguyen, D.**, Cosmin, R., Hermary, R., Astrid, M., Mejri, N., Ghorbel, E., Aouada, D.

(2026). "AUC Meets Polarization: Toward a More Realistic Evaluation of Deepfake Detectors". The European Conference on Computer Vision (ECCV'26) (Under Review).

PUBLICATIONS NOT INCLUDED IN THIS THESIS

1. **Nguyen, D.**, Nguyen, T. S., Pham, H. A. T., Hoang, T. T., Thanh, T. M. (2023). "Hybrid end-to-end approach integrating online learning with face-identification system". Computer Science, 24(2).

2. Singh, I. P., Mejri, N., **Nguyen, D.**, Ghorbel, E., Aouada, D. (2023, September). "Multi-label deepfake classification". In 2023 IEEE 25th International Workshop on Multimedia Signal Processing (MMSP) (pp. 1-5). IEEE.

1.5 Thesis Outline

This dissertation is organized as follows:

- **Chapter 2:** introduces the background concepts underpinning this work. It reviews the principal deepfake creation pipeline, blending-based data synthesis for generating pseudo-fakes, multi-scale feature representations, transformer architectures for visual data, and the AUC metric used as the primary evaluation measure throughout the thesis.

- **Chapter 3:** presents LAA-Net, a CNN-based self-supervised multi-task framework for quality-agnostic and generalizable deepfake image detection. It describes the proposed localized artifact attention mechanism, the Enhanced Feature Pyramid Network (E-FPN), and provides extensive experiments and analyses on multiple benchmarks.

- **Chapter 4:** extends the vulnerability-aware design to transformer architectures by introducing LAA-Former and LAA-Swin, which incorporate a Learning-based Local Attention (L2-Att) module. This chapter analyzes the limitations of plain Vision Transformers for deepfake detection, details the proposed architectures, and evaluates their generalization, robustness, and efficiency.

- **Chapter 5:** addresses deepfake video detection with FakeSTormer, a vulnerability-aware spatio-temporal framework built upon a revisited TimeSformer backbone. It introduces the Self-Blended Video (SBV) data synthesis strategy, describes the multi-task temporal and spatial vulnerability branches, and reports cross-dataset experiments on several challenging video benchmarks.

- **Chapter 6:** investigates the limitations of conventional cross-dataset evaluation and proposes Cross-dataset AUC (Cross-AUC), a metric that combines mean AUC with a polarization measure of prediction scores under domain shift. It formulates the problem, defines the metric, and validates it through experiments across multiple datasets and detectors.
- **Chapter 7:** concludes the thesis by summarizing the main findings and contributions and discussing several directions for future research on interpretable, universal, and practically deployable deepfake detection.

Chapter 2

Background

This chapter introduces the background concepts that are directly related to this thesis. It first recalls the typical deepfake creation pipeline, highlighting the key stages through which realistic facial forgeries are synthesized and where manipulation artifacts arise. It then presents blending-based data synthesis techniques often used for generating pseudo-fakes and improving the generalizability of detectors. This chapter also discusses multi-scale feature learning for capturing both localized and subtle forgery cues, and reviews core components and different variants of transformer architectures for visual data. Finally, it presents the Area Under the ROC Curve (AUC) metric, which is adopted as the primary evaluation measure for deepfake detection throughout the thesis.

2.1 Principal Deepfake Creation Pipeline

Understanding the process of deepfake generation is essential for designing effective detection methods, as it can provide insights about where visual inconsistencies occur, which features are the most informative, and how a detector can exploit them. This section reviews the principal pipeline of deepfake creation [40, 43, 80] and outlines potential directions for developing robust and generalizable detection models.

The deepfake generation process typically follows a sequential pipeline as illustrated in Figure 2.1. It begins with face detection to localize the target region, followed by facial landmark extraction and face alignment to ensure consistent pose and scale between source and target faces. The aligned face is then encoded into a compact latent representation using an encoder, and reconstructed through a decoder to produce a synthesized face. During training, two autoencoders are usually employed, including one for the source and one for the target, sharing a common encoder but using different decoders. The reconstruction loss

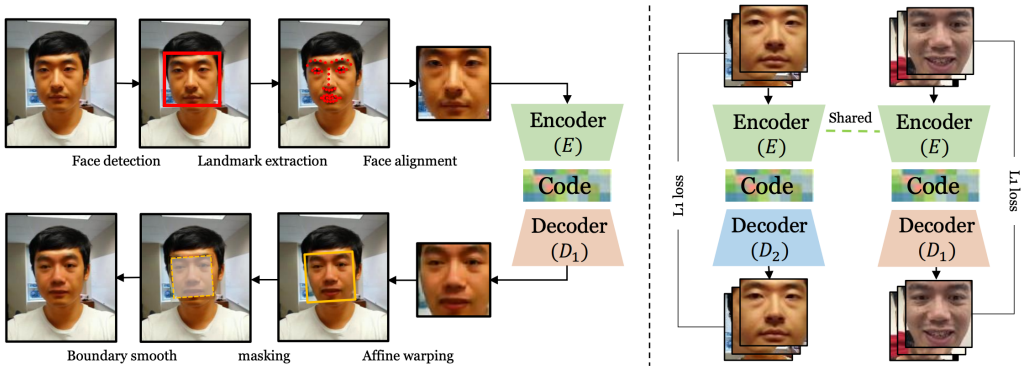


Figure 2.1 The principal pipeline of a basic deepfake creation algorithm. *Left*: Synthesis process; *Right*: Training process. The figure is cited from [80].

(often L_1) ensures visual fidelity and identity consistency. In the final stage, the synthesized face is affine-warped, masked, and blended into the original frame using boundary smoothing to achieve a natural composite result.

Throughout this process, artifacts can arise from several sources. Typical ones include boundary artifacts caused by imperfect blending or warping, color and illumination inconsistencies between the synthesized and original regions, and texture distortions introduced by the encoder-decoder reconstruction. In addition, geometric misalignment in facial features (*e.g.*, eyes or mouth) and subtle temporal inconsistencies across frames in videos are also common. These localized and often imperceptible traces provide valuable cues for detection. Therefore, developing a detector that can focus on these artifact regions can improve generalization and robustness across diverse manipulation techniques.

2.2 Data Synthesis

Generalization is a major challenge in deepfake detection, as real-world forgeries often arise from manipulation techniques that differ significantly from those seen during training. One practical way to alleviate this issue is to generate additional pseudo-fake samples that expose detectors to more diverse and generic artifact patterns. Data synthesis [43, 91], therefore, plays an important role, as it allows models to learn from artifacts that naturally arise during the synthesis process, without relying on specific deepfake generation methods.

In this thesis, data synthesis is used as an auxiliary tool to support the development of more generalizable detectors. In particular, we consider two synthesis-related components: *1) Blending-based data synthesis algorithms*, which generate pseudo-fake samples by swapping and blending facial regions, and *2) Blending boundaries*, a concept that focuses on the transitional regions where blending artifacts are likely to appear. The following subsections describe these two components in more detail.

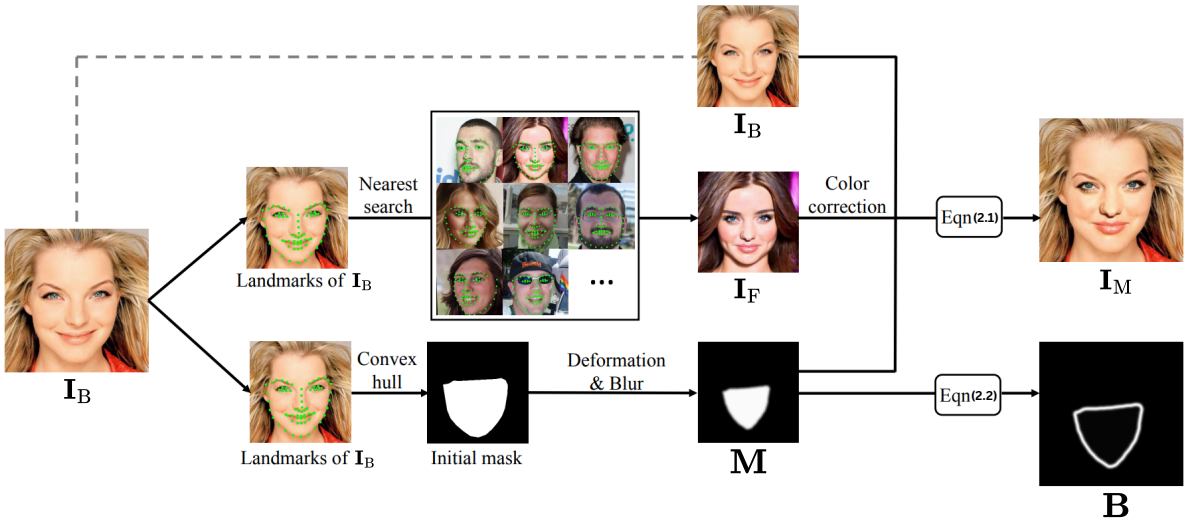


Figure 2.2 Overview of the generation process of a blending-based pseudo-fake. Given a real image $\mathbf{I}_B$, nearest search is applied to find another real image $\mathbf{I}_F$ based on a landmark similarity criterion. The $\mathbf{I}_F$ represents the manipulated variant of $\mathbf{I}_B$ and is used to generate a mask depicting the manipulated region for the synthesized process equation (2.1). The figure is adapted from [25].

2.2.1 Blending-based Pseudo-fake

Blending-based pseudo-fakes are synthesized by combining facial regions from two real images through an explicit blending operation [25, 33, 26, 31]. This process mimics the composition step that appears, either explicitly or implicitly, in many face manipulation pipelines [12]. Given a foreground image $\mathbf{I}_F$, a background image $\mathbf{I}_B$, and a blending mask $\mathbf{M} \in [0,1]^{H \times W}$, a pseudo-fake image $\mathbf{I}_M$ is generated as:

$$\mathbf{I}_M = \mathbf{M} \odot \mathbf{I}_F + (1 - \mathbf{M}) \odot \mathbf{I}_B \quad , \tag{2.1}$$

where H and W denote the height and width of the image and $\odot$ denotes element-wise multiplication. An illustration of this synthesis process is depicted in Figure 2.2.

The mask $\mathbf{M}$ typically defines the facial region to be inserted, while the soft transition between $\mathbf{M}$ and $(1 - \mathbf{M})$ creates blended areas that resemble the artifacts introduced during face swapping or reenactment. Although such images are not produced by an actual deepfake model, they still contain characteristic composition inconsistencies, such as mismatched appearance, illumination, or texture [26]. These artifacts arise from the blending operation itself and not from any specific generative architecture. As such, blending-based pseudo-fakes provide a simple yet effective way to introduce generic manipulation cues, which can later be used to improve the generalization ability of deepfake detectors in the next chapters.

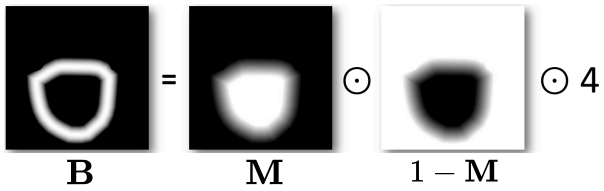


Figure 2.3 Extraction of a blending boundary.

2.2.2 Blending Boundary

Blending boundaries [25] correspond to the transitional regions where the foreground and background meet during the composition process. As a result, blending boundaries represent one of the most artifact-prone regions in both synthetic pseudo-fakes and real manipulated images.

Given a blending mask $\mathbf{M} \in [0,1]^{H \times W}$, the boundary map $\mathbf{B}$ (Figure 2.3) can be approximated by:

$$\mathbf{B} = 4 \cdot \mathbf{M} \odot (1 - \mathbf{M}) \quad , \tag{2.2}$$

where high values indicate pixels positioned near the transition zone between the swapped facial region and the background. This simple formulation highlights locations that are more likely to contain manipulation cues, independent of the specific generation method used.

Because blending boundaries capture structural inconsistencies that remain relatively stable across different manipulation techniques, they provide a useful clue for designing detectors that focus on artifact-prone regions. Subsequent chapters 3, 4, and 5 leverage this concept in different forms to guide both spatial and spatio-temporal deepfake detection models.

2.3 Multi-scale Features

In practice, deepfake artifacts arise at different levels of visual abstraction, from fine-grained texture irregularities to broader geometric or semantic inconsistencies. These heterogeneous cues cannot be fully captured at a single spatial resolution, making multi-scale feature representations an essential component of effective deepfake detection. Multi-scale processing enables a model to integrate complementary information across resolutions, strengthening its ability to identify diverse manipulation traces. This section introduces the connection between deepfake artifacts and multi-scale cues, followed by an overview of how multi-scale features are extracted in CNN architectures and formalized in feature-pyramid designs.

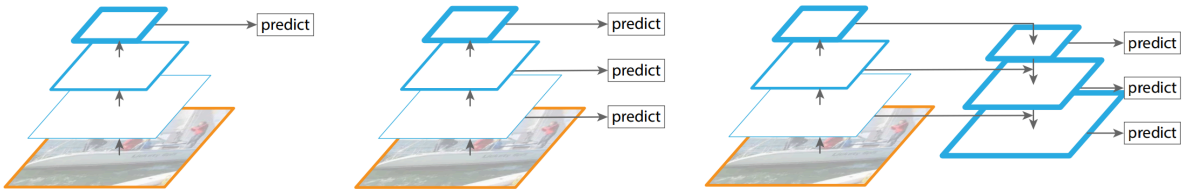


Figure 2.4 Common paradigms for extracting multi-scale feature representations. a) A single-scale feature map at the final layer is commonly used for predictions; b) Multi-scale feature maps are used to consider multi-level abstract patterns at higher-resolution layers, improving the performance; c) Feature pyramid network enhances the representation of both texture and semantic features by aggregating multiple layers in a top-down manner, thereby increasing the accuracy.

2.3.1 Interconnection between Deepfake Artifacts and Multi-scale Features

As mentioned earlier, deepfake manipulation traces appear across different spatial scales. This should therefore directly influence the choice of the feature representation. Local artifacts, such as blending, texture or color inconsistencies, typically reside at fine spatial resolutions, whereas distortions of facial geometry or inconsistencies in global appearance emerge at coarse scales. Since these artifacts arise from distinct stages of the manipulation pipeline, they exhibit complementary characteristics that no single-resolution representation can fully capture. Modeling the multi-scale nature of deepfake artifacts is therefore essential for integrating both localized and structural cues.

2.3.2 CNN-based Multi-scale Features

Convolutional Neural Networks [47, 46, 45] have become the dominant backbone in deepfake detection due to their strong capability in modeling visual patterns and capturing manipulation artifacts. Their effectiveness is mainly due to their hierarchical feature structure [92] as illustrated in Figure 2.4-b. Early layers operate at high spatial resolution and capture fine-grained cues such as edges, textures, and localized inconsistencies. In contrast, deeper layers progressively accumulate information over larger receptive fields through the stacking of convolutional and downsampling operations. This progression enables the network to encode increasingly abstract patterns, ranging from local appearance to higher-level structural cues. Formally, given an input image $\mathbf{X}$, a CNN backbone generates a sequence of feature maps $\{\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_L\}$, where each representation is obtained through

$$\mathbf{F}_\ell = \phi_\ell(\mathbf{F}_{\ell-1}), \quad \mathbf{F}_0 = \mathbf{X},$$

and ϕ_ℓ denotes the convolutional transformation at stage ℓ, typically composed of convolution, non-linearity, and optional downsampling. This formulation highlights the natural emergence of a feature hierarchy that encodes visual information at multiple spatial scales.

Despite this, most deepfake detection models [23, 24, 93] leave this structure underexplored. They mainly rely on standard backbones and formulate detection as a single binary decision made from the final-layer feature map (Figure 2.4-a), thereby discarding rich intermediate cues that encode diverse manipulation traces. To the best of our knowledge, only CADDM [27] has explored the integration of information across multiple depths, demonstrating that it can strengthen the detection capabilities.

2.3.3 Feature Pyramid Networks (FPNs)

Feature Pyramid Networks (FPNs) [94] were introduced to avoid relying solely on the deepest representation in CNNs. Instead of discarding intermediate feature maps, FPNs are constructed in a top-down manner by propagating high-level semantic information to earlier layers and combining them with high-resolution spatial details through lateral connections.

Formally, let $\{\mathbf{F}_1, \ldots, \mathbf{F}_L\}$ denote the backbone feature hierarchy, where deeper layers have lower spatial resolution but higher semantic abstraction. A traditional FPN (Figure 2.4-c) generates a corresponding pyramid $\{\mathbf{P}_1, \ldots, \mathbf{P}_L\}$ through a top-down pathway and lateral fusion:

$$\mathbf{P}_L = \mathcal{W}_L(\mathbf{F}_L), \tag{2.3}$$

$$\mathbf{P}_\ell = \mathcal{W}_\ell(\mathbf{F}_\ell) + \mathrm{Upsample}(\mathbf{P}_{\ell+1}), \quad \ell = L-1, \ldots, 1, \tag{2.4}$$

where $\mathcal{W}_\ell(\cdot)$ denotes a lateral (1×1) convolution used to align channel dimensions, and $\mathrm{Upsample}(\cdot)$ is typically a factor-of-two nearest-neighbor interpolation or a learnable operator. This construction yields multi-scale features $\mathbf{P}_\ell$ that integrate coarse semantic context from deeper layers with fine-grained spatial structure from shallower ones.

The ability of FPNs to unify information across scales has proven crucial in tasks where meaningful cues may appear at different levels of granularity. The effectiveness of FPNs has been repeatedly validated across numerous computer vision problems [95, 94, 96, 97].

2.4 Transformer Architectures

Transformers [68, 69, 98, 84, 85, 86] have reshaped modern computer vision by introducing a fundamentally different mechanism for modeling visual information. Rather than relying

on the locality and hierarchical aggregation inherent to CNNs, transformers employ self-attention (SA) [99, 68] to model global relationships across image or video tokens, often outperforming CNNs in several computer vision tasks [72, 98, 69, 70, 68, 100].

To provide the necessary background for the transformer-based frameworks developed later in this thesis, this section recalls the key components that form visual transformers. We begin with the tokenization and embedding process that converts images into sequences, followed by positional encoding and the SA mechanism. We then discuss representative spatial architectures, including the plain Vision Transformer (ViT) [68, 69] and the hierarchical Swin Transformer [85], as well as transformer variants extended to spatio-temporal data.

2.4.1 Foundations of Vision Transformers

Originally designed for natural language processing (NLP) [99], transformers operate on sequences of tokens and rely on self-attention (SA) to compute global interactions. Extending this paradigm to images requires representing visual data as a sequence, enabling long-range relationships to be modeled in a uniform framework.

Unlike CNNs which encode locality via fixed receptive fields, transformers enable token-to-token interactions. While this global connectivity is advantageous for capturing broad contextual structure, spatial locality remains essential for many visual tasks. As a result, modern Vision Transformers often incorporate architectural mechanisms such as patch embeddings [68, 69], hierarchical representations [86, 101], or window-based attention [84, 85] to reintroduce spatial bias while preserving the flexibility of attention.

A key implication of this design is that the same sequence-based formulation naturally generalizes beyond static images. Video Transformers such as TimeSformer [89] extend the representation to the temporal dimension by factorizing spatial and temporal attention. Thus, architectures for both images and videos derive from the same foundational principles, namely *token embeddings*, *positional encodings*, and the *self-attention mechanism*.

Input Embeddings

Transformers operate on sequences of fixed-dimensional vectors [99]. For visual data, this requires converting an image $\mathbf{X} \in \mathbb{R}^{C \times H \times W}$ into a structured sequence of token embeddings. This patch-based reparameterization provides a structured token sequence.

Visual Tokenization. The image is partitioned into non-overlapping patches of size $P \times P$, producing $N = \frac{HW}{P^2}$ patches. Each patch is flattened into a vector $\mathbf{x}_i \in \mathbb{R}^{1 \times (C \cdot P^2)}$ and

projected into a D-dimensional embedding using a learnable mapping matrix $\mathbf{E}$:

$$\mathbf{z}_i^0 = \mathbf{x}_i \mathbf{E}, \qquad \mathbf{E} \in \mathbb{R}^{(P^2) \times D}. \tag{2.5}$$

Stacking all projected tokens yields

$$\mathbf{z} = [\mathbf{z}_1^0; \mathbf{z}_2^0; \ldots; \mathbf{z}_N^0] = [\mathbf{x}_1\mathbf{E}; \mathbf{x}_2\mathbf{E}; \ldots; \mathbf{x}_N\mathbf{E}], \quad \mathbf{z} \in \mathbb{R}^{N \times D}. \tag{2.6}$$

Class Token. A learnable class token $\mathbf{x}_{\text{cls}} \in \mathbb{R}^{1 \times D}$ is prepended at the zero-index of the token sequence $\mathbf{z}$ for the classification:

$$\mathbf{z}' = [\mathbf{x}_{\text{cls}}; \mathbf{x}_1\mathbf{E}; \mathbf{x}_2\mathbf{E}; \ldots; \mathbf{x}_N\mathbf{E}], \quad \mathbf{z}' \in \mathbb{R}^{(N+1) \times D}. \tag{2.7}$$

Through the SA, this token interacts and aggregates information across all patch embeddings, providing a global summary that may capture holistic artifacts introduced during face synthesis or blending.

Positional Encodings. Since patchification removes the spatial ordering inherent in the original image, a positional encoding matrix $\mathbf{E}^{\text{pos}} \in \mathbb{R}^{(N+1) \times D}$ is added to the tokens $\mathbf{z}'$:

$$\mathbf{z}^0 = \mathbf{z}' + \mathbf{E}^{\text{pos}}. \tag{2.8}$$

where $\mathbf{z}_0 \in \mathbb{R}^{(N+1) \times D}$. These encodings restore spatial awareness, ensuring that the model can reason about both local arrangements and long-range dependencies.

Self-Attention Mechanism

Self-Attention (SA) is the central operation enabling transformers to compute interactions across all tokens in a sequence. Given an embedded input $\mathbf{z}^0 \in \mathbb{R}^{(N+1) \times D}$, SA forms token representations by aggregating information based on learned pairwise relationships.

Query, Key, and Value Projections. Each token is transformed into query, key, and value vectors through linear projections:

$$\mathbf{Q} = \mathbf{z}^0 \mathbf{w}_Q, \qquad \mathbf{K} = \mathbf{z}^0 \mathbf{w}_K, \qquad \mathbf{V} = \mathbf{z}^0 \mathbf{w}_V, \tag{2.9}$$

where $\mathbf{w}_Q, \mathbf{w}_K, \mathbf{w}_V \in \mathbb{R}^{D \times D_h}$ are learnable matrices, and D_h is the head dimension.

- **Queries (Q)** represent what each token is looking for.
- **Keys (K)** represent what each token contains.

- **Values (V)** represent the actual information to be aggregated.

Thus, $\mathbf{Q}$ and $\mathbf{K}$ determine how much one token should attend to another, while $\mathbf{V}$ determines what information is transferred.

Attention Weights. Interactions between tokens are computed by comparing queries with keys:

$$\mathbf{A} = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D_h}}\right), \qquad \mathbf{A} \in \mathbb{R}^{(N+1)\times(N+1)}. \tag{2.10}$$

The matrix $\mathbf{A}$ contains pairwise attention scores with a high value indicating strong relevance, while a low value suggests weak contribution. Thus, $\mathbf{A}$ determines which tokens influence each other and to which extent.

Token Aggregation. The output of SA is obtained by weighting the values using the attention matrix:

$$\mathbf{H} = \mathbf{A}\mathbf{V}. \tag{2.11}$$

Each token becomes a weighted combination of information from all other tokens.

Multi-Head Self-Attention. Multi-Head Self-Attention (MHSA) operates using multiple SA heads in parallel, each with a distinct learned projection. This allows the model to capture multiple types of relationships simultaneously (*e.g.*, texture-level relations, structural relations, or lighting-level relations) as depicted in the following,

$$\mathrm{MHSA}(\mathbf{z}^0) = \mathrm{Concat}(\mathbf{H}_1, \ldots, \mathbf{H}_h)\mathbf{w}_O \tag{2.12}$$

where $\mathbf{w}_O \in \mathbb{R}^{(hD_h)\times D}$ is a learnable projection matrix. Consequently, MHSA enhances representational capacity by encouraging each head to focus on different complementary aspects of the input.

Transformer Encoder

Beyond the SA mechanism itself, the core computational unit used in ViTs is the transformer encoder block [68]. Given a sequence of token embeddings $\mathbf{z}^0 \in \mathbb{R}^{(N+1)\times D}$ (including the class token and N patch tokens), an encoder block iteratively refines these representations through two sublayers: an MHSA layer and a feed-forward network (FFN). In each block, the input is first normalized and processed by MHSA, and the result is added back to the input via a residual connection, *i.e.*,

$$\mathbf{z}^{(l-1)'} = \mathbf{z}^{l-1} + \mathrm{MHSA}(\mathrm{LN}(\mathbf{z}^{l-1})), \tag{2.13}$$

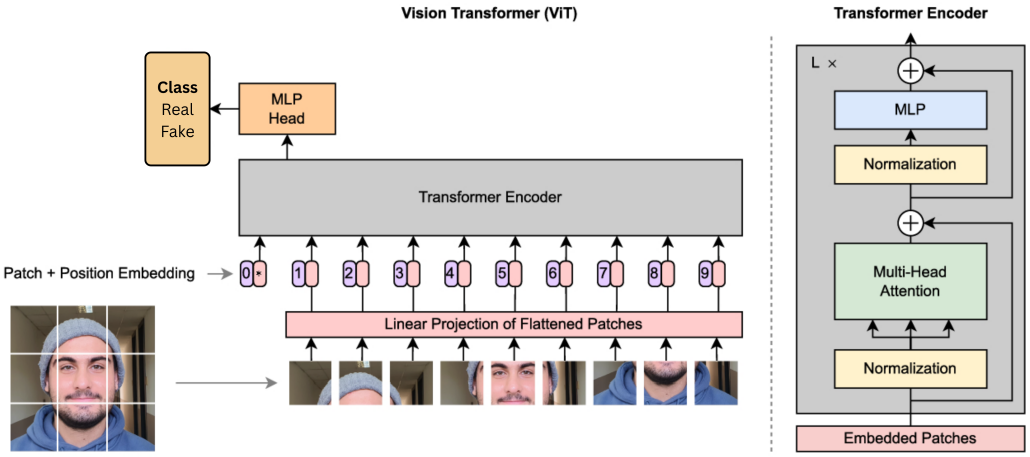


Figure 2.5 Overview of plain Vision Transformer (ViT). The figure is adapted from [102].

where $\mathbf{z}^{(l)}$ denotes the sequence at layer l and $\mathrm{LN}(\cdot)$ is a layer normalization. A second sublayer then applies an FFN, typically implemented as a two-layer MLP with a non-linearity, again wrapped with normalization and a residual connection,

$$\mathbf{z}^{l} = \mathbf{z}^{(l-1)'} + \mathrm{MLP}(\mathrm{LN}(\mathbf{z}^{(l-1)'})). \tag{2.14}$$

Stacking L encoder blocks yields a deep sequence model in which information can be iteratively exchanged across all tokens. In the standard ViT, the final embedding of the class token is used as a global image representation for classification, while the remaining patch tokens encode spatially distributed information. This generic encoder design is adapted to different architecture variants such as ViT [68], Swin [85], and TimeSformer [89], which will be reviewed in the following sections.

2.4.2 Vision Transformer Architectures

The introduction of the SA as a general-purpose building block has enabled the development of a family of transformer models designed specifically for visual data. Among these, the Vision Transformer (ViT) [68] serves as the canonical architecture, demonstrating that large-scale image recognition can be performed without relying on convolutional inductive biases. Building on this, several variants have been proposed to improve scalability, locality modeling, and computational efficiency. In this section, we first present the plain ViT formulation and subsequently discuss representative relevant extensions, namely, the Swin Transformer [85] and TimeSformer [89].

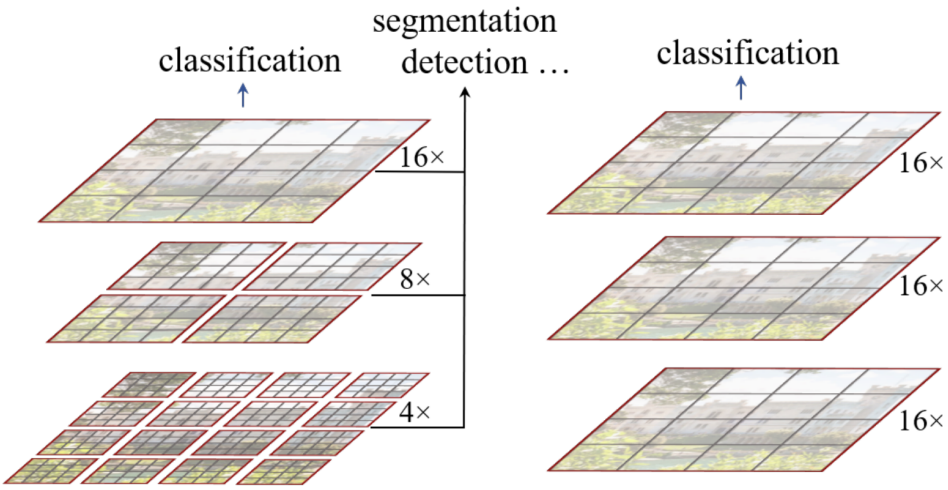


Figure 2.6 Swin Transformer (Left) versus Vision Transformer (Right).

Plain Vision Transformer

The Vision Transformer (ViT) [68] extends the Transformer encoder [99] to image analysis by treating an image as a sequence of patch embeddings. After tokenization and positional encoding, the sequence is processed through the succession of identical encoder blocks composed each of MHSA and a feed-forward subnetwork, both wrapped with residual connections and normalization layers. A learnable class token $\mathbf{x}_{\text{cls}}$ is appended to the sequence and serves as a global representation whose final embedding is used for classification. The overall framework of ViT is illustrated in Figure 2.5.

This architecture follows a flat, non-hierarchical design: all patches share the same resolution throughout the network, and attention is computed globally across all tokens. While this formulation demonstrates that attention-based models can match or surpass convolutional networks in image classification, its global attention mechanism leads to quadratic computational complexity and a lack of intrinsic locality modeling. These characteristics motivate several subsequent variants, including hierarchical designs such as the Swin Transformer [85] and factorized spatio-temporal formulations for video.

Swin Transformer

Swin Transformer (Swin) [85] was introduced to address two key limitations of plain ViTs [68, 69]: (i) the quadratic cost of global SA with respect to image size, and (ii) the lack of an intrinsic feature hierarchy for dense vision tasks. Swin replaces global attention with *window-based self-attention* and builds *hierarchical representations* through progressive patch merging, as shown in Figure 2.6. This design preserves the flexibility of attention while reintroducing locality and multi-scale structure, making Swin a strong general-purpose backbone in vision.

Window-based Self-Attention. Instead of attending to all tokens, Swin partitions the feature map into non-overlapping windows, and attention is computed within each window. This restricts computation to local neighborhoods, yielding near-linear complexity in image size while encouraging the modeling of fine-grained local patterns:

$$\mathbf{z}_w^l = \text{W-MHSA}(\text{LN}(\mathbf{z}_w^{l-1})) + \mathbf{z}_w^{l-1}. \tag{2.15}$$

Shifted Windows for Cross-Window Interaction. Pure W-MHSA lacks connections across windows. Swin resolves this by shifting the window partition between consecutive blocks. Standard W-MHSA alternates with *shifted window attention* (SW-MHSA), where windows are offset (typically by $\lfloor M/2 \rfloor$). This enables cross-window token communication without increasing complexity:

$$\mathbf{z}_w^{l'} = \text{SW-MHSA}(\text{LN}(\mathbf{z}_w^l)) + \mathbf{z}_w^l. \tag{2.16}$$

Through this alternating schema, Swin maintains locality while enabling the extraction of gradual global information flow.

Hierarchical Representation via Patch Merging. Swin is organized into multiple stages. Between stages, a *patch merging* layer reduces spatial resolution while increasing feature dimension. For every 2×2 group of patches, Swin concatenates their features and applies a linear projection:

$$N_{s+1} = \frac{N_s}{4}, \qquad D_{s+1} = 2D_s. \tag{2.17}$$

This results in a pyramid of feature maps with progressively coarser resolution and richer semantics, mirroring the multi-scale inductive bias of CNNs.

TimeSformer for Spatio-Temporal Modeling

TimeSformer [89] extends the ViT architecture to video by introducing a factorized spatio-temporal attention mechanism. Instead of applying full joint attention over all spatial patches across all frames, which is computationally expensive, TimeSformer (Figure 2.7) decomposes self-attention into separate spatial and temporal components. This enables the model to capture dynamic information across time while preserving the representational flexibility of ViT.

Spatial Attention. For each video frame, TimeSformer applies standard MHSA across spatial patches, allowing the model to encode intra-frame structure and appearance. Spatial attention is defined as,

$$\mathbf{z}_{\text{spatial}}^l = \text{MHSA}(\text{LN}(\mathbf{z}^{l-1})) + \mathbf{z}^{l-1}. \tag{2.18}$$

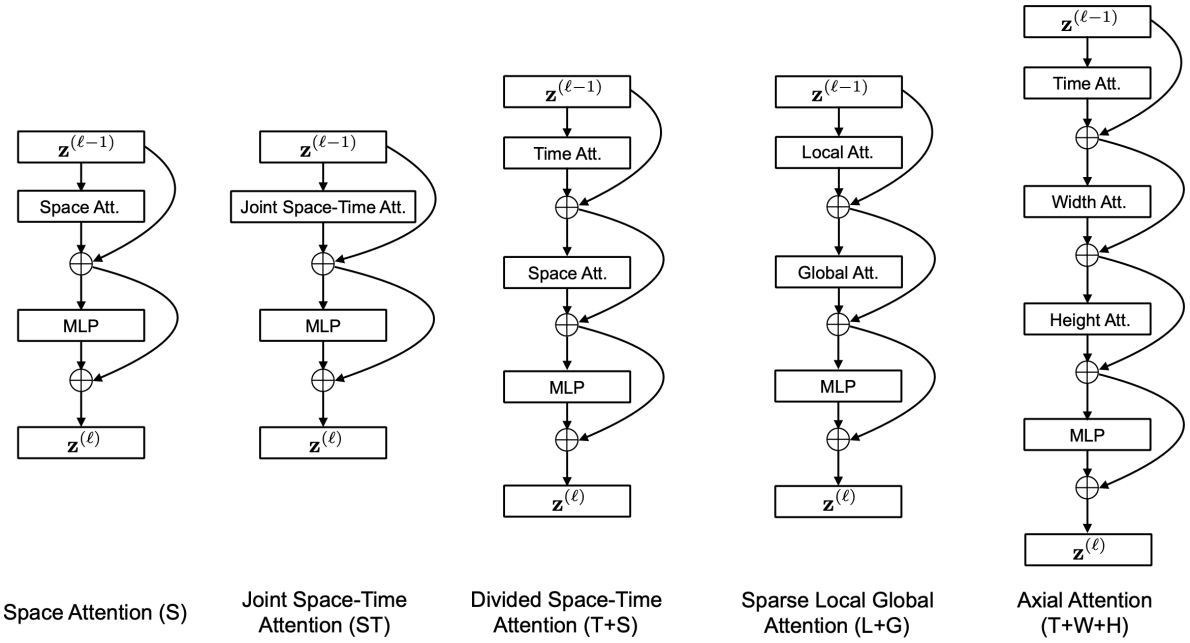


Figure 2.7 Different space-time self-attention schemes studied in Timesformer. In this thesis, we choose the divided Space-Time SA as it provides the best performance while remaining computationally efficient.

This captures fine-grained spatial patterns relevant to frame-level inconsistencies.

Temporal Attention. To incorporate motion and temporal coherence, TimeSformer applies SA across the temporal axis at each spatial location:

$$\mathbf{z}^{l}_{\text{temporal}} = \text{MHSA}(\text{LN}(\mathbf{z}^{l}_{\text{spatial}})) + \mathbf{z}^{l}_{\text{spatial}}. \tag{2.19}$$

Temporal attention enables the model to track changes over time and detect irregular dynamics that cannot be captured from individual frames.

By alternating spatial and temporal attention across layers, TimeSformer avoids the computational burden of full space-time attention while enabling global reasoning along both dimensions.

Clip-Level Representation. As for ViT, TimeSformer represents videos as sequences of frame-wise patch tokens, with spatial and temporal positional encodings. A class token or global pooling is used to aggregate clip-level information after propagation through the network.

2.5 Area Under the Receiver Operating Characteristic Curve (AUC)

The Area Under the ROC Curve (AUC) is adopted as the primary evaluation metric in this thesis and provides a threshold-independent measure of the separability induced by a detector scoring function $s(x)$. Let x^p and x^n denote fake (positive) and real (negative) samples respectively, following the notation introduced later in chapter 6. The detector produces corresponding scores $s(x^p)$ and $s(x^n)$. AUC is formally defined as the probability that a randomly chosen fake sample receives a higher score than a randomly chosen real sample:

$$\mathrm{AUC}(s) = \mathbb{E}_{x^p,\, x^n}\left[\mathbb{I}\Big(s(x^p) > s(x^n)\Big)\right]. \tag{2.20}$$

This formulation emphasizes that AUC evaluates the *ranking ability* of the scoring function rather than its performance at any particular operating threshold. An equivalent geometric interpretation is obtained by integrating the ROC curve constructed over all thresholds τ:

$$\mathrm{AUC} = \int_0^1 \mathrm{TPR}(\mathrm{FPR})\, d(\mathrm{FPR}). \tag{2.21}$$

Because AUC depends solely on the ordering of scores, it is insensitive to class imbalance and threshold selection. These properties are particularly advantageous for deepfake detection, where real and fake samples may follow mismatched or highly skewed score distributions across datasets and manipulation types. For this reason, AUC serves as the principal metric for evaluating the methods developed in chapters 3, 4, 5.

2.6 Summary

This chapter has introduced the fundamental components needed for understanding the main contributions of this thesis, including common manipulation artifacts, multi-scale visual cues, and different variants of transformer-based architectures. It also reviews the AUC metric as it is the primary evaluation metric used in deepfake detection. With this background in place, the next chapters present the different contributions proposed in this work towards robust and generalizable deepfake detection.

Chapter 3

LAA-Net: Localized Artifact Attention Network for Quality-Agnostic and Generalizable Deepfake Detection

This chapter presents the first contribution of the present thesis, an image-based quality-agnostic and generalizable deepfake detection framework called LAA-Net. Building on the data synthesis strategies and multi-scale representations introduced in chapter 2, it focuses on the specific challenge of modeling subtle and localized artifacts that characterize high-quality (HQ) images, data, which are often undetected by conventional CNN-based detectors. This chapter first revisits the limitations of existing attention-based and multi-task methods, and then introduces LAA-Net, a fine-grained CNN architecture that combines an explicit multi-branch attention mechanism guided by vulnerable pixels with a novel Enhanced Feature Pyramid Network (E-FPN). Finally, it reports extensive experiments and analysis demonstrating that LAA-Net improves both generalization and robustness to unseen realistic manipulations. Moreover, it provides, to a certain extent, interpretability of detection decisions through its auxiliary branches.

3.1 Introduction

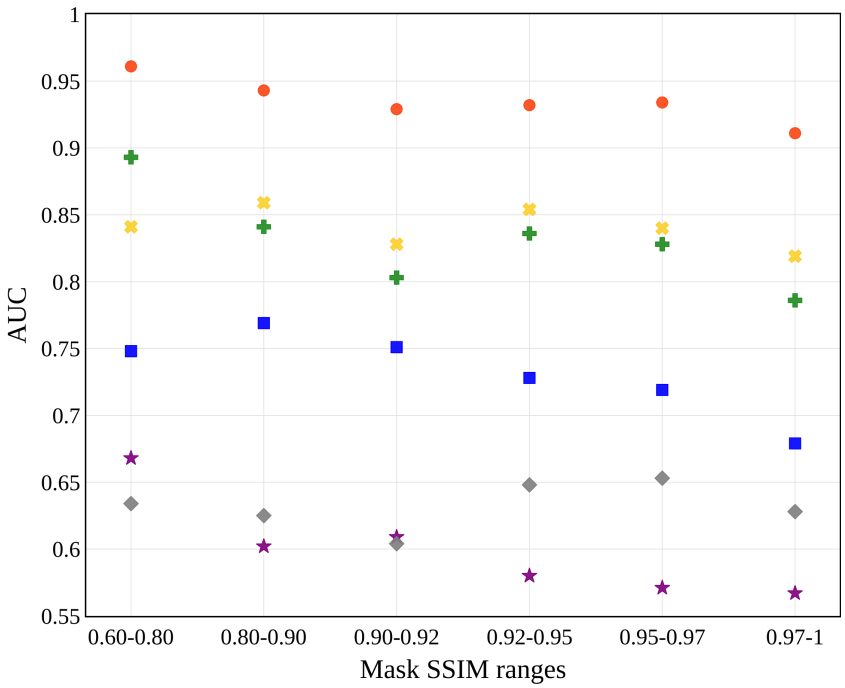
Thanks to the development of generative models, tremendous advances in deepfake creation have been witnessed. Unfortunately, these fake visual data can be employed for malicious purposes, as shown in [17, 18]. The fact that deepfake generation techniques are rapidly gaining in realism only exacerbates this issue. *It is, therefore, crucial to design methods capable of automatically detecting deepfakes, including the most realistic ones that are commonly referred to as high-quality deepfakes.* Nonetheless, detecting high-quality deepfakes remains extremely challenging as they usually enclose subtle and localized artifacts.

Recent works have mostly focused on improving the generalization capabilities of deepfake detection methods by adopting multi-task learning [25, 33, 31] and/or heuristic fake data generation [25, 26] strategies. However, most of these methods fail to model localized artifacts, which are critical for detecting high-quality deepfakes. This could be explained by the fact that Vanilla Deep Learning (DL) architectures are mainly used. These common architectures, such as XceptionNet [46] and EfficientNet [45], tend to learn global features, ignoring more localized cues [39, 48]. With the use of successive convolutions, localized features across layers gradually fade. Hence, proposing suitable mechanisms for capturing local and subtle artifacts turns out to be necessary.

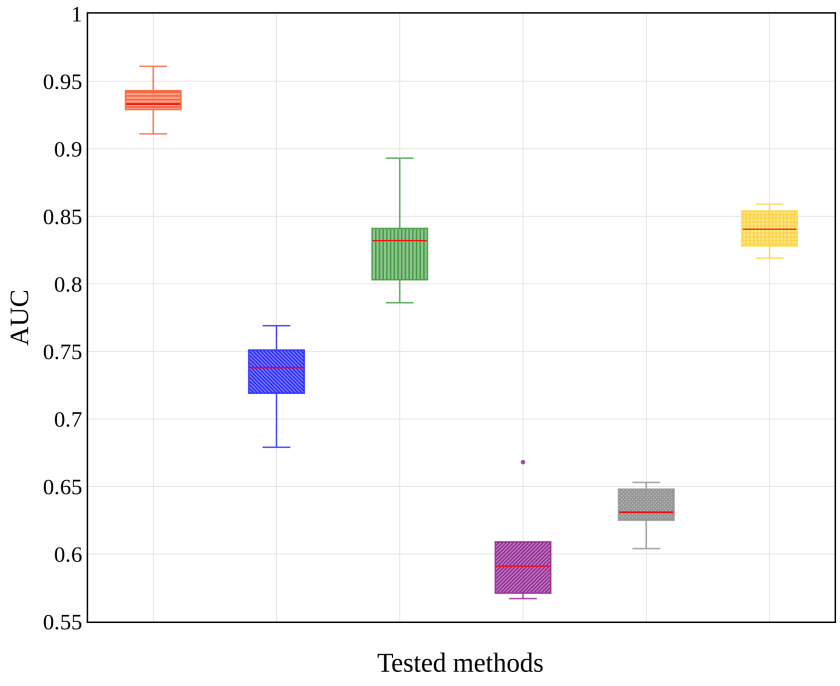
To the best of our knowledge, only a few research works have explored this research direction [39, 48]. They mainly introduce attention modules that implicitly model subtle inconsistencies through low-level representations [39, 48]. Nevertheless, they still rely on single binary classifiers trained with real/deepfake images without considering any additional strategy for avoiding generalization issues. This considerably restricts the practical usefulness of these methods.

Hence, our goal is to address the detection of high-quality deepfakes and, at the same time, improve the generalization performance. We argue that this can be achieved by designing an attention module compatible with generic deepfake detection strategies. In particular, the solution would be to introduce an explicit fine-grained mechanism within a multi-task learning framework supported by an appropriate pseudo-fake synthesis technique. Moreover, in addition to such a learning strategy, we posit that an adequate architecture preserving low-level features could implicitly contribute to better capturing localized artifacts.

More concretely, this chapter proposes a novel fine-grained approach called *Localized Artifact Attention Network (LAA-Net)* that relies on a multi-task learning framework. First, a new fine-grained mechanism that aims at focusing on small regions centered at the vulnerable pixels is introduced. By vulnerable pixels, we mean the pixels that are more likely to showcase



(a)



(b)

Figure 3.1 Comparison of LAA-Net (●) with respect to existing methods, namely, Multi-attentional (●) [39], SBI (●) [26], Xception (●) [23], RECCE (●) [32], CADDM (●) [27], using (a) the AUC performance with respect to different ranges of Mask SSIM, and (b) its associated boxplots. *The results were obtained using the official source codes pretrained on FF+ [23] and testing on Celeb-DFv2 [80]. Figure best viewed in colors.

a blending artifact[1]. This is achieved by considering two auxiliary branches, namely, *a heatmap branch* and a *self-consistency branch*. On the one hand, the heatmap branch allows localizing the set of vulnerable pixels while taking into account their neighborhood. On the other hand, the self-consistency branch estimates the similarity of pixels with respect to a randomly selected vulnerable point. To simulate fake data and generate ground-truth heatmaps and self-consistency matrices that are predicted by the additional branches, blending-based data synthesis such as [25, 26] are leveraged. Second, the proposed architecture incorporates a novel, simple, yet effective Feature Pyramid Network (FPN) [94] termed *Enhanced FPN* (E-FPN). It enables making use of multi-scale features while avoiding redundancy. In fact, it has been shown that reducing feature redundancy contributes to the regularization of Deep Neural Networks (DNNs) [103]. While the proposed attention mechanism guided by the vulnerable points helps the network to focus explicitly on artifact-prone regions, E-FPN forces the model to consider implicitly local cues. The association of these two complementary components makes LAA-Net a suitable candidate for fine-grained and generic deepfake detection. As reflected in Figure 3.1, our approach achieves better and more stable Area Under the Curve (AUC) performance as compared to existing methods [39, 26, 23, 32, 27] regardless of the quality of deepfakes, quantified using the Mask Structural SIMilarity (Mask-SSIM[2]). For a more comprehensive evaluation, in addition to the standard AUC, other metric is reported, namely, Average Precision (AP). We report experiments on several deepfake benchmarks and show that LAA-Net outperforms the state-of-the-art (SOTA).

Contributions. In summary, the chapter contributions are:

1. A novel multi-task learning method for fine-grained and generic deepfake detection called LAA-Net. It is trained using real data only.

2. An explicit attention mechanism for focusing on vulnerable points combining heatmap-based and self-consistency attention strategies.

3. A new FPN design, called E-FPN, ensures the efficient propagation of low-level features without incurring redundancy[3].

4. Extensive experiments and a comprehensive analysis reported on several benchmarks, namely, FF++ [23], CDF2 [80], DFD [83], DFDC [81], and DFW [82].

[1]A more formal definition is given in Section 3.3.1

[2]The Mask-SSIM [104, 80] has been proposed as a metric for quantifying the quality of deepfakes [80]. The Mask-SSIM is computed by computing the similarity in the head region between the fake image and its original version using the SSIM score introduced in [105]. Hence, a higher Mask-SSIM score corresponds to a deepfake of higher quality.

[3]E-FPN is generic and can be used in conjunction with any traditional encoder-decoder architecture.

Chapter Organization. The remainder of the chapter is organized as follows: Section 3.2 reviews related works. Section 3.3 introduces the proposed approach, and Section 3.4 reports the experiments and discusses the results. Finally, Section 3.5 concludes this work and suggests future investigations.

3.2 Related Works: Attention-based Deepfake Detection

Prior works are diverse in the way they approach the problem of deepfake detection [106, 107, 42, 40, 41, 108]. Earlier methods generally formulate it as a purely binary classification [23, 73], leading to poor generalization capabilities. As a solution, two main strategies have been investigated by the research community, namely, multi-task learning [25, 33, 31, 72, 109, 32] and/or pseudo-fake generation [25, 26, 65, 56].

Despite their great potential, the aforementioned models are less robust when considering high-quality deepfakes. Indeed, these SOTA methods mainly employ traditional DNN backbones such as XceptionNet [46] and EfficientNet [45]. Hence, through their successive convolution layers, they implicitly generate global features. As a result, low-level cues, that can be very informative, might be unintentionally ignored, leading to poor detection performance of high-quality deepfakes. It is, therefore, crucial to design adequate strategies for modeling more localized artifacts.

Alternatively, some attention-based methods such as [39, 48] have been proposed. Specifically, they have made attempts to integrate attention modules for implicitly focusing on low-level artifacts [39, 48]. Unfortunately, the two aforementioned methods make use of a unique binary classifier solely trained with real and deepfake images. This means that they do not consider any pseudo-fake generation technique or multi-task learning strategy. Consequently, as demonstrated experimentally, they do not generalize well to unseen datasets in comparison to other recent techniques [26, 65, 56].

3.3 Localized Artifact Attention Network (LAA-Net)

Our goal is to introduce a method that is robust to high-quality deepfakes yet capable of handling unseen manipulations. Accordingly, we introduce a fine-grained method called Localized Artifact Attention Network (LAA-Net) illustrated in Figure 3.2. LAA-Net incorporates: (1) an *explicit attention mechanism* and (2) a new architecture based on an *enhanced FPN*, called *E-FPN*.

First, the proposed attention mechanism aims at explicitly focusing on blending artifact-

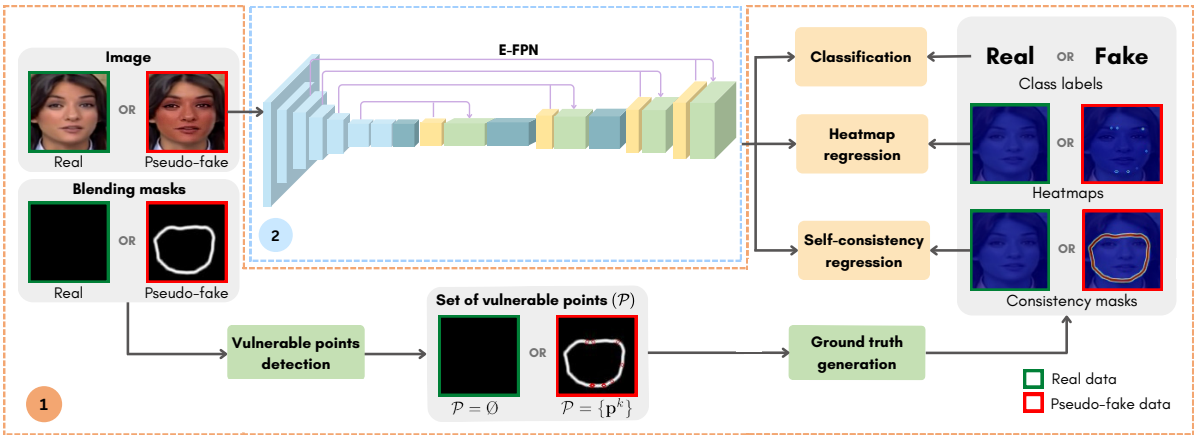


Figure 3.2 Overview of the proposed LAA-Net approach: it is formed by two components, namely, (1) an *explicit attention mechanism* based on a multi-task learning framework composed of three branches, i.e., the binary classification branch, the heatmap branch, and the self-consistency branch. The heatmap and self-consistency ground-truth data are generated based on the detected vulnerable points, and (2) an *Enhanced Feature Pyramid Network (E-FPN)* that aggregates multi-scale features.

prone pixels referred to as vulnerable points (a formal definition is given in Section 3.3.1). For that purpose, a hand-free annotation of vulnerable points is proposed by leveraging a blending-based data synthesis. Specifically, a multi-task learning framework composed of three simultaneously optimized branches, namely (a) classification, (b) heatmap regression, and (c) self-consistency regression, is introduced, as depicted in Figure 3.2. The classification branch predicts whether the input image is fake or real, while the two other branches aim at giving attention to vulnerable pixels. Second, E-FPN allows extracting multi-scale features without injecting redundancy. This enables modeling low-level features, which can better discriminate subtle inconsistencies.

3.3.1 Explicit Attention to Vulnerable Points

Proposed Multi-task Learning Framework

In addition to the deepfake classification branch, the network learns to focus on specific regions by taking advantage of the parallel *Heatmap* and *Self-consistency* branches. Our hypothesis is that deepfake detection can be formulated as a fine-grained classification. Therefore, giving more attention to the vulnerable points should be an effective solution for detecting high-quality deepfakes. For the sake of clarity, we start by formally defining the notion of *"vulnerable points"*.

Definition 1. - *Vulnerable points in a deepfake image are the pixels that are more likely to carry blending artifacts.*

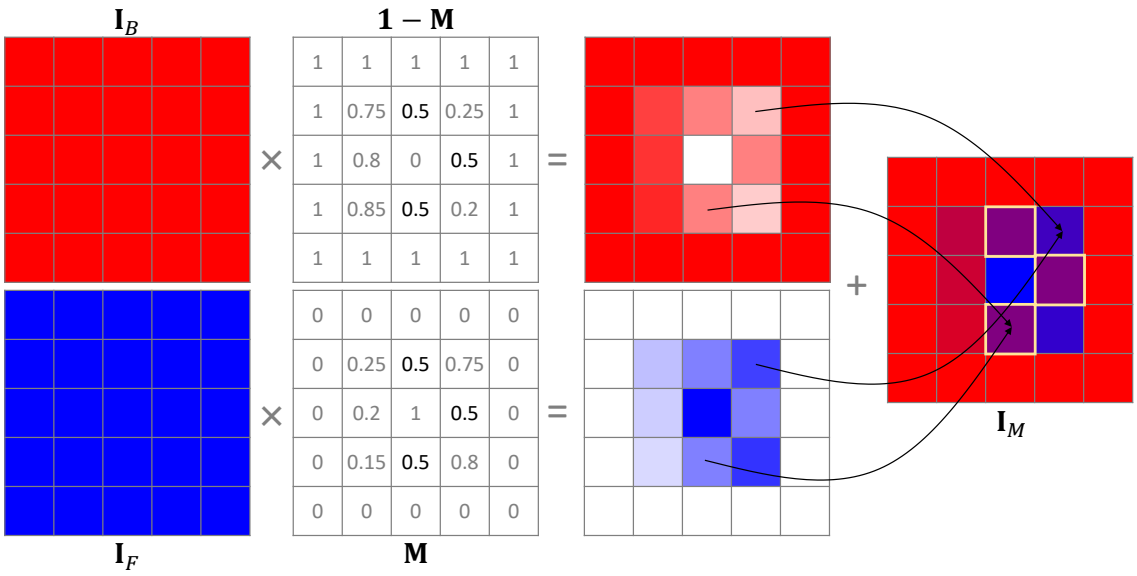


Figure 3.3 Extraction of the vulnerable points.

As discussed in Section 2.2, any deepfake generation approach involves a blending operation for mixing the background and the foreground of two different images $\mathbf{I}_B$ and $\mathbf{I}_F$, respectively. This implies the presence of blending artifacts regardless of the used generation approach. Thus, we posit that the vulnerable points can be seen as the pixels belonging to the blending regions with the most equivalent contributions from both $\mathbf{I}_B$ and $\mathbf{I}_F$.

In this chapter, we assume that we work under a realistic setting where we only have access to real data during training. A blending-based augmentation is, therefore, considered and leveraged for defining vulnerable pixels. Specifically, inspired from [25], a blending boundary mask $\mathbf{B} = (b_{ij})_{i,j\in[\![1,D]\!]}$ is firstly computed as described in Eq. (2.2).

The variable D is the height and width of $\mathbf{B}$, and b_{ij} its value at the position (i, j). A higher value of b_{ij} indicates that the position (i, j) is more impacted by the blending. Hence, if an input image is real, $\mathbf{B}$ should be set to $\mathbf{0}$. Then, the set of vulnerable pixels denoted by $\mathcal{P}$ is defined as follows,

$$\mathcal{P} = \underset{(i,j)\in[\![1,D]\!]^2}{\arg\max}\,(\mathbf{B}), \tag{3.1}$$

where $[\![\ ]\!]$ defines an integer interval. Figure 3.3 illustrates the extraction of vulnerable points. In the following, we describe how the notion of vulnerable points is used within the heatmap and self-consistency branches.

Heatmap Branch. In general, forgery artifacts not only appear in a single pixel but also affect its surroundings. Hence, considering vulnerable points as well as their neighborhood is more appropriate for effectively discriminating deepfakes, especially in the presence of images with local irregularities caused by noise or illumination changes. To model that, we

propose to use a heatmap representation that encodes at the same time the information of both vulnerable points as well as their neighbor points.

More specifically, ground-truth heatmaps are generated by fitting an *Unnormalized Gaussian Distribution* for each pixel $\mathbf{p}^k \in \mathcal{P}$. The pixel $\mathbf{p}^k$ is considered as the center of the Gaussian Mask $\mathbf{G}^k$. To take into account the neighborhood information of $\mathbf{p}^k$, the standard deviation of $\mathbf{G}^k$ is adaptively computed. In particular, inspired from the work of [110], the standard deviation σ_k of $\mathbf{p}^k$ is computed based on the width and the height of the blending boundary mask $\mathbf{B}$ with respect to the point $\mathbf{p}^k$. Similar to [110], a radius r_k is computed based on the size of the set of virtual objects that overlap the mask centered at $\mathbf{p}^k$ with an Intersection over Union (IoU) greater than a threshold t. In all our experiments, we set t to 0.7 and we assume that $\sigma_k = \frac{1}{3} r_k$. Hence, $\mathbf{G}^k = (g_{ij}^k)_{i,j \in [\![1,D]\!]}$ is computed as follows,

$$g_{ij}^k = e^{-\frac{i^2+j^2}{2\sigma_k^2}} , \tag{3.2}$$

where i and j refer to the pixel position. The ground-truth heatmap $\mathbf{H}$ is finally constructed by superimposing the set $\mathcal{G} = \{\mathbf{G}^k\}_{k \in [\![1,\text{card}(\mathcal{P})]\!]}$. A figure depicting the heatmap generation process is provided in supplementary materials.

For optimizing the heatmap branch, the following focal loss [95] is used,

$$L_{\text{H}} = \sum_{i,j}^{D} -(1 - \tilde{h}_{ij})^{\gamma} \log \tilde{h}_{ij} , \tag{3.3}$$

such that,

$$\tilde{h}_{ij} = \begin{cases} \hat{h}_{ij} & \text{if } h_{ij} = 1 , \\ 1 - \hat{h}_{ij} & \text{otherwise} , \end{cases} \tag{3.4}$$

with $\hat{h}_{ij}$ and h_{ij} being the value of the predicted heatmap $\hat{\mathbf{H}}$ and the ground-truth $\mathbf{H}$ at the pixel location (i, j), respectively. The hyperparameter γ is used to stabilize the adaptive loss weights.

Self-consistency Branch. To enhance the proposed attention mechanism, the idea of learning self-consistency proposed in [33] is revisited to fit our context. Instead of computing the consistency values for each pixel of the mask, we consider only the vulnerable location. Since the set $\mathcal{P}$ might include more than one pixel (the blending mask can include several pixels with equal values), we randomly choose one of them that we denote by $\mathbf{p}^s$ for generating the self-consistency ground-truth matrix. Hence, the generated matrices denoted by $\mathbf{C}$ are 2-dimensional and not 4-dimensional as in the original method. Given the randomly selected

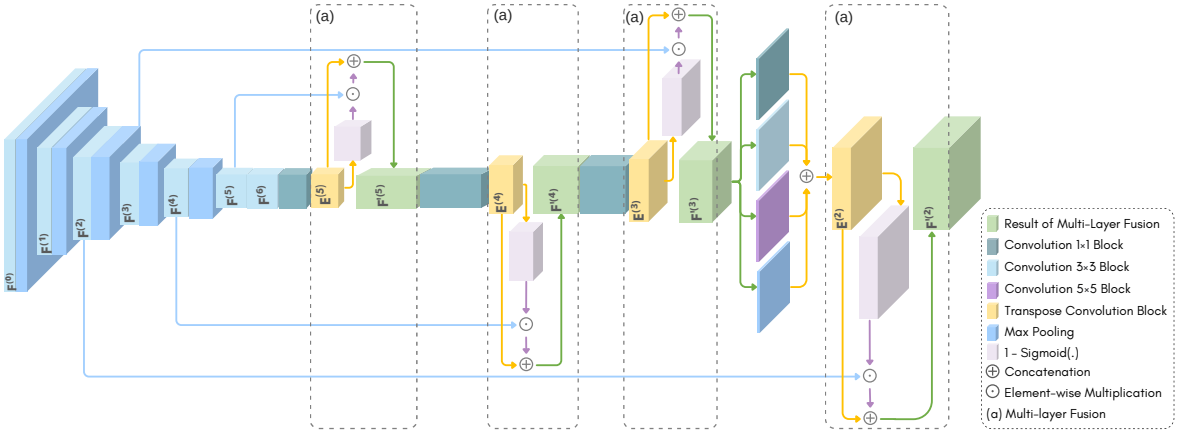


Figure 3.4 Architecture of the proposed Enhanced Feature Pyramid Network (E-FPN).

vulnerable point $\mathbf{p}^s = (u, v)$, the self-consistency $\mathbf{C}$ matrix is computed as,

$$\mathbf{C} = \mathbf{1} - |b_{uv}.\mathbf{1} - \mathbf{B}| \ , \tag{3.5}$$

where $|.|$ refers to the element wise modulus and $\mathbf{1}$ is an all-one matrix.

This refinement allows for reducing the model size and, consequently, the computational cost. It can also be noted that even though our method is inspired by [33], our self-consistency branch is inherently different. In [33], the consistency is calculated between the foreground and background, whereas we measure the consistency between the vulnerable point and the other pixels of the blended mask. The self-consistency loss L_{C} is then computed as a binary cross entropy loss between $\mathbf{C}$ and the predicted self-consistency $\hat{\mathbf{C}}$.

Training Strategy. The network is optimized using the following loss,

$$L = L_{\mathrm{BCE}} + \lambda_1 L_{\mathrm{H}} + \lambda_2 L_{\mathrm{C}} \ , \tag{3.6}$$

where L_{BCE} denotes the binary cross-entropy classification loss. L_{H} and L_{C} are weighted by the hyperparameters λ_1 and λ_2, respectively. Note that only real and pseudo-fakes are used during training.

3.3.2 Enhanced Feature Pyramid Network (E-FPN)

Feature Pyramid Networks (FPN) are widely adopted feature extractors capable of complementing global representations with multi-scale low-level features captured at different resolutions [94]. This makes them ideal candidates for implicitly supporting the heatmap and self-consistency branches towards fine-grained deepfake detection. Although some attempts

have been made to exploit multi-scale features [27], no previous works have considered FPN in the context of deepfake detection.

Over the last years, several FPN variants have been proposed for numerous computer vision tasks [95, 111, 96, 94]. Nevertheless, these FPN-based methods usually lead to the generation of redundant features, which might, in turn, lead to the overfitting of the model [103]. Moreover, as described in Section 3.1, small discrepancies are gradually eliminated through the successive convolution blocks [39], going from high-resolution low-level to low-resolution high-level features. Consequently, the last block outputs usually contain global features where local artifact-sensitive features might be discarded. To overcome this issue, we introduce a new alternative referred to as Enhanced Feature Pyramid Network (E-FPN) that is integrated in the proposed LAA-Net architecture. The E-FPN goal is to propagate relevant information from high to low-resolution feature representations.

As shown in Figure 3.4, we denote the output shape of the $N-1$ latest layers by $(n^{(l)}, D^{(l)}, D^{(l)})$ with $l \in [\![2, N]\!]$. For the sake of simplicity, we assume that the shape of the feature maps is square. For a given layer l, $n^{(l)}$, $D^{(l)}$ and $\mathbf{F}^{(l)}$ correspond, respectively, to its feature dimension, its height and width, and its output features. For strengthening the textural information in the ultimate layer $\mathbf{F}^{(N)}$, we propose to take advantage of the features generated by previous layers $\mathbf{F}^{(l)}$ with $l \in [\![2, N-1]\!]$. Concretely, for each layer l, a convolution followed by a transpose convolution is applied to $\mathbf{F}^{(l+1)}$. The obtained features are denoted by $\mathbf{E}^{(l)}$ and have the same shape as $\mathbf{F}^{(l)}$. Then, a sigmoid function is applied to $\mathbf{E}^{(l)}$ returning probabilities. The latter indicates the pixels that contributed to the final decision. For enriching $\mathbf{F}^{(l+1)}$ while avoiding redundancy related to the most contributing pixels, the features $\mathbf{F}^{(l)}$ are filtered by computing $(1 - \text{sigmoid}(\mathbf{E}^{(l)}))^{\gamma_w}$ resulting in a weighted mask. The latter is concatenated along the same axis with $\mathbf{E}^{(l)}$ for obtaining the final features. This operation is iterated for all the layers $l \in [\![2, N-1]\!]$. In summary, the final representation $\mathbf{F}'^{(l)}$ is obtained as follows,

$$\mathbf{F}'^{(l)} = (\mathbf{F}^{(l)} \odot (1 - \text{sigmoid}(\mathbf{E}^{(l)}))^{\gamma_w} \oplus \mathbf{E}^{(l)}) \,, \tag{3.7}$$

where $\mathbf{E}^{(l)} = \mathfrak{T}(f(\mathbf{F}'^{(l+1)})$ with $\mathbf{F}'^{(l+1)} = \mathbf{F}^{(l+1)}$ if $l = N-1$, such that f and $\mathfrak{T}$, are respectively the convolution and transpose convolution operators, and $\oplus$ refer to the concatenation operator. The hyper-parameter γ_w is set to 1 in all our experiments. The relevance of E-FPN in the context of deepfake detection is experimentally demonstrated in Section 3.4, as compared to the traditional FPN.

Method	Training set		Test set								
			In-dataset	Cross-dataset							
	Real	Fake	FF++	CDF2		DFW		DFD		DFDC	
			AUC (%)	AUC (%)	AP (%)	AUC (%)	AP (%)	AUC (%)	AP (%)	AUC (%)	AP (%)
Xception [23]	✓	✓	99.09	61.18	66.93	65.29	55.37	89.75	85.48	69.90	91.98
FaceXRay+BI [25]	✓	✓	99.20	79.5	-	-	-	95.40	93.34	65.5	-
LRNet [112]	✓	✓	-	53.20	-	-	-	52.29	-	-	-
LocalRL [63]	✓	✓	99.92	78.26	-	-	-	89.24	-	76.53	-
TI^{2}Net [113]	✓	✓	-	68.22	-	-	-	72.03	-	-	-
Multi-attentional [39]	✓	✓	-	68.26	75.25	73.56	73.79	92.95	96.51	63.02	-
RECCE [32]	✓	✓	-	70.93	70.35	68.16	54.41	98.26	79.42	-	-
SFDG [48]	✓	✓	99.53	75.83	-	69.27	-	88.00	-	73.63	-
EIC+IIE [114]	✓	✓	99.32	83.80	-	-	-	93.92	-	81.23	-
AltFreezing [56]	✓	✓	98.6	89.50	-	-	-	98.50	-	-	-
CADDM [27]	✓	✓	99.79	<u>93.88</u>	91.12	<u>74.48</u>	<u>75.23</u>	99.03	<u>99.59</u>	-	-
UCF [44]	✓	✓	-	82.4	-	-	-	94.5	-	80.5	-
Controllable GS [115]	✓	✓	-	84.97	-	-	-	-	-	81.65	-
PCL+I2G [33]	✓		99.11	90.03	-	-	-	99.07	-	74.27	-
SBI [26]	✓		99.64	93.18	85.16	67.47	55.87	97.56	92.79	86.15	93.24
AUNet [65]	✓		99.46	92.77	-	-	-	<u>99.22</u>	-	<u>86.16</u>	-
Ours (w/ BI)	✓		<u>99.95</u>	86.28	<u>91.93</u>	57.13	56.89	**99.51**	**99.80**	69.69	<u>93.67</u>
Ours (w/ SBI)	✓		**99.96**	**95.40**	**97.64**	**80.03**	**81.08**	98.43	99.40	**86.94**	**97.70**

Table 3.1 In-dataset and Cross-dataset evaluation in terms of AUC and AP on multiple deepfake datasets. **Bold** and <u>Underlined</u> highlight the best and the second-best performance, respectively.

3.4 Experiments

In this section, we start by presenting the experimental settings. Then, we compare the performance of LAA-Net to SOTA methods, both qualitatively and quantitatively. Finally, we conduct an ablation study to validate the different components of LAA-Net.

3.4.1 Experimental Settings

Datasets. The FF++ [23] dataset is used for training and validation. In our experiments, we follow the standard splitting protocol of [23]. This dataset contains 1000 original videos and 4000 fake videos generated by four different manipulation methods, namely, Deepfakes (DF) [116], Face2Face (F2F) [117], FaceSwap (FS) [118], and NeuralTextures (NT) [119]. In the training process, we utilize real images only to dynamically generate pseudo-fakes, as discussed in Section 3.3. To evaluate the generalization capability of the proposed approach as well as its robustness to high-quality deepfakes, we test the trained model on four datasets incorporating different quality of deepfakes, namely, Celeb-DFv2 [80] (CDF2), DeepFake Detection [83] (DFD), DeepFake Detection Challenge [81] (DFDC) and Wild Deepfake [82] (DFW). To assess the quality of the considered datasets, we compute the Mask-SSIM[2] for each benchmark. In particular, CDF2 [80] is formed by the most realistic deepfakes with an average Mask-SSIM [104] value of 0.92, followed by DFD and DFDC with an average Mask-SSIM of 0.88 and 0.84, respectively. We note that computing the Mask-SSIM [80] for DFW was not possible since real and fake images are not paired.

Method	Fake	Saturation	Contrast	Block	Noise	Blur	Pixel
Xception [46]	✓	99.3	98.6	99.7	53.8	60.2	74.2
FaceXray [25]	✓	97.6	88.5	99.1	49.8	63.8	88.6
LipForensics [54]	✓	<u>99.9</u>	99.6	87.4	<u>73.8</u>	96.1	95.6
CADDM [27]	✓	99.6	<u>99.8</u>	<u>99.8</u>	**87.4**	**99.0**	<u>98.8</u>
Ours		**99.96**	**99.96**	**99.96**	53.9	<u>98.22</u>	**99.80**

Table 3.2 Robustness inspection on the FF++ with different types of perturbation. **Bold** and <u>Underline</u> highlight the best and the second-best performance, respectively.

Evaluation Metrics. To compare the performance of LAA-Net with the state-of-the-art, we report the common Area Under the Curve (AUC) metric at the video-level and the Average Precision (AP) as in [25, 33, 26, 27]. More metrics, namely, Average Recall (AR) and mean F1-score (mF1) are provided in supplementary materials.

Implementation Details. To train our model, 128 training and 32 validation frames are used. RetinaNet [95] is used to crop faces with a conservative enlargement (by a factor of 1.25) around the face center. Note that all the cropped images are then resized to 384×384. In addition, 68 facial landmarks are extracted per frame using Dlib [120]. We adopt the EFNB4 variant of the EfficientNet [45] pretrained on ImageNet [121]. For each training epoch, 8 frames are dynamically selected and used for online pseudo-fake generation. The model is trained for 100 epochs with the SAM optimizer [122], a weight decay of 10^{-4}, and a batch size of 16. We apply a learning rate scheduler that increases from 5.10^{-5} to 2.10^{-4} in the first quarter of the training and then decays to zero in the remaining quarters. We freeze the backbone at the first 6 epochs and only train the remaining layers. For data augmentation, we apply horizontal flipping, random cropping, random scaling, random erasing [123], color jittering, Gaussian noise, blurring, and JPEG compression. The parameters λ_1 and λ_2, defined in Eq. (3.6), are set to 10 and 100. Furthermore, label smoothing [124] is utilized as a regularizer. To generate pseudo-fakes, two blending synthesis techniques are considered, namely, Blended Images (BI) [25] and Self-Blended Images (SBI) [26]. All experiments are carried out using a GPU Tesla V-100.

3.4.2 Comparison with State-of-the-art

In-dataset Evaluation. We compare the performance of LAA-Net to existing methods under the in-dataset protocol of [33, 27, 26, 65, 48, 56]. The first column in Table 3.1 reports the obtained results on the testing set of FF++. It can be seen that all methods achieve competitive performance on the forgeries of the FF++ dataset. Our method combined with SBI outperforms all methods with an AUC of 99.96%, while using only real data for training.

C	H	E-FPN	Test set AUC (%)				
			CDF2	DFD	DFDC	DFW	Avg.
×	×	×	74.54	92.24	70.81	59.81	74.35
×	✓	×	80.89	94.53	77.93	67.12	80.12(↑5.77)
×	×	✓	84.21	95.03	80.68	65.47	81.35(↑7.00)
×	✓	✓	**95.56**	**98.54**	82.21	74.98	87.82(↑13.47)
✓	×	✓	79.87	94.60	71.70	72.47	79.66(↑5.31)
✓	✓	×	91.56	98.27	78.35	73.02	85.30(↑10.95)
✓	✓	✓	95.40	98.43	**86.94**	**80.03**	**90.20(↑15.85)**

Table 3.3 Ablation study under the cross-dataset setup of the Consistency branch (C), Heatmap branch (H), and E-FPN.

	EFNB4					Test Set AUC (%)							
		E-FPN Integration				CDF2		DFD		DFW		DFDC	
	$\mathbf{F}^{(6)}$	$\mathbf{F}^{(5)}$	$\mathbf{F}^{(4)}$	$\mathbf{F}^{(3)}$	$\mathbf{F}^{(2)}$	FPN	E-FPN	FPN	E-FPN	FPN	E-FPN	FPN	E-FPN
(a)	√					91.56		98.27		73.02		78.35	
(b)	√	√				93.42	91.79	98.59	97.12	73.78	71.39	78.40	75.80
(c)	√	√	√			88.72	92.86	97.96	**98.95**	69.40	74.93	71.91	83.97
(d)	√	√	√	√		88.35	**95.40**	98.89	98.43	70.94	**80.03**	79.02	**86.94**
(e)	√	√	√	√	√	92.16	94.22	96.58	97.31	65.17	72.54	74.31	82.90
Avg						90.84	*93.16*	*98.06*	98.02	70.46	*74.38*	76.40	*81.59*

Table 3.4 Traditional FPN versus E-FPN, using the SBI-based data synthesis under the cross-dataset protocol. **Bold** and Underline indicate the best and the second-best performance, respectively. We report the results when integrating features $\mathbf{F}^{(i)}$ from different layers.

Cross-dataset Evaluation. We evaluate LAA-Net under the challenging cross-dataset setup [32, 48]. Table 3.1 reports the obtained results on CDF2, DFW, DFD, and DFDC, respectively. It can be noted that LAA-Net achieves state-of-the-art results on the four considered benchmarks, thereby demonstrating its robustness to different quality of deepfakes. The best performance is reached using SBI as a data synthesis, confirming the importance of modeling subtle artifacts. The performance of LAA-Net (w/BI) is slightly superior to LAA-Net (w/SBI) only on DFD, with an improvement of 1.08% and 0.4% of AUC and AP, respectively. A plausible explanation could be the fact that deepfake detection in DFD is less challenging. In fact, numerous methods report AUC and AP scores exceeding 98%.

Furthermore, LAA-Net clearly outperforms attention-based approaches such as Multi-attentional [39] and SFDG [48] by a margin of 27.14% and 19.57% in terms of AUC and AP on CDF2, respectively. This confirms the superior generalization capabilities of LAA-Net as compared to [39, 48]. These results are further supported by high AR and mF1, which are provided in the supplementary materials.

Robustness to Perturbations. Since deepfake videos are easily altered on various social platforms, the robustness of LAA-Net against some common perturbations is investigated. Following the same settings of [54, 27], we evaluate the performance of LAA-Net on FF++ [23]

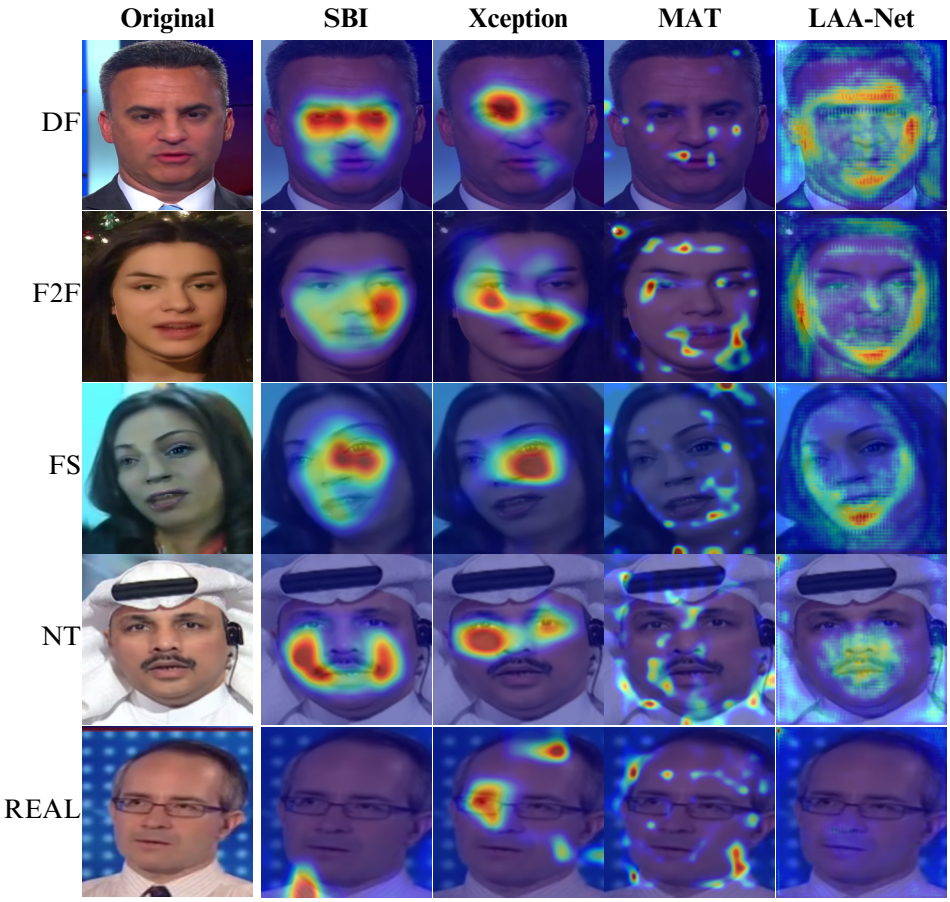


Figure 3.5 Grad-CAM [127] visualization on different types of manipulation from FF++ [23]. LAA-Net is compared to SBI [26], Xception [23], and MAT [39].

by applying different corruptions. The results are reported in Table 3.2. As our method focuses on vulnerable points, it can be seen that color-related changes such as saturation and contrast do not impact the performance. However, the proposed method is extremely sensitive to structural perturbations such as Gaussian Noise. In future work, strategies for ensuring more robustness to structural perturbations will be investigated. For instance, denoising methods [125, 126] will be considered for solving this issue.

Qualitative Results. We provide Grad-CAMs [127] in Figure 3.5, to visualize the image regions in deepfakes that are activated by LAA-Net, SBI [26], Xception [23], and Multi-attentional (MAT) [39] on FF++ [23]. Generally, attention-based methods such as MAT [39] and LAA-Net focus more on localized regions. However, in some cases, MAT [39] concentrates on irrelevant regions such as the background or the inner face areas. Conversely, LAA-Net consistently identifies blending artifacts and shows interesting capabilities on mouth-rendered Neural Textures (NT).

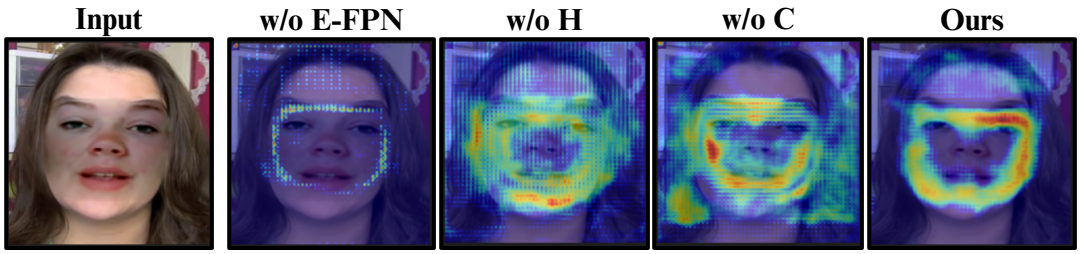


Figure 3.6 GradCAM [127] visualization of different components in LAA-Net. w/o E-FPN, w/o H, and w/o C refer to ablating E-FPN, heatmap branch, and self-consistency branch, respectively.

3.4.3 Ablation Study

Table 3.3 reports the cross-dataset performance of LAA-Net when discarding the following components: E-FPN, the consistency branch denoted by C and the heatmap branch denoted by H. The best performance is reached when all the components are integrated. It can be seen that the proposed explicit attention mechanism through the heatmap branch contributes more to improving the result. A qualitative example visualizing Grad-CAMs [127] with different components of LAA-Net is also given in Figure 3.6. The illustration clearly shows that by combining the three components, the network activates more precisely the blending region.

3.4.4 E-FPN versus Traditional FPN

To assess the effectiveness of the low-level features injected by E-FPN into the final feature representation, we combine different feature levels and compare the results of E-FPN and traditional FPN [94, 95] in Table 3.4. It can be seen that in general E-FPN outperforms FPN except for $\mathbf{F}^{(5)}$. This confirms the relevance of employing multi-scale features and the need for reducing their redundancy in the context of deepfake detection.

3.4.5 Sensitivity Analysis

In this subsection, we analyze the impact of the two hyperparameters, λ_1 and λ_2 given in Eq. (3.6). Table 3.5 shows the experimental results for different values of λ_1 and λ_2. It can be noted that our model is robust to different hyperparameter values, with the best average performance obtained with $\lambda_1 = 10$ and $\lambda_2 = 100$.

3.5 Conclusion

In this chapter, a fine-grained deepfake detection method called LAA-Net is introduced with the aim of detecting high-quality deepfakes while remaining generic to unseen manip-

λ_1	λ_2	Test Set AUC (%)			
		CDF2	DFDC	DFW	Avg
1	1	90.69	78.12	70.98	79.93
10	10	**95.73**	<u>85.87</u>	73.56	<u>85.05</u>
100	100	93.72	78.60	75.25	82.52
100	10	93.05	83.86	<u>76.72</u>	84.54
10	100	<u>95.40</u>	**86.94**	**80.03**	**87.46**

Table 3.5 Sensitivity analysis: The impact of the hyper-parameters λ_1 and λ_2 using the cross-dataset protocol on three datasets in terms of AUC.

ulations. For that purpose, two different components are proposed. On the one hand, we argue that by making the network focus on the most vulnerable points, we can detect both global and subtle artifacts. To this end, an explicit attention mechanism within a multi-task learning framework is used. In addition to the binary classification branch, heatmap and self-consistency branches are defined with respect to the vulnerable points. On the other hand, a novel E-FPN module for aggregating multi-scale features is proposed; hence enabling the integration of more localized features. The results reported on several benchmarks show the superiority of LAA-Net as compared to the state-of-the-art, including attention-based methods. In future works, strategies for improving the robustness to noise will be investigated using recent transformer architectures. In addition, an attempt to extend this idea by taking into account the temporal dimension will be explored.

Chapter 4

LAA-Former: Efficient Vulnerability-Driven Transformers for Quality-Agnostic and Generalizable Deepfake Detection

This chapter presents the second contribution of the thesis, LAA-Former, which extends vulnerability-aware deepfake detection to transformer-based architectures. While chapter 3 has demonstrated the benefits of explicit localized artifact attention and multi-scale features for image-level CNN-based detection, transformers which have recently emerged as a powerful alternative for visual recognition remain underexplored in deepfake detection. This might be explained by the fact that they often struggle with fine-grained forgery cues. Building on the vulnerability idea introduced in LAA-Net, this chapter first investigates the gap between plain Vision Transformers and CNN-based detectors in modeling localized artifacts. It then introduces LAA-Former and its Swin-based counterpart, LAA-Swin, which integrate a lightweight Learning-based Local Attention (L2-Att) module to steer attention toward vulnerable patches while preserving global context modeling. Finally, this chapter presents extensive experiments showing that these architectures improve generalization while being

relatively efficient.

4.1 Introduction

In the era of deep learning, it has never been easier to generate ultra-realistic facial images, undetectable to the naked eye. Such forged visual data, commonly called deepfakes [128, 129], are produced by manipulating the identity or the expression of a subject using generative deep neural networks such as Generative Adversarial Networks (GAN) [1]. Although beneficial in several applications, deepfakes can also be harmful to society if employed for malicious purposes such as spreading misinformation and fraud.

To face this growing threat, the research community is making ceaseless efforts to introduce approaches for automatically detecting deepfakes. Currently, the most successful deepfake detection methods are mostly relying on Convolution Neural Networks (CNNs) [27, 54, 61, 56, 26, 33, 31, 62, 63, 64, 44, 65, 66, 67]. Paradoxically, thanks to their simplicity and scalability, Vision Transformers [68, 69] (ViTs) are becoming increasingly popular in other visual classification tasks, often outperforming CNN architectures. However, ViT architectures remain relatively underexplored in the field of deepfake detection, given their lower performance as compared to CNNs in that context [37, 34].

In this chapter, we aim to: *(1) investigate the reasons behind this drop in performance with respect to CNNs* and *(2) suggest accordingly adequate yet simple strategies for enhancing the performance of ViT architectures in the context of deepfake detection, while benefiting from their efficiency*. Specifically, we first experimentally analyze the capability of ViTs to capture localized artifacts typically characterizing deepfakes, as compared to CNNs. Our analysis suggests that ViTs struggle more in this regard because they tend to focus on global context [68, 84, 87, 85, 86].

Therefore, to address this issue, we propose to inject a local attention mechanism in ViTs by explicitly detecting vulnerable patches using blending-based data synthesis techniques such as [26, 25]. For that purpose, we extend the notion of vulnerable points [62], which represent the pixels that are the most impacted by the blending operation, to vulnerable patches. These patches are the group of pixels that are the most likely to incorporate blending artifacts and can be used in conjunction with any transformer-like architecture. This attention module is referred to as Learning-based Local Attention mechanism (L2-Att). Even though the idea of attention to vulnerabilities has been explored in [62], our experimental study shows that generalizing this approach to transformers contributes to:

1. **Enhanced generalization performance:** As reflected in Figure 4.1, higher general-

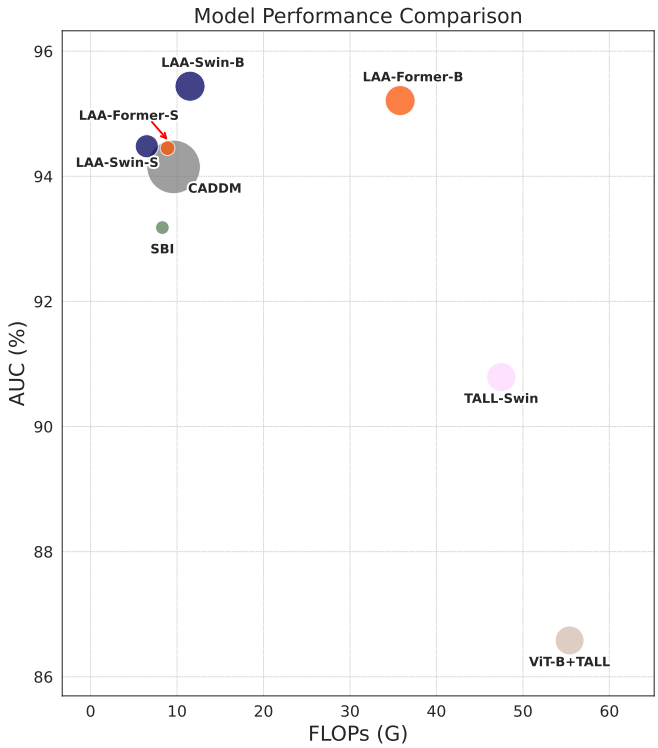


Figure 4.1 Comparison of LAA-Former (and LAA-Swin) to existing methods including SBI [26], CADDM [27], and Transformer-based approaches [36], namely TALL-Swin and ViT-B+TALL, in terms of model size, FLOPs, and AUC. The size of each bubble represents the number of model parameters. All methods are trained on FF++ [23] and tested on CDF2 [80].

ization capabilities are achieved by different transformer-based architectures enclosing the proposed L2-Att strategy, namely LAA-Former and LAA-Swin. Specifically, the LAA-Swin is a variant of LAA-Former where Swin [85] is used as a backbone. The notations -S and -B refer to the backbone size (small and base, respectively)[1].

2. **Friendly computational cost and reduced model size**: As illustrated in Figure 4.1, although achieving impressive performance, the proposed transformer models cost a relatively low number of FLOPs, and necessitate fewer parameters as compared to other methods.

3. **Mitigating the need for large-scale datasets**: ViTs are known to be data-hungry [68, 71, 100, 69]. Therefore, by employing generic blending-based synthesis methods as [25, 26] to define pseudo-vulnerable patches, we mitigate this problem.

This is confirmed by extensive experiments conducted on seven challenging datasets including FF++ [23], two versions of CDF [80], DFD [83], DFW [82], DFDCP [81], and DFDC [88]. Our method outperforms state-of-the-art approaches, including both CNN-based and ViT-based

[1]More details are reported in Section 4.5.

methods [37, 35, 34], although the latter [37, 35, 34] usually rely on a significantly larger training set.

Chapter Organization. Section 4.2 discusses related work. Section 4.3 analyzes the performance gap between ViTs and CNNs in deepfake detection. The proposed approach called LAA-Former is presented in Section 4.4. Section 4.5 discusses the experiments. Section 4.6 concludes this work and suggests future directions.

4.2 Related Work

Deepfake Detection with CNNs. In the field of deepfake detection, Convolution Neural Networks (CNNs) are often favored over other types of architectures. Pioneer works in deepfake detection have predominantly employed established CNN architectures such as XceptionNet [46] as binary classifiers [23, 40, 130]. However, these methods struggle to generalize to unseen manipulation methods. To address this challenge, a wide range of strategies have been investigated [27, 61, 32, 44, 26, 25] such as disentanglement learning [32, 44], multi-task learning [31, 62, 33], and pseudo-fake synthesis [26, 25, 33, 64]. With the advances in generative modeling, deepfakes are constantly gaining in realism, resulting in invisible localized artifacts. The aforementioned methods are exposed to the risk of becoming obsolete. Indeed, they still rely on standard CNNs, consequently facing a loss of local information through successive convolution layers [39, 48]. For capturing more effectively low-level features, methods with implicit attention strategies have been proposed [39, 48, 93], but have shown poor generalization [62]. Recently, Nguyen *et al.* [62] argued that to ensure robustness to high-quality deepfakes while maintaining good generalization capabilities, an explicit attention mechanism within a multi-task learning framework is needed. In particular, they introduce LAA-Net, a network guided to focus on vulnerable points (defined as the pixels that are the most likely to incorporate blending artifacts). To simulate vulnerable pixels, blending-based data synthesis techniques are leveraged. However, although LAA-Net achieves state-of-the-art performances, it is sensitive to noise and is based on a cumbersome CNN that is computationally costly, thereby limiting its practical deployment. In this chapter, we investigate how to adapt such an explicit attention mechanism to ViT-based architectures to improve the effectiveness of transformer-based deepfake detection methods and benefit from more efficient models.

Deepfake Detection with ViTs. Vision Transformers (ViTs) have attracted the great interest of the research community, as they are claimed to outperform CNNs in terms of both effectiveness and efficiency. While such a statement holds for several applications [97, 68,

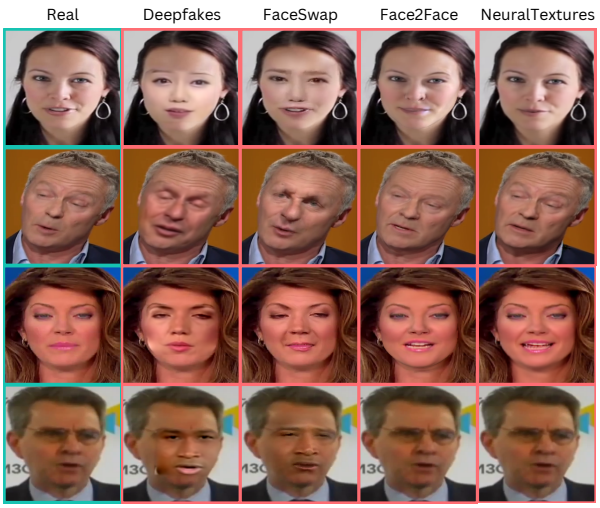


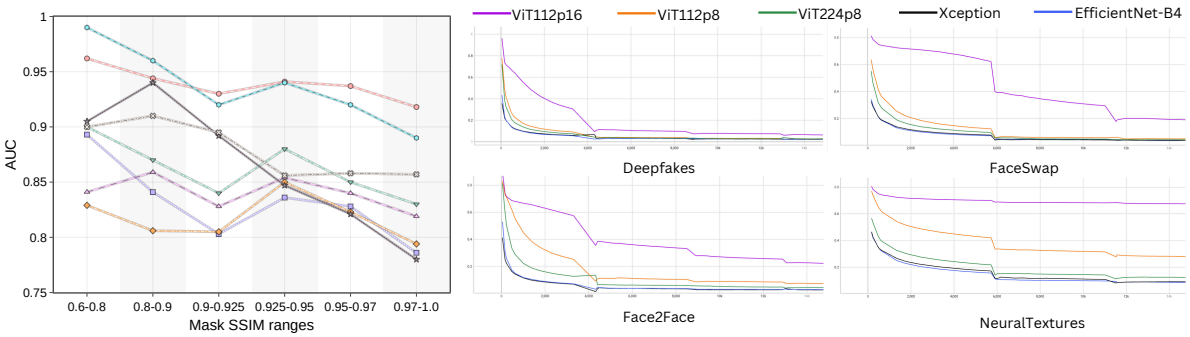
Figure 4.2 Examples are randomly selected to illustrate four types of deepfakes in the common FF++ [23] dataset. It includes Deepfakes (DF) [116], FaceSwap (FS) [118], Face2Face (F2F) [117], and NeuralTextures (NT) [119].

71, 72, 69, 70], it does not apply to the field of deepfake detection. So far, only a handful of transformer-based deepfake detection methods have been explored in the literature. For instance, ICT [37] utilizes a basic ViT to detect deepfakes by assessing identity consistency between inner and outer faces in suspect images. To capture multi-scale features, M2TR [34] and DFDT [35] simultaneously process inputs through multiple transformers with different patch sizes. However, despite training on large-scale datasets [131] and/or requiring costly computational resources, these methods still exhibit limited generalization performance compared to CNN-based approaches. This motivates us to investigate in Section 4.3 the underlying reasons explaining the unsuitability of ViTs for this use-case.

4.3 Vision Transformer and Deepfake Detection: Where is the Gap?

In this section, we investigate the specific challenges associated with the use of plain ViTs in deepfake detection. Our primary hypothesis is as follows: unlike CNNs, ViTs are inclined to focus more on global semantic representations [68, 84, 87, 85, 86], given their patch-based architecture and SA mechanism. Consequently, they struggle to effectively capture local features [84, 87], especially in higher layers [132], that are crucial for identifying subtle artifacts in HQ deepfakes, as highlighted in previous studies [78, 39, 48]. The plausibility of this assumption is investigated by conducting the following experiments described below.

Performance with respect to the quality of deepfakes. Here, our goal is to quantify the detection capability of ViTs and compare to other methods with respect to the quality of the



(a) (b)

Figure 4.3 Experiments to analyze the capability of transformer-based networks in deepfake detection. *(a)* Generalization performance comparison of base classifiers (ViT[68]+SBI[26](●), Swin[85]+SBI[26](●)) with two specialized transformer-based (FAViT [133](●), ForensicsAdapter [28](●)) and four CNN-based methods (LAA-Net[62]+SBI[26](●), CADDM[27](●), EfficientNet-B4[45]+SBI[26](●), Xception [46]+SBI [26](●)) across different ranges of Mask-SSIM [104]. Except for CADDM, FAViT, and ForensicsAdapter, which are trained with their own data, the other methods are trained with SBI synthesis pseudo-fakes. All methods adopt FF++ [23] for the training and are tested cross-dataset on CDF2 [80]. *(b)* Evolution of the training loss using ViT under different configurations (variation of input resolution and patch size), Xception [46] and EfficientNet-B4 [45], across four types of deepfakes in FF++ [23].

encountered deepfakes. To that aim, we compare in Figure 4.3a the performance of base classifiers (plain ViT [68] and Swin [85])[2] to specialized transformer-based methods (FAViT [133] and ForensiscAdapter [28]) and CNN-based methods (LAA-Net [62], CADDM [27], EfficientNet [45], and Xception [46]) across different ranges of Mask-SSIM [104] on the CDF2 [80] dataset. It should be noted that besides specialized networks [62, 27, 133, 28], two considered CNNs, i.e., EfficientNet [45] and XceptionNet [46], have been commonly chosen as the backbone for several SOTA methods [62, 27, 26, 39, 23, 31, 65, 44, 32, 134]. We clarify that for a fair comparison with the other approaches, except for CADDM, FAViT, and ForensicsAdapter trained with their own data algorithm, we train ViT, Swin, EfficientNet, and Xception using the same advanced data synthesis algorithm, i.e., SBI [26] used in SBI and LAA-Net[3].

It is observed from Figure 4.3a that the performance of ViT drops more significantly for higher Mask-SSIM values as compared to other methods. The performance of Swin, on the other hand, does not deteriorate as much, possibly due to its ability to extract low-level features with its local window design; however, it still shows less stability than LAA-Net and ForensicsAdapter. These observations provide initial support for our hypothesis.

Performance of ViTs with respect to the patch size and the type of deepfakes. We further investigate whether there exists a *correlation* between the patch size and the per-

[2]Note that the small(-S) architecture configuration is employed for both ViT and Swin. The FLOPs and the number of parameters are reported in Section 4.5.

[3]LAA-Net, CADDM, and SBI use EfficientNet-B4 as the default backbone.

formance of ViTs in deepfake detection. Specifically, we anticipate that smaller patch sizes would help model more localized artifacts. For that purpose, we train a plain ViT with several configurations, by varying both the patch size and the input resolution. Figure 4.3b depicts the evolution of the training loss through epochs. The notation ViTXpY in Figure 4.3b denotes an input resolution of X with a patch size of Y. We also compare ViTs to two widely-adopted CNNs, *i.e.*, XceptionNet and EfficientNet in this study. Both ViTs and CNNs are trained on four types of deepfakes in FF++ [23]: Deepfakes (DF) [116], FaceSwap (FS) [118], Face2Face (F2F) [117], and NeuralTextures (NT) [119] as shown in Figure 4.2. For the training setups, following the conventional splits [23], we train all models for 50 epochs, using 128 and 32 frames uniformly extracted from each video for training and validation, respectively. Hence, there are a total of 460800 and 22400 frames for each corresponding task. More details, *e.g.*, optimizer, learning-rate scheduler, etc., are provided in the supplementary materials.

As F2F and NT correspond to face reenactment manipulations while FS and DF represent face-swap approaches, it is more likely to observe more subtle artifacts in F2F and NT. When comparing ViT112p16 and ViT112p8, it can be seen that a ViT with smaller patches exhibits faster convergence compared to those with larger ones. Moreover, increasing the input resolution while conserving the same patch size (see ViT112p8 and ViT224p8) contributes to the amplification of the local information encoded in each patch, also resulting in faster convergence. In both cases, this convergence gap is even more pronounced for more subtle deepfake types such as F2F and NT, indicating the importance of locality in detecting deepfakes with subtle inconsistencies. However, reducing the patch size leads to a quadratically increasing complexity (*i.e.*, the FLOPs are 2.2G, 8.4G, and 33.6G for ViT112p16, ViT112p8, and ViT224p8, respectively). Additionally, it is to note that CNNs converge more rapidly compared to ViTs under different setups (*i.e.*, even with the smallest patch size). This further strengthens our hypothesis.

Hence, we posit that by proposing a mechanism that allows focusing on subtle artifact-prone regions, we can enhance the performance of ViTs for the task of deepfake detection. While some attempts have been made to introduce local ViTs such as Swin [85], we argue that this remains insufficient for effectively detecting deepfakes, especially HQ variants. As demonstrated for CNNs [62], incorporating local features does not guarantee that artifact-prone regions are effectively considered, highlighting the need to introduce attention strategies for explicitly focusing on localized artifacts.

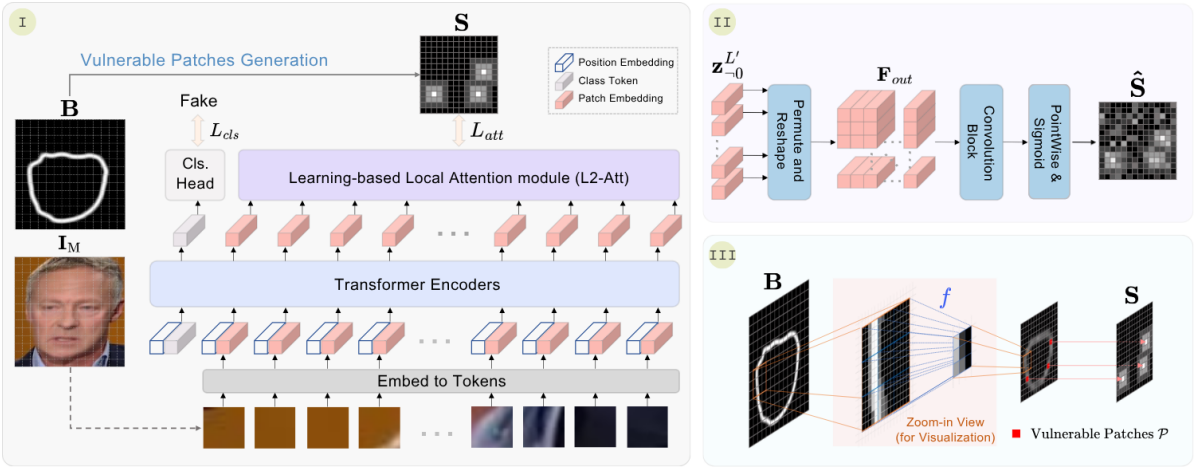


Figure 4.4 The proposed LAA-Former method. *(I)* The overall LAA-Former framework, *(II)* the L2-Att module, and *(III)* the ground-truth generation of vulnerable patches.

4.4 LAA-Former

Based on the observations made in Section 4.3, we propose LAA-Former, a Transformer-based approach integrating an attention mechanism that induces the modeling of subtle inconsistencies. Aside from its effectiveness, LAA-Former is not data-hungry and is computationally efficient. An overview of the proposed approach can be seen in Figure 4.4-I, consisting of a Vision Transformer (ViT) coupled with a Learning-based Local Attention module (L2-Att). L2-Att predicts the locations of vulnerable patches which generalize the notion of vulnerable points introduced in [62]. LAA-Former is trained using only normal data and utilizes blending-based data synthesis techniques [25, 26]. It can be noted that we utilize ViT to formulate our methodology for the sake of simplicity. However, our method is compatible with other transformer-based architectures, as demonstrated in Section 4.5. In what follows, we depict the different components of LAA-Former.

4.4.1 Vision Transformer (ViT)

Given an image $\mathbf{X} \in \mathbb{R}^{C \times H \times W}$ as input, we first reshape it into a sequence of non-overlapping flattened 2D patches, denoted as $\{\mathbf{x}_i \in \mathbb{R}^{C.P^2}$ with $i \in [\![1, N]\!]\}$, where (H, W) represents the input resolution, C denotes the number of channels, $P \times P$ indicates the size of an image patch, and $N = \frac{H \times W}{P^2}$ denotes the number of patches. The ViT linearly maps each $\mathbf{x}_i$ into a patch embedding $\mathbf{z}_i^0 \in \mathbb{R}^D$ using a learnable matrix $\mathbf{E} \in \mathbb{R}^{(C.P^2) \times D}$. Subsequently, a learnable embedding $\mathbf{x}_{cls} \in \mathbb{R}^D$ is prepended at the zero-index of embeddings $\mathbf{z}^0$ for the classification. Additionally, we use a learnable positional embedding $\mathbf{E}^{pos}$ to incorporate the

position information of patches. The aforementioned process is described as follows,

$$\mathbf{z}^0 = [\mathbf{x}_{cls}; \mathbf{x}_1\mathbf{E}; \mathbf{x}_2\mathbf{E}; \cdots; \mathbf{x}_N\mathbf{E}] + \mathbf{E}^{pos}, \tag{4.1}$$

where $\mathbf{E}^{pos} \in \mathbb{R}^{(N+1)\times D}$ and $\mathbf{z}^0 \in \mathbb{R}^{(N+1)\times D}$. Afterward, $\mathbf{z}^0$ is fed into several transformer encoder blocks. Similar to ViT [68], LAA-Former has L blocks, each one containing a multi-head self-attention (MHSA), Layernorm (LN), and a multi-layer perceptron (MLP). The feature extraction process is described as follows,

$$\begin{aligned} \mathbf{z}^l &= \mathrm{MHSA}(\mathrm{LN}(\mathbf{z}^{l-1})) + \mathbf{z}^{l-1}, \\ \mathbf{z}^{l'} &= \mathrm{MLP}(\mathrm{LN}(\mathbf{z}^l)) + \mathbf{z}^l, \end{aligned} \tag{4.2}$$

with $l \in [\![1, L]\!]$ and $\mathbf{z}^l, \mathbf{z}^{l'} \in \mathbb{R}^{(N+1)\times D}$. The extracted feature from the classification embedding $\mathbf{z}_0^{L'}$ is processed by a classification head composed of an MLP, resulting in the predicted category output $\hat{\mathbf{y}}$. In the task of deepfake detection, the categories consist of *real* or *fake*.

4.4.2 Attention to Vulnerable Patches

To overcome the challenge discussed in Section 4.3, we propose integrating the Learning-based Local Attention Module (L2-Att) within ViT to guide the network to focus on the most artifact-prone patches, termed *vulnerable patches*. In this section, we start with how we leverage the concept of blending-based data synthesis (as presented in Section 2.2.1) to define vulnerable patches, and detail their extraction process as well as the generation of the ground-truth data. Finally, we describe the architecture of the proposed L2-Att module. We note that LAA-Former is trained using only real data and blending-based pseudo-fakes.

Vulnerable Patches

Inspired by [62], we extend the definition of *vulnerable points* to *vulnerable patches*.

Definition 2. - *Vulnerable patches are the patches in an image that are the most likely to contain blending artifacts.*

This definition allows tailoring the idea of local vulnerable regions to the self-attention mechanism in ViTs, which occurs at the patch level. As in [62], we propose focusing on blending artifacts that are generic across different deepfake generation methods, thereby helping to achieve good generalization capabilities to unseen deepfakes. Inspired by [62], as vulnerable patches represent the subregions that are the most impacted by the blending, we assume that they correspond to the blending regions with the highest mixture magnitude

from the original images $\mathbf{I}_B$ and $\mathbf{I}_F$. To generate vulnerable patches, we leverage blending-based synthesis techniques. We believe that data synthesis algorithms are convenient to ensure generalization while significantly reducing the need for large-scale training datasets, typically needed for transformers. Hence, similar to [25], we calculate the blending boundary $\mathbf{B} = (b_{lm})_{l\in[\![1,H]\!],m\in[\![1,W]\!]}$ resulting from the combination of $\mathbf{I}_B$ and $\mathbf{I}_F$ as described in Eq. (2.2). A higher value of b_{rc} at a specific position $(r, c) \in [1, H] \times [1, W]$ indicates a more significant mixture between $\mathbf{I}_B$ and $\mathbf{I}_F$. We then apply to $\mathbf{B}$, a patching function f_1 that extracts N non-overlapping patches denoted as $\tilde{\mathbf{B}} = (\tilde{\mathbf{B}}_{ij})_{i,j\in[\![1,\sqrt{N}]\!]}$ of dimension $P \times P$ such that $\tilde{\mathbf{B}} = f_1(\mathbf{B})$. Finally, to quantify the vulnerability of each patch, a second function f_2 operating at the patch level is defined. In summary, the vulnerable patches $\mathcal{P}$ are obtained as follows,

$$\mathcal{P} = \arg\max_{(i,j)}(f_2(\tilde{\mathbf{B}}_{ij})), \tag{4.3}$$

For the sake of simplicity, we choose f_2 as the *max* operation, which implies that $f = f_2 \circ f_1$ corresponds to maxpooling. We also explore the *mean* operation in Section 4.5.3. In future works, we intend to investigate additional patch-based vulnerability functions. Note that $\mathcal{P}$ can also include more than one patch, since $\tilde{\mathbf{B}}$ can be maximal at several locations. This process is illustrated in Figure 4.4-III.

Ground Truth Generation for L2-Att

Finally, to obtain the ground truth to be compared to the output of L2-Att denoted as $\mathbf{S}$, we generate a weighted map $\mathbf{S}_p$ for each element $\mathbf{p} = (p_x, p_y) \in \mathcal{P}$. To take into account the neighborhood patches, we use an unnormalized Gaussian distribution to calculate $\mathbf{S}_p$ as follows,

$$\mathbf{S}_p(x, y) = e^{-\frac{(x-p_x)^2+(y-p_y)^2}{2\sigma^2}}, \tag{4.4}$$

where $(x, y) \in [[1, \sqrt{N}]]$ represents the spatial position, and the standard deviation σ is fixed to 1 by default. We obtain $\mathbf{S}$ by overlaying $\{\mathbf{S}_p\}_{p\in[[1,\text{card}(\mathcal{P})]]}$. The ground truth generation process is also illustrated in Figure 4.4-III. It can be noted that, for real data, $\mathbf{S}$ is set to a zero matrix.

Learning-based Local Attention Module (L2-Att)

Our hypothesis is that since the patch size can be too large relative to the area of artifacts, the features encoded within a patch embedding may hold insufficient information about them. Consequently, the implicit self-attention mechanism might overlook or miss important patches, as those containing forgeries can appear too similar to those without. Therefore, we

propose an explicit attention mechanism to ensure that the model pays more attention to these critical patches. To this end, by predicting the locations of vulnerable patches, L2-Att diffuses artifact-prone local information through the ViT. In particular, L2-Att first takes the patch embeddings $\mathbf{z}_{\neg 0}^{L'} \in \mathbb{R}^{N \times D}$ as input and processes them to produce spatial features as follows,

$$\begin{aligned} \mathbf{F} &= \text{Permute}(\mathbf{z}_{\neg 0}^{L'}), \quad \mathbf{F} \in \mathbb{R}^{D \times N}, \\ \mathbf{F}_{out} &= \text{Reshape}(\mathbf{F}), \quad \mathbf{F}_{out} \in \mathbb{R}^{D \times \sqrt{N} \times \sqrt{N}}, \end{aligned} \tag{4.5}$$

After that, $\mathbf{F}_{out}$ is fed into a convolution block (ConvBlock) with a kernel size of (3×3), followed by a pointwise convolution [135] and a sigmoid activation. The predicted weighted heatmap denoted as $\hat{\mathbf{S}}$ describing the presence probability of vulnerable patches is obtained as follows,

$$\hat{\mathbf{S}} = \sigma(\text{PointWise}(\text{ConvBlock}_{3\times 3}(\mathbf{F}_{out}))), \tag{4.6}$$

where $\hat{\mathbf{S}} \in \mathbb{R}^{1 \times \sqrt{N} \times \sqrt{N}}$. A detailed illustration of the L2-Att module can be seen in Figure 4.4-II.

4.4.3 Training Objective

To train the network, we optimize two losses, namely the Binary Cross Entropy (BCE) loss for classification denoted as $L_{cls}(\hat{\mathbf{y}}, \mathbf{y})$, and the regression loss related to the prediction of vulnerable patches locations, denoted as $L_{att}(\hat{\mathbf{S}}, \mathbf{S})$. Therefore, the total loss L is defined as follows,

$$L = L_{cls} + \lambda_{att} L_{att}, \tag{4.7}$$

where λ_{att} is a balancing factor between the two losses. Similarly in LAA-Net, we employ the focal loss [95] to compute $L_{att}(\hat{\mathbf{S}}, \mathbf{S})$.

4.5 Experiments

4.5.1 Setups

Datasets. As in [25, 26, 136, 64, 31, 27, 36], we use the FaceForensics++ (FF++) [23] dataset for training. Only real data and blended-based pseudo-fakes [25, 26] are employed for training. We consider both the *in-dataset* and the *cross-dataset* protocols by testing on FF++ and other benchmarks including Celeb-DF (CDF1, CDF2) [80], WildDeepfake (DFW) [82], DeepFakeDetection (DFD) [83], Deepfake Detection Challenge Preview (DFDCP) [81], and

Deepfake Detection Challenge (DFDC) [88], respectively. More details about the datasets are provided in the supplementary materials.

Data Processing. Following the splitting convention [23], we sample 128, 32, and 32 frames from each video for training, validation, and testing respectively. This significantly reduces the amount of training data compared to previous ViT-based works [37, 35, 34]. The facial regions are extracted using RetinaFace [137]. To align two images for generating pseudo-fakes [25, 26], Dlib [120] is used as a landmark extractor. Further details are provided in the supplementary materials.

Training. We train LAA-Former for 200 epochs using the AdamW [138] optimizer with a weight decay of 10^{-4} and a batch size of 32. The weights are initialized using a ViT [98, 68] pretrained on ImageNet [121]. The learning rate maintains at 5×10^{-5} during the first quarter of iterations, then gradually decays to zero over the remaining epochs. We freeze the backbone (*i.e.*, ViT without the head) for the first 6 epochs, before training all layers. As a regularizer, we integrate LabelSmoothing [124] with the loss function (Eq. 4.7). For each video in the batch data, we dynamically select only m frames, with $m = 8$ or $m = 16$ when using SBI [26] or BI [25], respectively. Data augmentation techniques include color jittering, random cropping, scaling, horizontal flipping, Gaussian noise, blurring, and JPEG compression. All experiments are conducted on 4 NVIDIA A100 GPUs.

Evaluation Metrics. To compare LAA-Former with existing works, we employ two common evaluation metrics: the Area under the Curve (AUC) and the Average Precision (AP). For a fair comparison, similar to previous studies [27, 33, 26, 65, 36, 54], we report the video-level performance.

Architectures. We utilize two variants of LAA-Former that we call LAA-Former-S and LAA-Former-B. By default, we use the lightweight LAA-Former-S, where $H = W = 112$ and $P = 8$. LAA-Former is based on the vanilla vision transformer, *i.e.*, ViT [68]. However, although LAA-Former is based on ViT, we also assess its applicability using another transformer architecture, namely *LAA-Swin*, which is based on Swin [85]. Similarly to LAA-Former, we consider two variants: LAA-Swin-S and LAA-Swin-B. Further details are provided in the supplementary materials.

4.5.2 Comparison with State-of-the-art

In this section, we compare the performance of LAA-Former to SOTA methods. In all tables, **bold** and underlined results highlight the best and the second-best performance, respectively.

Overall Cross-dataset Evaluation. The overall comparison with SOTA methods under the cross-dataset setup can be seen in Table 4.1. As observed, LAA-Former generalizes better

Method	Training set		Test set (%)											
	Real	Fake	CDF1		CDF2		DFW		DFD		DFDCP		DFDC	
			AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
Xception [23]	✓	✓	58.81	65.59	61.18	66.93	65.29	55.37	89.75	85.48	69.90	91.98	45.60	-
FaceXRay [25]	✓	✓	80.58	73.33	-	-	-	-	95.40	93.34	80.92	72.65	-	-
LocalRL [63]	✓	✓	-	-	78.26	-	-	-	89.24	-	76.53	-	-	-
Multi-attentional [39]	✓	✓	69.14	74.03	68.26	75.25	73.56	73.79	92.95	96.51	83.81	96.52	-	-
RECCE [32]	✓	✓	49.96	63.04	70.93	70.35	68.16	54.41	98.26	79.42	80.93	92.76	-	-
SFDG [48]	✓	✓	-	-	75.83	-	69.27	-	88.00	-	73.63	-	-	-
EIC+IIE [114]	✓	✓	-	-	83.80	-	-	-	93.92	-	81.23	-	-	-
CADDM [27]	✓	✓	89.36	93.25	93.88	91.12	74.48	75.23	**99.03**	**99.59**	-	-	73.85	-
UCF [44]	✓	✓	-	-	82.4	-	-	-	94.5	-	80.5	-	-	-
M2TR [34]	✓	✓	-	-	68.2	-	-	-	-	-	-	-	-	-
DFDT [35]	✓	✓	-	-	88.3	-	-	-	-	-	76.1	-	-	-
TALL-Swin [36]	✓	✓	-	-	90.79	-	-	-	-	-	76.78	-	-	-
LSDA [139]	✓	✓	-	-	91.10	-	-	-	-	-	77.00	-	-	-
ICT [37]	✓		81.43	-	85.71	-	-	-	84.13	-	-	-	-	-
SBI [26]	✓		92.53	79.91	93.18	85.16	67.47	55.87	97.56	92.79	86.15	93.24	72.42	-
LAA-Former	✓		**97.25**	**98.36**	**94.45**	**97.15**	**81.74**	**83.72**	96.12	98.31	**96.30**	**99.50**	**78.91**	**80.01**

Table 4.1 Cross-dataset evaluation. Comparisons in terms of AUC (%) and AP (%) on six benchmark datasets including CDF1 [80], CDF2 [80], DFW [82], DFD [83], DFDCP [81], and DFDC [88]. **Bold** and underlined results highlight the best and the second-best performance, respectively. The results for comparisons are directly extracted from the original papers.

Method	Training data		#Params	FLOPs	Test set AUC(%)	
	Real	Fake			FF++	CDF2
ViT-B [36]	✓	✓	84M	55.4G	-	82.33
ViT-B+TALL [36]	✓	✓	84M	55.4G	-	86.58
LAA-Former-S	✓		23M	8.9G	97.67	94.45
LAA-Former-B	✓		91M	35.8G	97.76	95.21
Swin-B [36]	✓	✓	86M	47.5G	-	83.13
Swin-B+TALL [36]	✓	✓	86M	47.5G	99.87	90.79
LAA-Swin-S	✓		55M	6.5G	99.89	94.48
LAA-Swin-B	✓		91M	11.5G	**99.94**	**95.44**

Table 4.2 Performance of Transformer-based approaches. Comparisons in terms of AUC (%) on CDF2 [80], and FF++ [23]. All methods are trained on FF++ [23] and tested on the other datasets.

than SOTA methods, including those trained with fake data, those trained with pseudo-fakes (without fake data), and transformer-based models such as ICT [37], M2TR [34], and DFDT [35], although our model is trained with a significantly lower amount of data.

Comparison with Transformer-based Approaches. Table 4.2 compares LAA-Former to Transformer-based methods in terms of model size (#Params), computational cost (FLOPs), and AUC on FF++ and CDF2. Even with the smaller variant (i.e., -S), our method outperforms vanilla ViT-B and Swin-B. This highlights the effectiveness of the proposed L2-Att module, resulting in a lightweight model. Moreover, regardless of the transformer backbone, LAA-Former and LAA-Swin generalize better than TALL [36] despite their simplicity. It should be noted that the ViT-B and the Swin-B results are directly recovered from [36], which means that the backbone configurations might be different.

Comparison with Blending-based Synthesis Approaches. Table 4.3 and Table 4.4 com-

Method	Params	FLOPs	Test set AUC(%)		
			CDF2	DFDCP	DFDC
FaceXRay+BI [25]	41.23M	-	79.5	65.5	-
ICT [37]+BI [25]	21.45M	8.4G	85.71	-	-
LAA-Net [62]+BI [25]	27.13M	11.6G	86.28	69.69	64.18
LAA-Former+BI [25]	22.77M	8.9G	**90.34**	**78.71**	**76.84**

Table 4.3 Performance using BI [25]. Comparisons in terms of AUC (%) using cross-dataset evaluation on CDF2 [80], DFDCP [81], and DFDC [88].

Method	Params	FLOPs	Test set AUC(%)		
			CDF2	DFDC	DFDCP
SBI [26]	19.34M	8.3G	93.18	72.42	86.15
LAA-Net [62]+SBI [26]	27.13M	11.6G	**95.40**	72.43	86.94
LAA-Former+SBI [26]	22.77M	8.9G	94.45	**78.91**	**96.30**

Table 4.4 Performance using SBI [26]. Comparisons in terms of AUC (%) using cross-dataset evaluation on CDF2 [80], DFDCP [81], and DFDC [88].

pare our method to SOTA when using Blending-Images (BI) [25] and Self-Blended-Images (SBI) [26], respectively. As observed, we generally achieve better performances than SOTA methods using the same blending algorithm on CDF2, DFDC, and DFDCP datasets, despite training on a drastically lower number of data as compared to other ViT-based approaches such as ICT [37]. Moreover, LAA-Net [62], as a CNN-based method, requires a larger model size as compared to our lightweight LAA-Former.

Robustness to Unseen Perturbations. Since deepfakes can be altered by common perturbations on social media. Following the evaluation setting of [60], experiments evaluating the robustness of LAA-Former against five types of perturbation on FF++ [23] are reported in Table 4.5. We can observe the superiority of LAA-Former and LAA-Swin as compared to existing methods.

Qualitative Results. We show Grad-CAMs [127] in Figure 4.5 to visualize the image regions

Method	Training set		Perturbation set					
	Real	Fake	Saturation	Contrast	Block	Noise	Pixel	Avg.
Xception [46]	✓	✓	99.3	98.6	99.7	53.8	74.2	85.12
FaceXray [25]	✓	✓	97.6	88.5	99.1	49.8	88.6	84.72
CNN-aug [91]	✓	✓	99.3	99.1	95.2	54.7	91.2	87.90
LipForensics [54]	✓	✓	99.9	99.6	87.4	73.8	95.6	91.26
SBI [26]	✓	×	92.0	92.3	92.2	62.2	79.1	83.56
LSDA [139]	✓	✓	98.7	94.4	98.3	66.4	90.7	89.70
LAA-Net [62]	✓	×	**99.96**	**99.96**	**99.96**	53.90	**99.80**	90.72
LAA-Former	✓	×	98.04	95.96	97.02	75.28	91.14	91.49
LAA-Swin	✓	×	99.79	99.77	99.92	**81.60**	94.88	**95.19**

Table 4.5 Robustness to unseen perturbations.

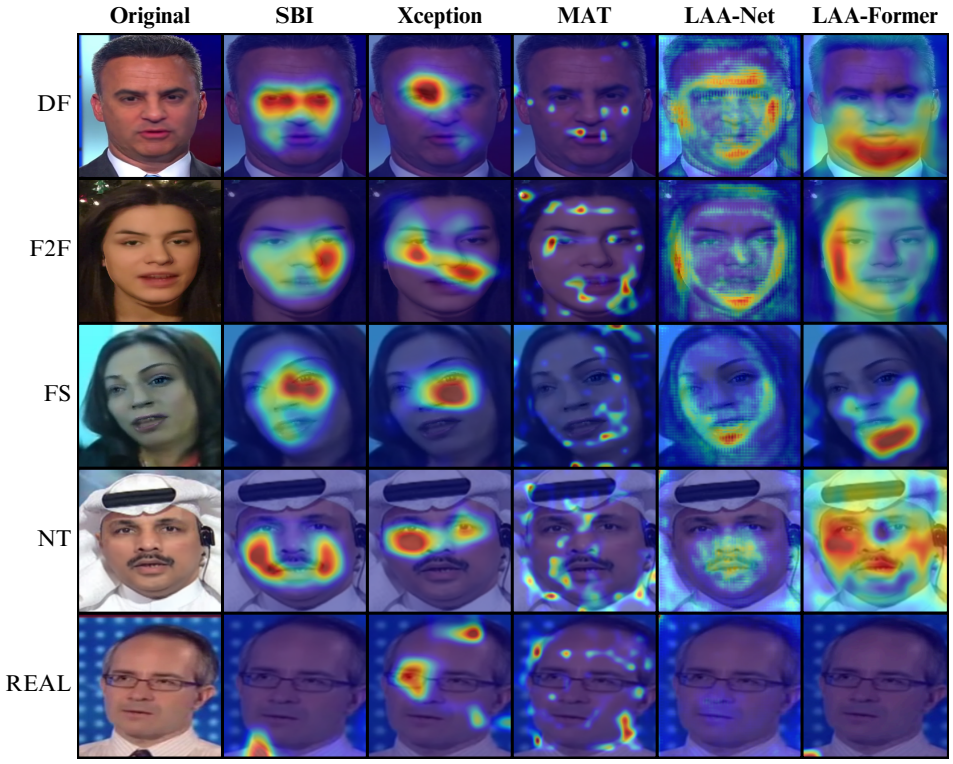


Figure 4.5 Visualization of saliency maps: on different types of manipulation from FF++ [23]. LAA-Former and LAA-Net are compared to SBI [26], Xception [23], and MAT [39].

Res. & Pat.	Test set AUC (%)						
	FF++	CDF2	DFD	DFW	DFDCP	DFDC	Avg.
112P16	81.83	85.26	73.56	76.87	93.28	72.53	80.56
112P8	97.67	94.45	96.12	81.74	96.30	78.91	90.87
224P8	**99.93**	**96.84**	**99.54**	**82.11**	**96.99**	**79.01**	**92.40**

Table 4.6 Effect of input resolution and patch size.

of deepfake images that are highlighted by LAA-Former, LAA-Net, SBI [26], Xception [23], and Multi-attentional (MAT) [39] on FF++ [23]. In general, attention-based methods such as MAT [39], LAA-Net, and LAA-Former tend to focus more on localized regions. However, in some cases MAT [39] concentrates on irrelevant areas, such as the background or inner facial regions. In contrast, thanks to the explicit attention mechanism implemented in L2-Att, LAA-Former consistently identifies blending artifacts and exhibits a strong ability to focus on mouth-rendered Neural Textures (NT) without having seen those during training.

4.5.3 Ablation Study

Effect of Patch Size. To further investigate the impact of the patch size on the performance, we provide, in addition to the experiment reported in Figure 4.3b, the obtained cross-evaluation results of LAA-Former when varying the input size and the patch size in Table 4.6. It can be

Model	Lr	L2-Att	Test set AUC (%)				
			FF++	CDF2	DFD	DFDC	Avg.
ViT [68] (w/ SBI)	1×10^{-3}	×	68.86	67.29	60.58	61.32	64.51
LAA-Former		✓	73.52	71.78	65.34	62.51	68.29 (↑3.78)
ViT [68] (w/ SBI)	5×10^{-4}	×	75.68	72.99	57.19	62.38	67.06
LAA-Former		✓	80.08	87.45	65.62	71.04	76.05 (↑**8.99**)
ViT [68] (w/ SBI)	1×10^{-4}	×	95.99	89.27	89.71	78.50	88.36
LAA-Former		✓	96.14	**95.20**	91.14	78.85	90.33 (↑1.97)
ViT [68] (w/ SBI)	5×10^{-5}	×	97.48	92.62	95.72	77.35	90.79
LAA-Former		✓	**97.67**	94.45	**96.12**	**78.91**	**91.79** (↑1.00)
Swin [85] (w/ SBI)	1×10^{-3}	×	99.48	81.37	96.05	66.69	85.89
LAA-Swin		✓	99.88	94.91	97.17	74.33	91.57 (↑**5.68**)
Swin [85] (w/ SBI)	5×10^{-4}	×	**99.98**	89.00	98.94	71.15	89.76
LAA-Swin		✓	99.97	**95.43**	99.58	74.97	92.49 (↑2.73)
Swin [85] (w/ SBI)	1×10^{-4}	×	99.89	90.18	99.54	73.38	90.74
LAA-Swin		✓	**99.98**	93.87	99.62	**77.92**	92.85 (↑2.11)
Swin [85] (w/ SBI)	5×10^{-5}	×	99.75	90.89	99.59	74.20	91.11
LAA-Swin		✓	99.89	94.48	**99.68**	77.47	**92.88** (↑1.77)

Table 4.7 Ablation study of LAA-Former's components. Models are trained on FF++ [23], and test cross-dataset on [80, 83, 88].

observed that either reducing the patch size or increasing the input resolution contributes to improving the generalization performance of the model. This confirms that the patch size and the input size implicitly affect the extraction of localized features that characterize deepfakes.

Ablation study of LAA-Former's components. To validate the impact of L2-Att module, we compare ViT [68]/Swin [85] and LAA-Former (ViT+L2-Att)/LAA-Swin (Swin+L2-Att) - all trained with SBI [26]. The results on several datasets [80, 83, 88] are presented in Table 4.7. As shown, L2-Att consistently contributes to the enhancement of both ViT and Swin, confirming the relevance of the proposed explicit attention mechanism.

Vulnerable Points versus Vulnerable Patches. To demonstrate the compatibility of vulnerable patches (VPatch) as compared to vulnerable points (Vpoint) with transformers, we report in Table 4.8 the obtained results when replacing VPatch with VPoint within LAA-Former and LAA-Swin. It can be observed that the use of VPatch not only results in better performance but also maintains a lower computational cost as compared to VPoint. We note that the higher number of parameters and FLOPs associated with using VPoint is caused by a decoder designed to locate these points. Meanwhile, VPatch does not incur any decoders, making it more computationally efficient.

Selection of f_2. Table 4.9 compares two aggregation functions f_2 defined in Eq. (4.3) coupled with LAA-Former: *mean* and *max* operations. In both cases, the stability of the results can be seen. By default, we select the *max* operation as it gives slightly better results. In future works, we plan to investigate further selections of f_2, e.g., learnable alternatives.

Model	Target	Params	FLOPs	Test set AUC(%)			
				CDF2	DFD	DFDC	Avg.
LAA-Former	V-Point	23.61M	9.4G	93.39	93.70	77.71	88.26
	V-Patch	22.77M	8.9G	**94.45**	**96.12**	**78.91**	**89.83**(↑1.67)
LAA-Swin	V-Point	57.25M	8.1G	94.25	99.30	74.99	89.51
	V-Patch	54.89M	6.5G	**94.48**	**99.68**	**77.47**	**90.54**(↑1.03)

Table 4.8 Vulnerable Patches (V-Patch) vs. Vulnerable Points (V-Point).

f_2	Test set AUC (%)				
	CDF2	DFD	DFW	DFDC	Avg.
mean	94.16	95.03	81.02	78.28	87.12
max	**94.45**	**96.12**	**81.74**	**78.91**	**87.81**(↑0.69)

Table 4.9 Selection of f_2.

Learning Rate Sensitivity. In addition to the variations of patch size analyzed in Section 4.5.3 and Section 4.3, we further hypothesize that the initial learning rate (Lr) may also affect the learning capability of transformer architectures, especially when training with HQ pseudo-fakes such as SBI [26], which encode subtle forgeries. To investigate this, we keep the training protocol fixed and vary Lr for LAA-Former, LAA-Swin, and their plain counterparts. The evaluation results on four datasets [23, 80, 83, 88] are reported in Table 4.7. We observe that, for larger Lr values, the plain ViTs struggle to learn robust representations, leading to poor cross-dataset generalization. By contrast, the hierarchical design of Swin allows it to capture localized features more effectively and thus maintain relatively stable performance across all four datasets. Interestingly, integrating L2-Att consistently improves the generalizability of both ViT and Swin across all tested Lr values, with the gains being particularly noticeable at higher learning rates. This further highlights the impact of L2-Att in the context of deepfake detection.

Impact of Loss Balancing Factor λ_{att}. The weight λ_{att} defined in Eq. (4.7) is set empirically to 10 as it yields the best performance on average. We report the results using different values of λ_{att} within LAA-Former in Table 4.10. It can be observed that the generalization across four testing benchmarks remains robust regardless of the value of λ_{att}.

λ_{att}	Test set AUC (%)			
	CDF2	DFD	DFDC	Avg.
1	93.67	94.62	77.91	88.73
10	<u>94.45</u>	**96.12**	**78.91**	**89.83**
100	**94.96**	<u>94.88</u>	<u>78.58</u>	<u>89.47</u>

Table 4.10 Impact of loss balancing factor λ_{att} in Eq. (4.7).

4.6 Conclusion

In this chapter, we identify the shortcomings of plain ViTs in capturing localized features, explaining its limited adoption in the field of deepfake detection. To address this, we introduce LAA-Former, a lightweight ViT-based framework for generalizable deepfake detection. LAA-Former incorporates L2-Att, an attention strategy directing the focus toward local inconsistency-prone regions, thus enhancing the detection capabilities of transformers. Leveraging the synthetic data generation and the proposed attention mechanism mitigates the need for extensive training data and large models. Detailed experiments demonstrate the superiority of LAA-Former over state-of-the-art methods. In next chapter, we will investigate strategies to extend vulnerability-driven strategies for capturing spatio-temporal artifacts and introducing robust deepfake detection methods at the video level.

Chapter 5

Vulnerability-Aware Spatio-Temporal Learning for Generalizable Deepfake Video Detection

This chapter presents the third proposed detection framework of the thesis, FakeSTormer, which targets generalizable deepfake video detection through vulnerability-aware spatio-temporal learning. While chapter 3 and chapter 4 focused on image-level detectors based on CNNs and transformers, this chapter extends the vulnerability-driven, multi-task design paradigm to the video domain, where spatial and temporal artifacts are tightly intertwined and often subtle in high-quality forgeries. Building on the transformer foundations introduced in chapter 2, FakeSTormer combines a revisited TimeSformer backbone with two auxiliary branches that explicitly model temporal and spatial vulnerabilities, supported by a Self-Blended Video (SBV) data synthesis strategy for generating high-quality (HQ) pseudo-fakes. The remainder of the chapter first reviews related work on video-based deepfake detection, then details the proposed methodology, and finally reports extensive cross-dataset experiments and analyses.

5.1 Introduction

With the advances in generative modeling [1, 12], deepfake videos have become alarmingly realistic. Despite their interest in several applications, such as entertainment and education,

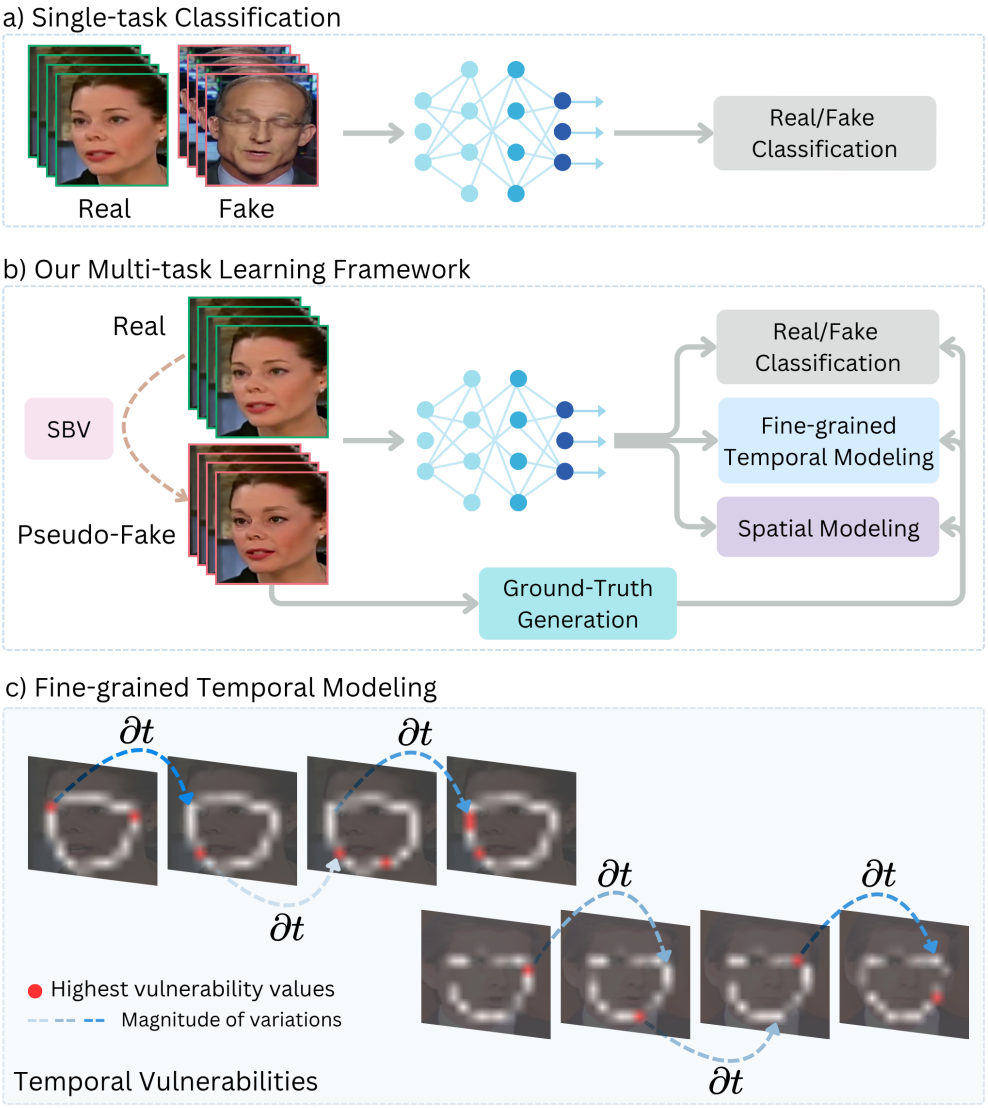


Figure 5.1 a) Traditional video-based methods [52, 56, 54, 36, 53, 51, 55, 50] versus b) the proposed multi-task learning framework; c) Visualization of the temporal vulnerabilities. Note that only some temporal locations are shown.

this type of technology also raises societal concerns [17, 20, 19]. There is therefore an urgency for developing effective deepfake detection methods.

In the literature, several deepfake detection techniques aim to model spatial artifacts by treating each frame independently [23, 25, 39, 31, 32, 26, 27, 64, 140, 62]. While this is reasonable when dealing with frame-level generation methods [117, 141, 142], it becomes less adequate in the presence of video-level manipulation techniques [3, 4, 49], where temporal and spatial artifacts are *intertwined*.

For that reason, researchers have explored video-based deepfake detection methods capable of modeling spatio-temporal artifacts [50, 51, 52, 53, 54, 55, 56]. Those methods mainly rely on a deep neural network formed by a single binary classification branch that is trained using a fixed dataset with real and fake data (see Figure 5.1-a). As a result, they suffer

from two main limitations, namely: (1) **The lack of generalizability -** As highlighted in [62, 25, 26], models trained with a standalone binary classifier tend to overfit the type of deepfakes they are trained on, resulting in poor generalization to unseen manipulations; (2) **The lack of robustness to high-quality (HQ) deepfake videos -** The quality of deepfake videos is improving continuously, resulting in subtle spatio-temporal artifacts. As such, vanilla single-branch architectures trained solely with binary supervision fail to fully capture them, necessitating the design of appropriate attention mechanisms.

To address the generalization issue, video-level data synthesis approaches [58, 57, 29] have been introduced to encourage models to learn more generic representations. However, these methods usually simulate exaggerated temporal variations that are inherently different from artifacts in hyper-realistic deepfake videos. On the other hand, to model localized spatio-temporal inconsistencies, some recent methods [53, 59] have used dedicated architectures integrating an implicit attention mechanism. Nevertheless, these models still rely solely on a binary classifier with no guarantee of extracting artifact-prone *fine-grained* traces.

Interestingly, in image-based deepfake detection, it has been recently demonstrated that the use of a tailored multi-task learning framework for explicitly attending artifact-prone small regions coupled with a subtle data synthesis strategy can be a way to enhance generalization and, at the same time, robustness to high-quality deepfakes [62, 140]. Nevertheless, such an approach has been disregarded in the field of video-level deepfake detection, as its extension to the video level is not straightforward. In particular, it would necessitate the characterization of subtle temporal artifacts that are inherently different from spatial ones, within both the multi-task learning framework and the data synthesis.

In this chapter, we redefine deepfake video detection as a fine-grained detection task by proposing a multi-branch network that leverages synthesized data and incorporates specialized learning objectives specifically targeting both subtle spatial and temporal artifacts. As shown in Figure 5.1-b, a novel multi-task learning framework, termed FakeSTormer is introduced. It is formed by two auxiliary parallel branches in addition to the standard classification head, namely: **(1) a regression temporal branch** incorporating an explicit attention that aims at locating the vulnerability-prone temporal locations. It has been shown that regressing spatial vulnerabilities in specific points [62] or patches [140] can help improve the generalizability of a deepfake detector model. We refer to the definitions given in [140, 62] which describe: "*vulnerable patches/points as the patches/points that are the most likely to embed blending artifacts*". To generalize this concept to the temporal domain, we propose locating temporal high changes in spatial vulnerable patches (see Figure 5.1-c). **(2) a spatial branch** to ensure a balance between the spatial and the temporal domains. In fact, detecting spatial artifacts

in addition to temporal ones is crucial [56, 53]. For that purpose, we propose predicting frame-wise spatial vulnerabilities.

To create hand-free ground truths for the proposed branches, we introduce a HQ video-level data synthesis algorithm, called "Self-Blended Video (SBV)", inspired by "Self-Blended Image (SBI) [26]", enforcing temporal coherence using two proposed modules on top of SBI (detailed in Section 5.3.1). Our experiments demonstrate that simply training a baseline classification model on SBV enables achieving on par performance w.r.t.state-of-the-art (SOTA), highlighting the effectiveness of SBV. Finally, for enhancing spatial and temporal modeling, we revisit the TimeSformer [89] architecture that we use as our backbone. In particular, we leverage TimeSformer's decomposed temporal and spatial attention on embedded patches, appending classification tokens for each frame and for each patch across frames, rather than a single token for the entire video. These classification tokens are then used within the spatial and classification heads, while the embedded patches are used within the temporal head. Extensive experiments on several well-known deepfake detection benchmarks show that our method outperforms the existing SOTA approaches.

Contributions. In summary, we propose in this chapter:

- A novel multi-task learning framework using only real data for fine-grained video-based deepfake detection.
- Two auxiliary branches that capture both temporal and spatial vulnerabilities, that are fined-grained by definition.
- A video-level data synthesis technique called SBV that generates high-quality pseudo-fakes and is supported by a vulnerability-driven cutout augmentation strategy to avoid overfitting specific artifact-prone regions.
- A revisited version of the TimeSformer [89], specifically tailored for the proposed video-based deepfake detector.
- Extensive experiments and analyses conducted on several challenging datasets.

Chapter organization. Section 5.2 reviews related work on video deepfake detection. Section 5.3 describes the proposed FakeSTormer method. Section 5.4 presents experiments and results. Finally, Section 5.5 concludes with future work.

5.2 Related Work

Video-based Deepfake Detection. As highlighted in [52, 56], using a naive spatio-temporal binary classification model for video-level deepfake detection can lead the model to overfit obvious artifacts, resulting in poor generalization to unseen manipulations. To address this, FTCN [52] proposes a fully temporal convolution network by reducing the spatial kernel size to one, hence decreasing the likelihood of focusing only on spatial artifacts. LipForensics [54] considers solely the mouth region, while spatio-temporal dropout [55] randomly removes parts of the input frames in both spatial and temporal domains. AltFreezing [56] separates convolution layers into spatial and temporal ones, failing to model long-term dependencies. Instead of using convolution layers, ISTVT [53] utilizes a video-based Vision Transformer [68] with self-attention to extract longer-range correlations. Meanwhile, [143] decomposes features into spatial and temporal components. TALL [36] employs an image-level deepfake detector by converting video frames into a thumbnail layout. Despite being promising, most of the aforementioned methods solely rely on a single binary classifier that implicitly guides the feature extraction. As highlighted in the literature on image-based deepfake detection [25, 31, 62, 27], this approach might lead to overfitting specific artifacts present in training datasets. Moreover, the absence of an explicit attention mechanism to spatio-temporal artifact-prone regions can lead to poor robustness to high-quality artifacts.

Data Synthesis. A highly effective approach for enhancing the generalizability of deepfake detectors is training models with synthesized data. While frame-level solutions have been extensively studied [26, 25, 33, 31, 27], video-level augmentations remains relatively underexplored. In recent works, STC [58] generates pseudo-fake samples via time-shuffling, VB [29] perturbs landmarks per frame without imposing temporal coherence, while ST-SBV [57] injects temporal artifacts through random face scaling and blurring over time. However, these methods often introduce exaggerated temporal distortions that differ from HQ deepfakes typically exhibiting finer temporal inconsistencies.

5.3 Methodology

Let $\mathcal{V} \triangleq \cup_{i=1}^{N}\{(\mathbf{X}_i, y_i)\}$ be a training dataset formed by N videos, where $\mathbf{X}_i$ denotes the i^{th} video sample and y_i its associated label indicating whether the clip is real ($y_i = 0$) or fake ($y_i = 1$). Traditional methods [56, 50, 36, 54, 52, 53, 55, 51] aim to learn jointly a feature extractor $\Phi : \mathcal{V} \mapsto \mathcal{F}$ and a binary classifier $f : \mathcal{F} \to \{0, 1\}$ by minimizing the standard binary cross-entropy (BCE) loss $\mathcal{L}_{BCE}(f(\Phi(\mathbf{X}_i)), y_i)$ using the entire training set $\mathcal{V}$, with $\mathcal{F}$

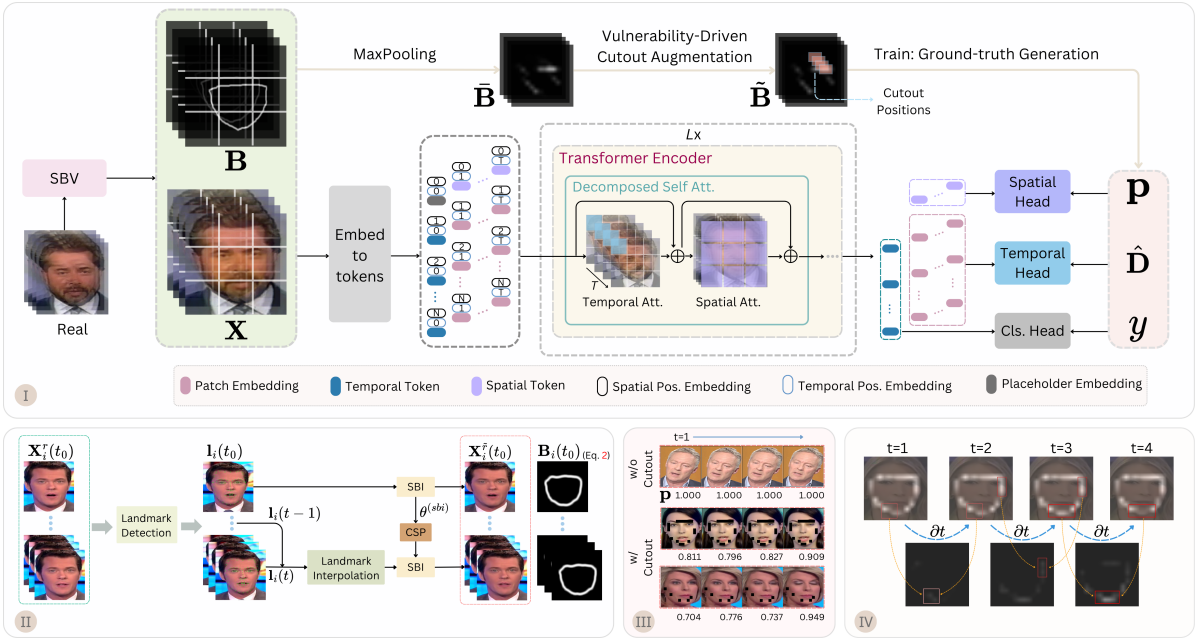


Figure 5.2 **I) Overview of the proposed framework:** Our multi-task learning framework, FakeSTormer, consists of three branches, i.e., the temporal branch (h), the spatial branch (g), and the standard classification branch (f). Those branches are specially designed to facilitate the disentanglement learning of spatial-temporal features. The hand-free ground-truth data to train the framework are generated based on our proposed video-level data synthesis algorithm coupled with a vulnerability-driven Cutout strategy. **II) Overview of generating a self-blended video:** It contains two main components, including a landmark interpolation module (*LI*) and the consistent utilization of synthesized parameters (*CSP*). **III) Examples of pseudo-fake videos:** with(*w/*) and without(*w/o*) vulnerability-driven Cutout and their corresponding soft labels. We apply the Cutout data augmentation at the same spatial locations throughout video frames. **IV) Extraction of temporal vulnerabilities:** We compute derivatives of the spatial vulnerabilities over time.

being the learned feature space. As previously discussed in [25, 62, 26] and also highlighted in Section 5.1, such a strategy might lead to poor generalization capabilities to unseen generation methods while providing only binary outputs that are not interpretable.

To tackle these issues, inspired by the literature on image-level deepfake detection [25, 62, 31, 33, 27], we introduce a novel multi-task learning framework called FakeSTormer that only relies on the real data subset denoted as $\mathcal{V}^r \subset \mathcal{V}$. Specifically, in addition to the binary classifier f, our framework includes two additional branches $h : \mathcal{F} \rightarrow \mathcal{H}$ and $g : \mathcal{F} \rightarrow \mathcal{G}$ that aim at triggering the learning of localized temporal and spatial artifact-prone features, respectively, through relevant auxiliary tasks. Note that $\mathcal{H}$ and $\mathcal{G}$ denote respectively the output spaces of h and g. The proposed branches are depicted further in Section 5.3.2. To provide ground truth to those branches and at the same time avoid overfitting to specific manipulations, we apply to each video belonging to $\mathcal{V}^r$ a data synthesis method described in Section 5.3.1, resulting in a pseudo-fake subset denoted as $\mathcal{V}^{\tilde{r}}$. Hence, our framework is trained using $\tilde{\mathcal{V}} \triangleq \{\mathcal{V}^r \cup \mathcal{V}^{\tilde{r}}\}$.

5.3.1 Video-Level Data Synthesis and Augmentation

Self-Blended Video. Blending-based data synthesis methods have demonstrated great performance in image-based deepfake detection [26, 25, 62, 31, 64, 33, 140]. In fact, as the blending step is common to different manipulation types, they contribute to the improvement of the generalization aspect in deepfake detection [62, 25]. Nevertheless, such an approach has been overlooked in the context of video-based deepfake detection. Hence, we propose to extend blending-based data synthesis to the video level. In particular, we revisit Self-Blended Image (SBI) [26], given its ability to produce high-quality pseudo-fake images. The proposed data synthesis approach, termed Self-Blended Video (SBV), is constituted of two main components building on top of SBI, *i.e.*, a Consistent Synthesized Parameters (CSP) module followed by a Landmark Interpolation module (LI) for preserving the temporal coherence of synthesized videos, which is essential for producing high-quality synthesized videos.

Specifically, given a real video $\mathbf{X}_i^r \in \mathcal{V}^r$ formed by T consecutive frames, we start by extracting a set of 2D landmarks $\mathbf{L}_i(t) = \{\mathbf{l}_{ij}(t)\}_{1 \leq j \leq n}$ at each instant t from $\mathbf{X}_i^r(t)$, where n refers to the number of landmarks and $\{\mathbf{l}_{ij}(t)\} \in \mathbb{R}^2$. Then, we apply SBI to the 1$^{\text{st}}$ video frame denoted as $\mathbf{X}_i^r(t_0)$ to obtain a pseudo-fake image, i.e., $\mathbf{X}_i^{\tilde{r}}(t_0)$ and a blending mask $\mathbf{M}_i(t_0)$. All related blending parameters $\theta^{(sbi)}$ (e.g., ConvexHull type, Mask deformation kernels, blending ratio, etc.) are then conserved for synthesizing the remaining video frames. However, using those parameters solely cannot guarantee the temporal consistency of pseudo-fake videos since the geometry of landmarks can significantly vary over time. To mitigate the issue, we

propose to re-interpolate each landmark $\mathbf{l}_i(t)$ based on $\mathbf{l}_i(t-1)$ for $t > t_0$ as follows,

$$\mathbf{l}_i(t) = \begin{cases} \mathbf{l}_i(t-1) + \dfrac{\mathbf{l}_i(t) - \mathbf{l}_i(t-1)}{\text{round}(d/\bar{d})}, \text{if } d > \tau \\ \quad \text{where } d = \ \|\mathbf{l}_i(t) - \mathbf{l}_i(t-1)\|_2/n \\ \mathbf{l}_i(t), \text{otherwise,} \end{cases} \tag{5.1}$$

where d represents the normalized distance between the position of a given landmark at the instants t and $t-1$, τ is a constant threshold for determining when to interpolate (interpolation intervenes only in the presence of drastic changes), and $\bar{d}$ is empirically chosen. Overall, a higher d value will push the updated point to move closer to the previous landmark position $\mathbf{l}_i(t-1)$, hence contributing to smooth the landmark position over time. However, excessive smoothing can be disadvantageous, as it can discard temporal artifacts. To address this, we use a round operator to incorporate slight errors. Hence, the proposed SBV data synthesis produces high-quality pseudo-fake videos incorporating subtle temporal artifacts.

As a result, we obtain the pseudo-fake video $\mathbf{X}_i^{\tilde{r}} \in \mathcal{V}^{\tilde{r}}$ and its blending mask sequence $\mathbf{M}_i \in \mathbb{R}^{T\times H\times W}$, with H and W being the image height and width, respectively. An illustration of SBV is given in Figure 5.2-II. Additional samples, as well as the detailed algorithm, are provided in supplementary materials. It is important to note that, despite being simple, SBV is generic and applicable to any existing video-level deepfake detection approach.

Vulnerability-Driven Cutout Augmentation. Previous works [123, 93] have demonstrated that deep learning methods are often impacted by overfitting. Deepfake detectors might even be more sensitive to this phenomenon as deepfakes are typically characterized by localized artifacts [62]. One solution to regularize training is data augmentation. As such, we propose, in addition to SBV, a novel Cutout data augmentation driven by vulnerable patches, i.e., image patches that are prone to blending artifacts [140]. We posit that by masking the most vulnerable regions, overfitting risks will be reduced, as the model will be pushed to learn from other areas. This masking strategy has already been explored in other computer vision fields such as Classification [144, 145], Object Detection [146] demonstrating great potential.

Specifically, similar to [25], we create a set of blending boundaries $\mathbf{B}$ using a randomly generated blending mask $\mathbf{M}$ as described in Eq. (2.2). Inspired by [140], vulnerability values are then quantified at the patch level in a non-overlapping manner by applying a MaxPooling function as follows,

$$\bar{\mathbf{B}} = \text{MaxPooling}(\mathbf{B}), \ \ \bar{\mathbf{B}} \in \mathbb{R}^{T\times\sqrt{N}\times\sqrt{N}}, \tag{5.2}$$

where N indicates the number of patches.

After that, we define a threshold τ_{cutout} that is randomly selected from the range $(0.5, 1.0]$. We use the latter to define the set of patches to be masked $\mathcal{P} = \left\{(l, m) \mid \bar{\mathbf{B}}_{l,m}(t_0) > \tau_{cutout}\right\}$ within the first frame. The set $\mathcal{P}$ is then used to mask out patches at those locations not only in the first frame but also in the entire video to enforce temporal consistency that is crucial for generating high-quality pseudo-fakes. After masking those patches over time, we finally obtain the masked blending boundary denoted as $\tilde{\mathbf{B}}$. This results in masking the most vulnerable regions, i.e., the regions that are the most likely to include blending artifacts. Figure 5.2-III shows some examples of the proposed cutout augmentation.

5.3.2 FakeSTormer

Our multi-task framework, called FakeSTormer, is inspired by [62, 140], where auxiliary branches are designed to push the feature extractor to focus on vulnerabilities. As discussed earlier, the vulnerability is defined in [62, 140] as the pixels/patches that are the most likely to be impacted by blending artifacts. This strategy is therefore claimed to allow the detection of subtle artifacts that are generic across different types of manipulations. While such a vulnerability-driven approach has shown very promising results [62, 140], it does not take into account the temporal nature of videos. Therefore, in addition to spatial vulnerabilities, we argue that there is a need to model temporal vulnerabilities, which we define as significant temporal changes in the blending boundary. Specifically, we introduce two additional branches, namely a temporal head h and a spatial one g. The branch h predicts the derivatives of the blending boundary over time which can reflect high changes, typically characterizing temporal artifacts. Moreover, we suggest the use of a spatial branch g which enables predicting soft labels representing the forgery intensity encoded in each frame, computed from vulnerability information. The proposed framework relies on the TimeSformer backbone [89], which we revisit for better modeling spatial and temporal information.

Herein, we first describe the proposed revisited TimeSformer-based feature extractor Φ in Section 5.3.2. We then detail the two additional temporal and spatial heads in Section 5.3.2 and Section 5.3.2, respectively. Finally, we give the overall training details in Section 5.3.2.

Backbone: Revisited TimeSformer

We choose TimeSformer [89] as our feature extractor given its ability to effectively capture separate long-range temporal information and spatial features. In TimeSformer, a video input $\mathbf{X} \in \mathbb{R}^{C \times T \times H \times W}$ results in an embedding matrix input $\mathbf{Z}^0 \in \mathbb{R}^{T \times N \times D}$. A global class token $\mathbf{z}_{cls}$ attends all patches and is then used for classification. This mechanism implicitly captures mixed spatio-temporal features, which might lead to overfitting one type of artifact. We

revisit it slightly in order to decouple the spatial and temporal information by considering two sorts of additional tokens (one spatial and one temporal).

For that purpose, we attach in each dimension of $\mathbf{Z}^0$, a spatial token $\mathbf{z}_s^0 \in \mathbb{R}^D$ and a temporal token $\mathbf{z}_t^0 \in \mathbb{R}^D$, respectively. These tokens will independently interact only with patch embeddings belonging to their dimension axis by leveraging the decomposed SA [89]. This mechanism not only facilitates the disentanglement learning process of spatio-temporal features but is also beneficial to optimize the computational complexity of $\mathcal{O}(T^2 + N^2)$ as compared to $\mathcal{O}(T^2 \cdot N^2)$ in vanilla SA. Those tokens will be then fed into L ($L = 12$ as default) transformer encoder blocks, as described in Figure 5.2-I. Formally, the feature extraction process can be summarized as follows,

$$[\mathbf{Z}^L, \mathbf{z}_s^L, \mathbf{z}_t^L] = \Phi(\mathbf{X}), \tag{5.3}$$

where $\mathbf{Z}^L$ is the final patch embedding matrix, $\mathbf{z}_s^L$ the resulting set of spatial tokens, and $\mathbf{z}_t^L$ the resulting set of temporal tokens that will be respectively sent to the temporal head h, the spatial head g, and the classification head f. More details about the implementation of the proposed revisited TimeSformer are given in supplementary materials.

Temporal Head h

Ground Truths. Our temporal head h aims to model fine-grained temporal vulnerabilities in deepfake videos through a regression task. First, to generate ground truth data for the branch h, we hypothesize that temporal high-changes in the blending boundary can reflect the presence of temporal artifacts (see Figure 5.2-IV). To achieve this, we compute $\mathbf{D}$ based on $\tilde{\mathbf{B}}$ such that:

$$\mathbf{D} = \frac{\partial \tilde{\mathbf{B}}}{\partial t}, \;\; \mathbf{D} \in \mathbb{R}^{T \times \sqrt{N} \times \sqrt{N}}. \tag{5.4}$$

More details regarding the derivative calculation are provided in supplementary materials. To stabilize training, $\mathbf{D}$ is standardized resulting in $\hat{\mathbf{D}} \in \mathbb{R}^{T \times \sqrt{N} \times \sqrt{N}}$. Experiments with different normalization strategies are reported in supplementary materials.

Architecture Design. In order to construct the regression head for predicting $\hat{\mathbf{D}}$, we take the patch embedding matrix $\mathbf{Z}^L$ as input and process them to produce 3D features as follows,

$$\mathbf{F} = \text{Reshape}(\mathbf{Z}^L), \mathbf{F} \in \mathbb{R}^{D \times T \times \sqrt{N} \times \sqrt{N}}. \tag{5.5}$$

To estimate temporal derivatives, we employ two 3D convolution blocks (3DCnvB) with

3-dimensional temporal kernels and 1-dimensional spatial kernels [52] as follows,

$$\tilde{\mathbf{D}} = h(\mathbf{F}) = \text{3DCnvB}_{3\times1\times1}(\text{3DCnvB}_{3\times1\times1}(\mathbf{F})), \tag{5.6}$$

where $\tilde{\mathbf{D}} \in \mathbb{R}^{T\times\sqrt{N}\times\sqrt{N}}$. Each convolution block comprises a 3D convolution layer, followed by a BatchNorm and a GELU layer.

Objective Function. For training the temporal branch, we optimize the following Mean Squared Error (MSE) loss,

$$\mathcal{L}_h = \frac{1}{T\times N}\|\hat{\mathbf{D}} - \tilde{\mathbf{D}}\|_2^2, \tag{5.7}$$

with $\|.\|_2$ referring to the L_2 norm.

Spatial Head g

Ground Truths. To avoid overfitting one type of artifact, we enforce the model to explicitly predict soft labels representing the intensity level of spatial artifacts for each video frame. Note that several works [98, 69, 147, 148, 124] have leveraged soft labels for training regularization. Given a pseudo-fake video $\mathbf{X} = (\mathbf{X}(t))_{t\in[[1,T]]}$ formed by T frames and $\tilde{\mathbf{B}} = (\tilde{\mathbf{B}}(t))_{t\in[[1,T]]}$ its associated cutout blending boundary, the ground truth for these soft labels is generated for each frame t as follows,

$$p(t) = \max_{l,m\in[[1,\sqrt{N}]]}(\tilde{\mathbf{B}}(t)), \tag{5.8}$$

resulting in the ground truth for training the spatial branch denoted as $\mathbf{p} = (p(t))_{t\in[[1,T]]}$. We note that $\mathbf{p} = \mathbf{1}^T$ if cutout is not applied and $\mathbf{p} = \mathbf{0}^T$ for a real video.

Architecture Design. To predict the proposed soft labels, a Multi-Layer Perceptron (MLP) is applied to the set of spatial tokens $\mathbf{z}_s^L$, as follows,

$$\tilde{\mathbf{p}} = g(\mathbf{z}_s^L) = \text{MLP}(\mathbf{z}_s^L), \;\; \tilde{\mathbf{p}} \in \mathbb{R}^T. \tag{5.9}$$

Objective Function. To train the spatial branch, we optimize the following Binary Cross Entropy (BCE) loss similar to [148, 69],

$$\mathcal{L}_g = \text{BCE}(\tilde{\mathbf{p}}, \mathbf{p}). \tag{5.10}$$

Overall Training Objective

Finally, for the standard classification head f, we use the set of temporal tokens $\mathbf{z}_t^L$ such that the predicted label $\tilde{y}$ is given by,

$$\tilde{y} = f(\mathbf{z}_t^L) = \text{MLP}(\mathbf{z}_t^L). \tag{5.11}$$

The classification loss $\mathcal{L}_c$ is then given by applying a BCE between the ground-truth label y and the predicted label $\tilde{y}$.

Overall, the network is trained by optimizing the following loss:

$$\mathcal{L} = \lambda_c \mathcal{L}_c + \lambda_h \mathcal{L}_h + \lambda_g \mathcal{L}_g, \tag{5.12}$$

where $\lambda_c, \lambda_h, \lambda_g$ are hyper-parameters to balance the training of the three branches.

Method	Training		Test set AUC (%)					
	Real	Fake	CDF	DFD	DFDCP	DFDC	DFW	DiffSwap
Xception [23]	✓	✓	73.7	-	-	70.9	-	-
MATT [39]	✓	✓	68.3	92.9	63.0	-	65.7	-
RECCE [32]	✓	✓	70.9	<u>98.2</u>	-	-	68.2	-
SBI [26]	✓	×	90.6	-	-	72.4	-	-
SFDG [48]	✓	✓	75.8	88.0	73.6	-	<u>69.3</u>	-
LSDA [139]	✓	✓	<u>91.1</u>	-	77.0	-	-	-
STIL [50]	✓	✓	75.6	-	-	-	-	-
LipForensics [54]	✓	✓	82.4	-	-	<u>73.5</u>	-	-
RealForensics [149]	✓	✓	86.9	82.2	75.9	-	-	-
FTCN [52]	✓	✓	86.9	94.4	74.0	71.0	-	-
ISTVT [53]	✓	✓	84.1	-	74.2	-	-	-
AltFreezing [56]	✓	✓	89.5	**98.5**	-	-	-	-
Swin+TALL [36]	✓	✓	90.8	-	76.8	-	-	-
StyleLatentFlows [143]	✓	✓	89.0	96.1	-	-	-	-
LFGDIN [59]	✓	✓	90.4	-	<u>80.8</u>	-	-	<u>85.7</u>
FakeSTormer ($T = 4$)	✓	×	**92.4**	**98.5**	**90.0**	**74.6**	**74.2**	**96.9**
FakeSTormer ($T = 8$)	✓	×	92.4	98.2	90.0	74.9	75.9	97.1
FakeSTormer ($T = 16$)	✓	×	92.8	98.6	90.2	75.1	75.3	97.2

Table 5.1 Generalization to unseen datasets. AUC (%) comparisons at *video-level* on multiple unseen datasets [80, 83, 81, 88, 82, 13]. All detectors are trained on FF++(c23). Results are directly extracted from the original papers and from [62, 54]. **Bold** and <u>Underlined</u> text, respectively highlight the best and the second best performance, excluding the variants of our framework with $T = 8$ and $T = 16$.

Method	Training set		Cross-dataset			DF40 subset		
	Real	Fake	CDF	DFDCP	DiffSwap	BlendFace	FSGAN	MobileSwap
Face X-ray [25]	✓	✓	79.5	-	-	-	-	-
PCL+I2G [33]	✓	✓	90.0	74.3	-	-	-	-
SLADD [31]	✓	✓	79.7	-	-	-	-	-
SBI [26]	✓	×	93.2	86.2	90.6	86.5	85.4	86.6
LAA-Net [62]	✓	×	95.4	86.9	92.1	**91.2**	94.2	93.9
STC-Scratch [58]	✓	×	83.4	86.8	-	-	-	-
STC-Pretrain [58]	✓	×	<u>95.8</u>	89.4	-	-	-	-
ST-SBV [57]	✓	×	90.3	91.2	-	-	-	-
StA+VB [29]	✓	×	94.7	90.9	-	90.6	**96.4**	<u>94.6</u>
TimeSformer [89] + SBV	✓	×	94.9	<u>93.0</u>	<u>93.3</u>	89.7	<u>94.6</u>	<u>94.6</u>
FakeSTormer	✓	×	**96.5**	**94.1**	**97.7**	<u>91.1</u>	**96.4**	**95.0**

Table 5.2 AUC(%) comparison at video-level with other data synthesis methods. For fair comparison, we train our FakeSTormer on raw data of FF++(c0), and test *cross-dataset* on [80, 81, 13] and *cross-manipulation* on three subsets of [14].

5.4 Experiment

5.4.1 Settings

Datasets. We set up our datasets following several works [56, 52, 36, 53, 143, 44, 48, 32]. For both training and validation, we employ **FaceForensics++** (FF++) [23], which consists of four manipulation methods for the fake data (Deepfakes (DF) [116], FaceSwap (FS) [118], Face2Face (F2F) [117], and NeuralTextures (NT) [119]). It can be noted that, for training, we use only the real videos and generate pseudo-fake data using our synthesized method, SBV. *By default, the c23 version of FF++ is adopted*, following the recent literature [52, 56, 53, 36, 143]. For further validation, we also evaluate on the following datasets: **Celeb-DFv2** (CDF) [80], **DeepfakeDetection** (DFD) [83], **Deepfake Detection Challenge Preview** (DFDCP) [81], **Deepfake Detection Challenge** (DFDC) [88], **WildDeepfake** (DFW) [82], **DF40** [14], and **DiffSwap** [59, 13] generated using a recent diffusion-based approach [13]. Further details on these datasets are provided in supplementary materials.

Data Pre-processing. Following the splitting convention [23], we extract 256, 32, and 32 consecutive frames for training, validation, and testing, respectively. Facial regions are cropped using Face-RetinaNet [137] and resized to a fixed resolution of 224×224. Additionally, we store 81 facial landmarks for each frame, extracted using Dlib [120]. Further details are provided in the supplementary materials.

Evaluation Metrics. For fair comparisons with SOTA methods, we use the widely adopted Area Under the Curve (AUC) metric at the video level [56, 143, 52, 54, 149, 36, 53].

Implementation Details. Our framework is initialized with pretrained MAE weights [72] and trained for 100 epochs using the SAM optimizer [122] with a weight decay of 10^{-4} and a batch size of 32. The learning rate starts at 5×10^{-4} for the first quarter of training and

Method	Training set		FF++ LQ (%)	
	Real	NT	DF	FS
Xception [23]	✓	✓	58.7	51.7
Face X-ray [25]	✓	✓	57.1	51.0
F3Net [24]	✓	✓	58.3	51.9
RFM [93]	✓	✓	55.8	51.6
SRM [150]	✓	✓	55.5	52.9
SLADD [31]	✓	✓	62.8	56.8
TALL-Swin [36]	✓	✓	63.2	51.4
ResNet3D* [151]	✓	✓	66.8	<u>60.6</u>
TimeSformer* [89]	✓	✓	<u>73.3</u>	54.4
Ours	✓	×	**85.3**	**62.1**

Table 5.3 Generalization on heavily compressed data (LQ). AUC (%) comparisons on FF++ (LQ) [23] with a high compression level (c40). The results for comparison are directly extracted from [31, 152]. The symbol $*$ denotes our implementation.

decays to 0 thereafter. The backbone is frozen for the first 5 epochs for warm-up, then all layers are unfrozen. Data augmentation includes ColorJittering at the video level and our proposed Cutout. Experiments are conducted on four NVIDIA A100 GPUs, with $\tau = 0.35$ and $\bar{d} = 0.2$ (Eq. (5.1)), and $T = 4$ frames in most experiments.

5.4.2 Comparison with State-of-the-art Methods

Generalization to Unseen Datasets. To assess the generalization capabilities of our method, we conduct evaluations using the challenging *cross-dataset* setup [56, 52, 62, 48, 32], validating on unseen datasets (i.e., datasets other than FF++). The results are detailed in Table 5.1 and Table 5.2.

As shown, our method achieves comparable results on DFD while surpassing SOTA methods on other datasets. Specifically, it significantly outperforms prior video deepfake detection techniques, including spatio-temporal learning-based methods like AltFreezing [56] and ISTVT [53], as well as various data synthesis approaches. Moreover, our method exhibits superior performance on the large-scale DFDC dataset and the challenging in-the-wild DFW dataset. These results further confirm the enhanced generalization ability of FakeSTormer compared to recent methods.

Generalization on Heavily Compressed Data. Following previous work [152, 31], we also evaluate FakeSTormer on heavily compressed FF++(c40) data. In addition to comparing with several SOTA methods, we train ResNet3D [151], commonly used in deepfake video detection [52, 56, 143], and TimeSformer [89] on NT, then test on DF and FS. The comparison results are presented in Table 5.3. Our method achieves notably higher AUC scores than other

Method	Training set		FF++ (%)				
	Real	Fake	DF	FS	F2F	NT	Avg.
Xception [23]	✓	✓	93.9	51.2	86.8	79.7	77.9
Face X-ray [25]	✓	✓	99.5	93.2	94.5	92.5	94.9
SBI [26]	✓	×	98.6	95.4	92.6	82.3	92.2
LSDA [139]	✓	✓	96.9	95.1	96.4	94.9	95.8
LipForensics [54]	✓	✓	99.7	90.1	99.7	99.1	97.1
FTCN [52]	✓	✓	99.8	99.6	98.2	95.6	98.3
RealForensics [149]	✓	✓	**100**	97.1	99.7	99.2	99.0
AltFreezing [56]	✓	✓	99.8	99.7	98.6	96.2	98.6
StyleLatentFlows [143]	✓	✓	99.7	98.8	98.6	96.4	98.4
NACO [153]	✓	✓	99.9	99.7	99.8	99.4	99.7
LFGDIN [59]	✓	✓	96.2	80.5	90.5	81.7	87.2
Ours (c23)	✓	×	99.9	97.8	98.5	97.2	98.4
Ours (c0)	✓	×	**100**	**99.8**	**99.9**	**99.7**	**99.9**

Table 5.4 Generalization to unseen manipulations. AUC (%) comparisons on FF++ [23], which consists of four manipulation methods (DF, FS, F2F, NT).

SBV	V-CutOut	g	h	Test set AUC (%)						Avg.
				CDF	DFD	DFDCP	DFDC	DFW	DiffSwap	
×	×	×	×	61.5	62.8	59.4	58.5	65.2	71.6	63.2
✓	×	×	×	90.7	95.7	87.9	72.2	70.9	92.9	85.1(↑21.9)
✓	✓	×	×	91.1	96.0	87.6	72.6	71.0	93.1	85.2(↑22.0)
✓	✓	✓	×	92.2	95.4	88.5	72.8	71.3	93.8	85.7(↑22.5)
✓	×	×	✓	**93.4**	**98.5**	88.5	72.8	69.6	**97.3**	86.7(↑23.5)
✓	✓	✓	✓	92.4	**98.5**	**90.0**	**74.6**	**74.2**	96.9	**87.8**(↑24.6)

Table 5.5 Ablation study of framework's components. Gray indicates the use of original fake data for training.

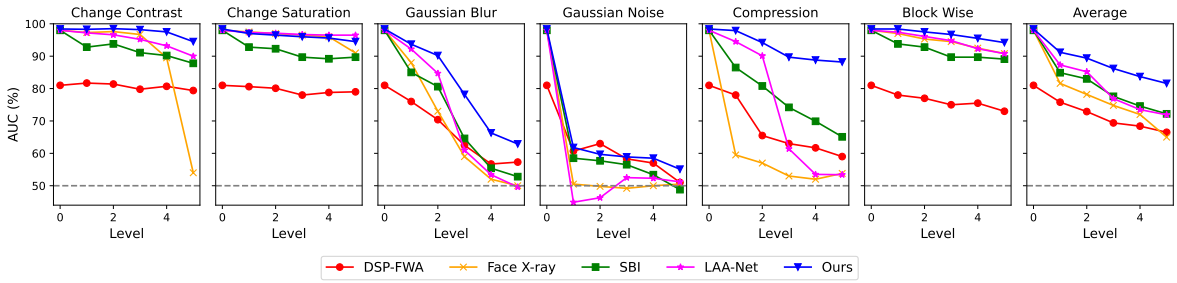


Figure 5.3 Robustness to unseen perturbations. AUC (%) under five different degradation levels for various types of perturbations [60] on FF++ [23]. "Average" denotes the mean across all corruptions at each level. Best viewed in color.

methods across both testing subsets, highlighting its robust generalization capability under various data compression conditions.

Generalization to Unseen Manipulations. Table 5.4 compares our framework with SOTA methods on FF++. Other methods [54, 56, 143, 52] use a cross-manipulation setup, training on three forgery types and evaluating on the remaining one. In contrast, our approach trains only on real videos, treating all manipulations as unseen. Despite this, our method shows competitive performance with the others, even without being trained on specific forgery types.

Robustness to Unseen Perturbations. Deepfakes are widely shared on social media, where various perturbations can affect their appearance. Following [60], we evaluate FakeSTormer's robustness across six unseen degradation types at five levels, comparing it with other augmented-based methods [43, 25, 26, 62]. Figure 5.3 shows AUC scores for each method on these perturbations, using models trained on FF++. Our results demonstrate that FakeSTormer outperforms prior methods on most distortions, with a slight drop compared to LAA-Net [62] for Change Saturation. Nonetheless, FakeSTormer achieves higher performance on average, especially at higher severity levels, highlighting its superior generalization and robustness. Detailed scores are in supplementary materials.

5.4.3 Additional Discussions

Ablation Study of the FakeSTormer's Components. We conduct ablation studies to assess the impact of each component in our framework, as shown in Table 5.5. Using TimeSformer trained on FF++ as the baseline, we experiment with different combinations of components: Self-Blended Video (SBV), Vulnerability-driven CutOut (V-CutOut), the spatial branch (g), and the temporal branch (h). Each component improves performance, with SBV providing the most significant boost by generating high-quality pseudo-fake data that aids generalization to unseen datasets.

Method	Comp.	CSP	LI	Test set AUC (%)					
				CDF	DFD	DFDCP	DFDC	DFW	DiffSwap
Stacked SBIs	c23	×	×	48.4	49.1	48.9	51.7	52.8	55.5
+ CSP	c23	✓	×	84.1	89.2	86.1	69.5	65.5	85.7
SBV	c23	✓	✓	<u>90.7</u>	<u>95.7</u>	<u>87.9</u>	<u>72.2</u>	<u>70.9</u>	<u>92.9</u>
SBV	c0	✓	✓	**94.9**	**97.6**	**93.0**	**76.4**	**75.3**	**93.3**

Table 5.6 Ablation study of SBV's components. Performance analyses of different SBV's components using cross-evaluation on multiple datasets [80, 83, 81, 88, 82, 13].

λ_c	λ_h	λ_g	Test set AUC (%)						Avg.
			CDF	DFD	DFDCP	DFDC	DFW	DiffSwap	
0.9	1	0.1	89.5	91.0	**93.2**	71.5	**74.7**	90.6	85.1
0.9	10	0.1	**92.5**	95.7	86.8	71.7	72.7	94.5	85.7
0.9	100	0.1	91.6	<u>98.0</u>	87.3	73.6	70.6	96.2	86.2
0.8	100	0.2	<u>92.4</u>	**98.5**	<u>90.0</u>	**74.6**	<u>74.2</u>	<u>96.9</u>	**87.8**
0.5	100	0.5	<u>92.4</u>	<u>98.0</u>	88.1	<u>74.5</u>	72.1	**97.1**	<u>87.0</u>

Table 5.7 Impact of loss balancing factors. AUC (%) comparisons of FakeSTormer trained with different values of λ_c, λ_h, and λ_g on cross-dataset setup, demonstrating robustness to varying hyperparameter settings.

Ablation Study of the SBV's Components. SBV enhances SBI [26] with CSP and LI for robust pseudo-fake generation in video data. Table 5.6 shows that without these components, simply stacking frame-wise SBIs fails to produce consistent temporal features, leading to overfitting on more obvious artifacts and poor generalization [56]. A qualitative comparison is provided in supplementary materials.

Influence of Number of Frames. Increasing the number of frames T provides more fine-grained temporal information. In Table 5.1, we vary T values by fixing it to 4, 8, and 16. Our results show a consistent performance improvement with more frames, confirming our hypothesis. However, increasing T also incurs a higher computational cost.

Impact of Loss Balancing Factors. We introduce three hyperparameters, λ_c, λ_g, and λ_h in Eq. (5.12) to balance the training among the three branches of our framework. In Table 5.7, we analyze the impact of these hyperparameters using various values. Our results show that the method is robust to a range of hyperparameter values, with the best performance achieved when λ_c, λ_g, and λ_h are set to 0.8, 0.2, and 100, respectively.

5.4.4 Visualization of Saliency Maps

To analyze the contribution of the two proposed branches h and g in the detection performance of FakeSTormer, we visualize the input regions activated by those branches.

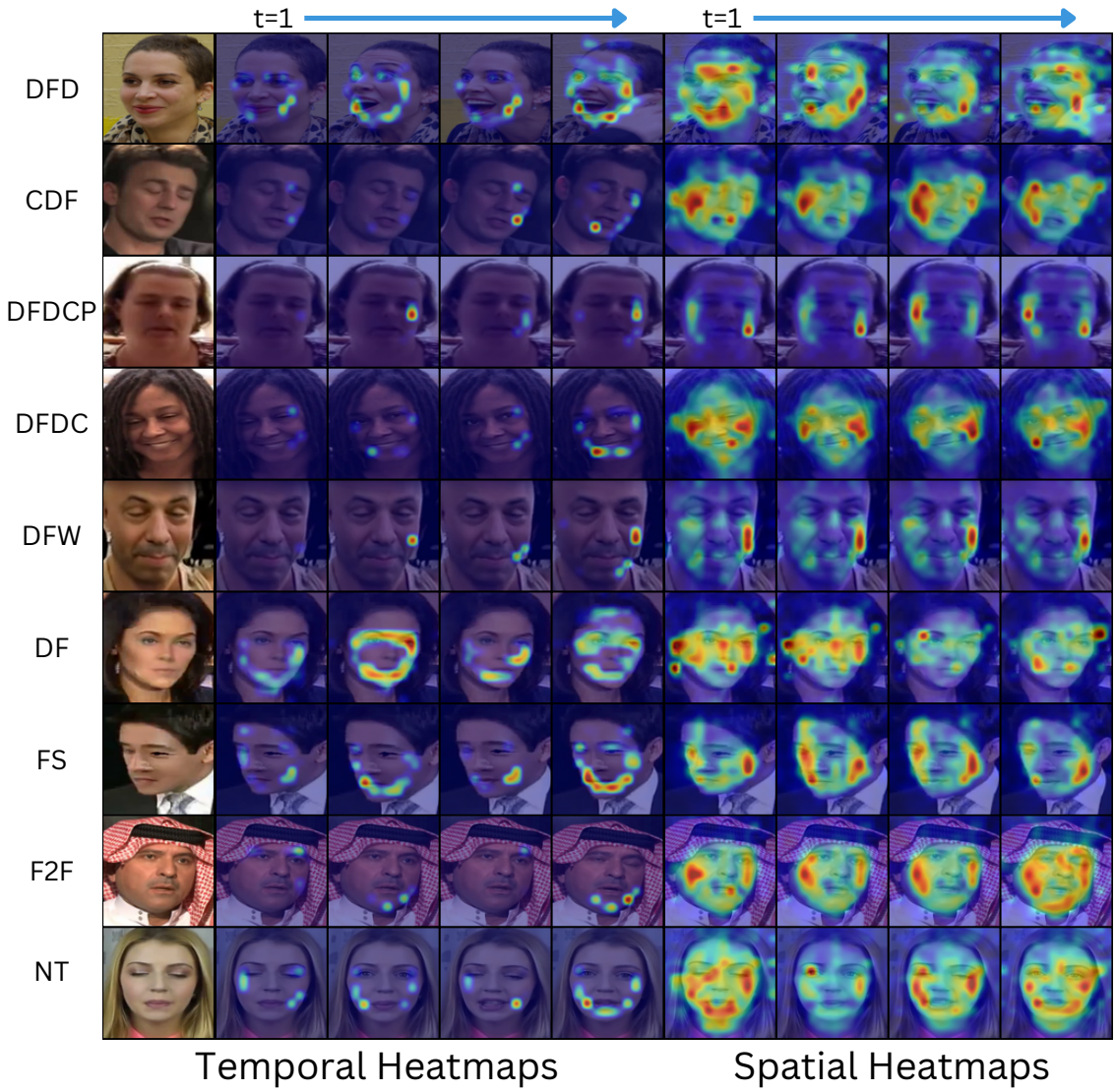


Figure 5.4 Visualization of Saliency Maps. The second-fifth and sixth-ninth columns represent temporal heatmaps and spatial heatmaps on different frames in the video, respectively. All datasets are unseen during validation.

For that purpose, we adopt Grad-CAM [127] for the temporal branch h and utilize the final SA scores of spatial tokens for the spatial branch g. The visualization results from various datasets are presented in Figure 5.4. It can be observed that FakeSTormer can discriminate between real and fake videos by focusing on very few, different local areas, even without having seen those types of forgeries during training.

5.5 Conclusion

This chapter introduces a fine-grained approach for generalizable deepfake video detection with two main contributions. First, we propose a multi-task learning framework that targets both subtle spatial and fine-grained temporal vulnerabilities in high-fidelity deepfake

videos, incorporating a standard classification branch along with two new auxiliary branches (temporal and spatial). These proposed branches help the model focus on vulnerable regions and provide more valuable insights into how the network *sees* the data while offering more robustness to high-quality deepfakes. This framework is further supported by the introduction of a high-quality pseudo-fake generation technique. Extensive experiments on several challenging benchmarks demonstrate that FakeSTormer achieves superior performance compared to SOTA methods.

Chapter 6

AUC Meets Polarization: Toward a More Realistic Evaluation of Deepfake Detectors

This chapter shifts the focus from model design to evaluation and introduces the final contribution of the thesis: Cross-AUC, a more realistic metric for assessing deepfake detectors under domain shift. While AUC is often used and averaged across different deepfake datasets, as discussed in chapter 2, this measure can hide the limitations of existing methods when dealing with heterogeneous datasets. As such, this chapter discusses the limitations of AUC in the presence of distributional shift, particularly in the context of deepfake detection. Then, it proposes Cross-AUC, a metric that jointly accounts for ranking performance and prediction-score polarization across domains.

6.1 Introduction

Recent advances in generative AI, such as diffusion models and open-source face-swapping tools, have facilitated the creation of ultra-realistic forged facial images. Such manipulated data, also known as deepfakes, have already caused real harm, from financial scams [19] to the diffusion of non-consensual explicit content [20].

In response to this threat, the field of deepfake detection has become a very active research topic [23, 37, 32, 31, 26, 25, 65, 62, 140, 90, 27, 39, 28, 154]. Earlier methods mostly employ binary classifiers using standard deep architectures (e.g., XceptionNet [46], EfficientNet [45]) that learn to distinguish between real and fake data. Nevertheless, these approaches tend to

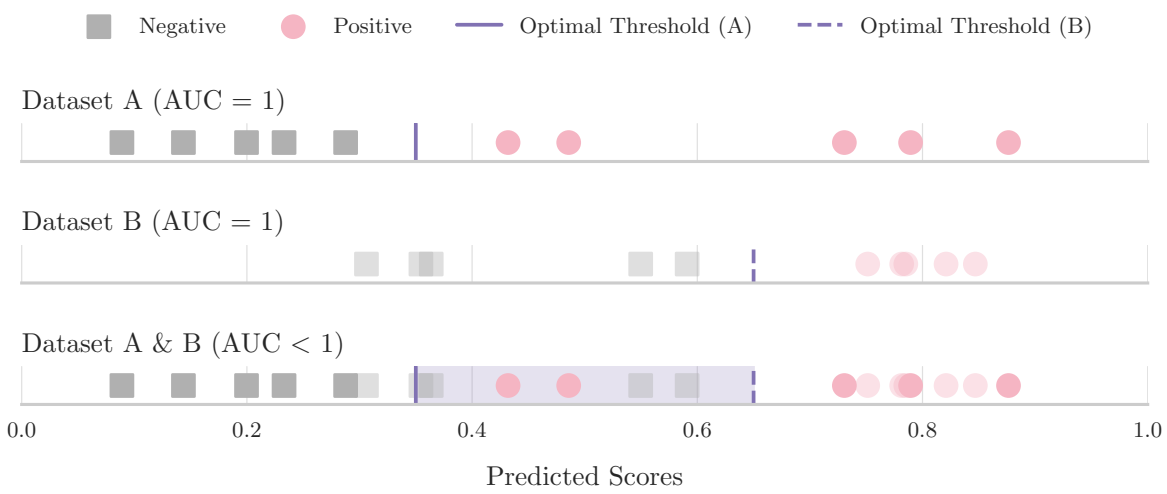


Figure 6.1 Example of optimal thresholds for predictions on different datasets, A (top) and B (middle). While a perfect AUC $= 1$ is reached on both datasets separately, an obvious drop in AUC will appear when combining them to simulate real-world complexities (bottom), since the scores are no longer well ordered. The anisotropic nature of score distributions leads to an irreconcilable disparity between optimal thresholds (purple range), demonstrating the importance of taking into account the polarization of probability predictions in real-world scenarios.

overfit specific artifact patterns introduced by the generation techniques used to create the fake training samples; thereby, achieving poor generalization capabilities when considering unseen generation techniques. Solving this issue is crucial because new generative methods are constantly emerging, introducing distinct and inherently different artifact traces. To overcome this problem, recent state-of-the-art methods have adopted multi-task learning [33, 31, 32, 62, 27] and/or data synthesis strategies [26, 31, 25, 64, 29], claiming an increased robustness to unseen manipulations. To demonstrate the generalization capabilities of these methods, a cross-dataset evaluation protocol is typically followed, where different datasets are employed during training and testing phases [26, 31, 25, 62]. Given its robustness to imbalanced data, the Area Under the Receiver Operating Characteristic (ROC) Curve (AUC) is then reported on each testing dataset separately. However, such an evaluation protocol cannot fully reflect the relevance of a given deepfake detector in a real-world setting. In practice, an effective detector should be able to deal with uncontrolled deepfake data featuring different kinds of artifacts at the same time, regardless of their origin. As such, it is likely that the encountered data resembles a mixture of multiple datasets. Therefore, evaluating AUC separately on each dataset, as commonly done, fails to capture this realistic setting. The AUC measure, which estimates the class separability of a binary classification model, can be high when evaluated on each dataset individually. However, it can drop drastically when considering two datasets incorporating a domain shift, as illustrated in Figure 6.1.

Therefore, we propose a new metric, called *Cross-dataset AUC (Cross-AUC)*, that can better

assess the generalization capabilities of deep detectors under domain shift. It relies on the mean AUC on diverse datasets while taking into account the polarization extent of the probability predictions. A stronger polarization of probability predictions (predictions closer to 0 and 1) suggests that the classes are more distinctly separated within the score space, and a fixed threshold is more likely to remain effective under varied conditions, contributing to enhanced generalization capabilities. The polarization extent is estimated using the Wasserstein Distance (WD) between the distribution of positive and negative probability predictions. Consequently, the proposed Cross-AUC captures both the relative ranking performance (via the mean AUC) and the absolute separability of predictions (via the polarization), offering a more adequate estimate of generalization performance in real-world scenarios. Finally, we empirically show that the Cross-AUC is very close to the AUC measured on a combination of all the considered datasets, making it a practical measure for evaluating cross-dataset generalization.

In summary, this work presents the following contributions:

- We discuss and analyze the sensitivity of AUC under domain shift and its unsuitability for assessing the generalization capabilities of deepfake detectors.
- We introduce a new metric called *Cross-dataset AUC (Cross-AUC)* to evaluate the generalization performance of deepfake detectors, taking into account the mean AUC across different datasets, as well as the polarization of probability predictions.
- We investigate the impact of prediction polarization between classes and show that similar AUC values can correspond to vastly different decision behaviors. This is quantified using the WD between score distributions.
- We conduct an experimental evaluation and analysis based on the proposed Cross-AUC for comparing several deepfake detectors, including recent ones using seven well-known benchmarks [23, 80, 83, 82, 81, 88, 14].

6.2 Problem formulation: AUC under domain shift

Let us denote by $\mathbf{X} \in \mathbb{R}^{c \times h \times w}$ an input image and $y \in \{0, 1\}$ its associated label, which follow a joint probability distribution $p(\mathbf{X}, y)$. We assume that $y = 0$ and $y = 1$ if $\mathbf{X}$ belongs to the negative and positive classes (i.e., real and fake), respectively. As such, we denote by $p^p(\mathbf{X}) = p(\mathbf{X} \mid y = 1)$ and $p^n(\mathbf{X}) = p(\mathbf{X} \mid y = 0)$ the conditional probability density of the positive and negative classes, respectively. Furthermore, let $s : \mathbb{R}^{c \times h \times w} \mapsto [0, 1]$ be a scoring function (often computed using a deep neural network in the context of deepfake detection) that estimates the positive posterior probability of an input sample $p(y = 1 \mid \mathbf{X})$.

Given a fixed threshold τ, the classifier that gives the predicted label $\hat{y}$ is then defined as follows: $\hat{y} = \mathbb{I}(p(y = 1 \mid \mathbf{X}) > \tau)$, where $\mathbb{I}$ is the indicator function. The Area Under the ROC Curve (AUC) is a standard metric often used in deepfake detection and more generally in binary classification. Specifically, it reflects how well a model is able to separate positive from negative samples in the score space and is independent from the classification threshold τ. It measures the probability that a randomly chosen positive instance (e.g., a fake video) receives a higher confidence score than a randomly chosen negative one (e.g., a real video) [155, 156, 157, 158]. Formally, the AUC can be defined as,

$$\mathrm{AUC}(s) = \mathbb{E}_{\mathbf{x}^p \sim p^p(\mathbf{x}),\, \mathbf{x}^n \sim p^n(\mathbf{x})} \left[\mathbb{I}(s(\mathbf{x}^p) > s(\mathbf{x}^n)) \right], \tag{6.1}$$

This formulation emphasizes that AUC reflects the model's ability to rank positive instances higher than negative ones, depending only on the relative ordering of the scores.

Although AUC is effective from a dataset to another when the class-conditional distributions are stable, it becomes problematic in the presence of a domain shift. Let i and j denote two different datasets with their distribution of positive and negative prediction samples denoted as $p_i^p(\mathbf{X})$, $p_j^p(\mathbf{X})$ and $p_i^n(\mathbf{X})$, $p_j^n(\mathbf{X})$, respectively.

In the presence of a **domain overlap** between i and j, the class-conditional distributions are approximately aligned:

$$p_i^p(\mathbf{X}) \approx p_j^p(\mathbf{X}), \quad p_i^n(\mathbf{X}) \approx p_j^n(\mathbf{X}). \tag{6.2}$$

Hence, the model produces comparable confidence scores across domains. The indicator function in the AUC formula, $\mathbb{I}(s(\mathbf{x}^P) > s(\mathbf{x}^N))$, remains meaningful, and global AUC reliably reflects the model's performance.

Nevertheless, in the presence of a **domain gap** in the prediction probability space, the two distributions differ significantly,

$$p_i^p(\mathbf{X}) \not\approx p_j^p(\mathbf{X}), \quad p_i^n(\mathbf{X}) \not\approx p_j^n(\mathbf{X}). \tag{6.3}$$

In this context, the positive and the negative posterior probabilities $p^p(\mathbf{X})$ and $p^n(\mathbf{X})$ can be expressed as the mixture of domain-specific posterior probabilities as follows,

$$p^p(\mathbf{X}) = \sum_i \alpha_i \, p_i^p(\mathbf{X}) \not\approx p_i^p(\mathbf{X}), \ \forall i, \tag{6.4}$$

$$p^n(\mathbf{X}) = \sum_i \beta_i \, p_i^n(\mathbf{X}) \not\approx p_i^n(\mathbf{X}), \ \forall i, \tag{6.5}$$

where α_i and β_i represent the mixture weights. This is typically the case when considering different deepfake detection datasets such as FaceForensics++ [23], Celeb-DF [80], and DFDC [88]. This divergence can result from differences in deepfake generation methods, compression levels, facial attributes, resolution, or lighting conditions across datasets. This can lead to inconsistencies in how the model scores samples across domains as we can deduce from Eq.(6.1). For example, the model might assign higher confidence scores to fake videos from one dataset, but not on the other dataset. These inconsistencies can distort global rankings and degrade AUC, even when performance within each domain is strong. This makes AUC an unreliable performance measure if considered on different domains separately, as experimentally demonstrated in Section 6.4.2.

6.3 Cross-dataset AUC

We propose a revised evaluation protocol that more faithfully reflects the deployment challenges of deepfake detection. As discussed in Section 6.2, the conventional evaluation metric, AUC, is computed independently on individual datasets and provides an incomplete or overly optimistic view of model robustness. This practice overlooks two critical issues: (i) the instability of decision thresholds across datasets, and (ii) the implicit assumption of isotropic distribution shifts, that all datasets differ from each other in a uniform and symmetric way. In this section, we will quantify the cross-dataset instability and define our proposed metric, Cross-AUC.

6.3.1 Threshold Instability and Polarization

We first expose score instabilities empirically by locating the separation between the negative and positive probability predictions for a given method over multiple datasets. This separation is known as a threshold and is needed for any deployment scenario. Its value can depend on the application (*e.g.*, maximizing detection rate or reducing false alarm rate), but in a generic study, an ideal threshold should be the best balance between correctly classified positives and correctly classified negatives. We chose this optimal threshold τ that gives the closest score to perfect classification (*i.e.*, FPR $= 0$, TPR $= 1$). From the set of all possible thresholds T, it is therefore the value that minimizes the distance between this perfect score and the values of the ROC curve axes (FPR and TPR):

$$\tau = \arg\min_{t \in T} \left(\sqrt{(1 - \text{TPR}_t)^2 + \text{FPR}_t^2} \right). \tag{6.6}$$

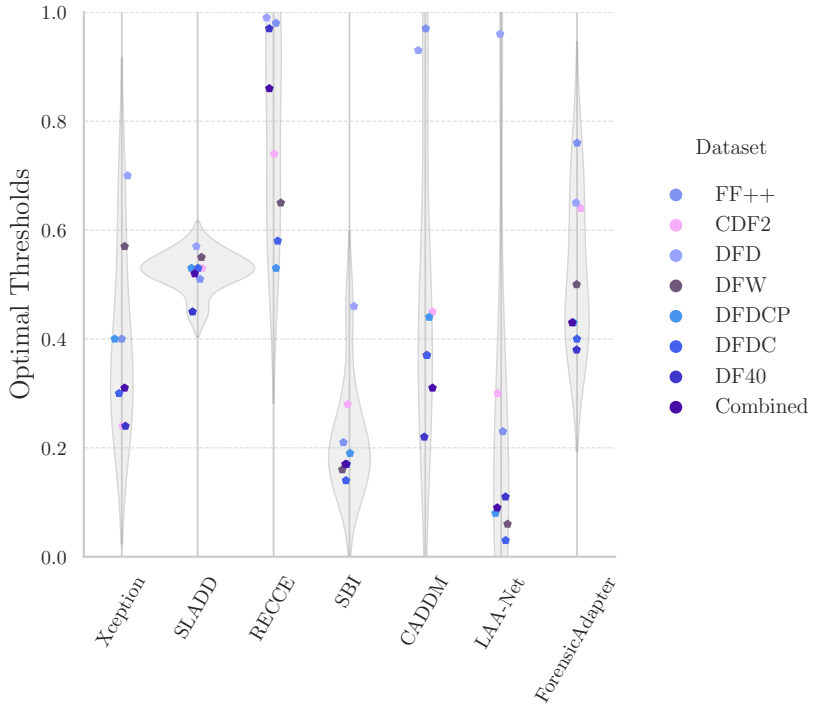


Figure 6.2 Optimal thresholds for the studied SOTA methods on the seven different datasets (and all of them combined).

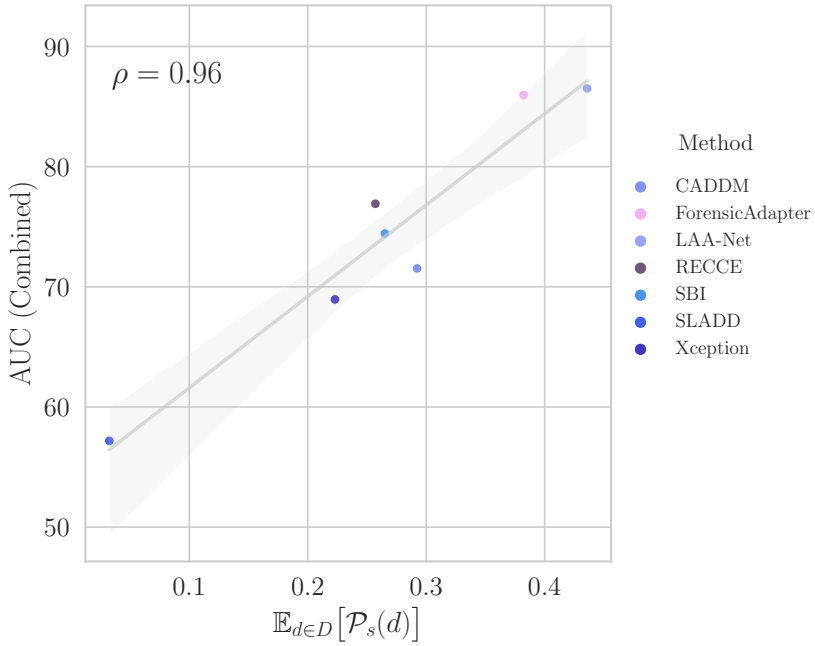


Figure 6.3 Correlation between estimated polarity (averaged across datasets) and the target combined AUC. Pearson coefficient $\rho = 0.96$.

Figure 6.2 shows the optimal threshold for each method and dataset. Our analysis reveals large threshold variations, with standard deviations reaching 0.16 for Xception [23], or even 0.31 for LAA-Net [62]. This large discrepancy between optimal thresholds will cause lower performance than expected after deployment.

However, the stability of the optimal threshold does not necessarily reflect the stability of the model if the probability predictions vary in a small range. On the other hand, a model robust to high variance thresholds is more likely to be stable across domain shifts. We propose to measure this robustness criterion via *polarization*. For the following, we consider a set of trained models $\mathcal{S}$ as our scoring functions, and a set of datasets $\mathcal{D}$. Herein, we introduce ***Polar Sets*** and ***Polarity*** as follows.

Definition (Polar Sets): Let $s \in \mathcal{S}$ be a model and $d \in \mathcal{D}$ a dataset. The polar sets associated to s and d are defined as $\mathcal{R}_{s,d} = \{s(\mathbf{X}) \mid y = 0, \forall \mathbf{X} \in d\}$ and $\mathcal{F}_{s,d} = \{s(\mathbf{X}) \mid y = 1, \forall \mathbf{X} \in d\}$. Thus, the sets $\mathcal{R}_{s,d}$ and $\mathcal{F}_{s,d}$ contain the probability predictions given by s of all real data and fake data, respectively.

Definition (Polarity): The polarity $P_s(d)$ of a model $s \in \mathcal{S}$ on a dataset $d \in \mathcal{D}$ is a divergence (or distance) between the estimated probability distributions $R_{s,d} = \Theta(\mathcal{R}_{s,d})$ and $F_{s,d} = \Theta(\mathcal{F}_{s,d})$, with Θ a given density estimation function.

An ideal model s^* would converge towards $R_{s,d} = \delta_0$ and $F_{s,d} = \delta_1$, which represent Dirac impulses at both extremities of the scoring space $[0, 1]$. By assuming that the distance between the distribution is calculated using the Wasserstein distance W, which we will use throughout the whole study, we can intuitively find that s^* has a polarity $P_{s*,d}$ of $W(\delta_0, \delta_1) = |0-1| = 1$, for any $d \in \mathcal{D}$. On the other hand, the worse performing model s^- would end up with $R_{s^-,d} \approx F_{s^-,d}$ and a polarity $P_{s^-,d} \approx 0$.

As we focus on the robustness of models approaching perfect convergence, we can safely assume both densities $R_{s,d}$ and $F_{s,d}$ to be close to the objective densities ($R_{s,d} \approx \delta_\alpha$, $F_{s,d} \approx \delta_{1-\beta}$, α and β representing small shifts). Upon this condition, we can derive the definition space of τ_d for each dataset, denoted as $[\alpha_d, 1 - \beta_d]$. It gives a tolerance of $1 - \max_{d \in D}(\beta_d) - \max_{d \in D}(\alpha_d)$ for the cross-dataset optimal threshold, τ. This suggests that the lower α and β are across numerous datasets, the greater the tolerance over different thresholds will be. This directly translates to having a maximum polarity across datasets. The hypothesis is empirically verified and illustrated in Figure 6.3. Instead of reporting the AUC on each dataset separately, we construct a **Combined dataset** mixing all the datasets, on which we report the AUC. Then, the AUC results on the Combined dataset are reported according to the empirical average of polarities, $\mathbb{E}_{d \in D}\Big[P_s(d)\Big]$. We can note a high correlation between the empirical average of polarities, $\mathbb{E}_{d \in D}\Big[P_s(d)\Big]$ and the AUC.

6.3.2 Leveraging polarization and AUC for cross-dataset evaluation

To better evaluate the reliability of a model and take into account conflicting optimal thresholds, we propose to use the defined polarity as a reward parameter. We further establish that a stable AUC and polarities throughout the datasets will ensure better reliability. To this end, we define a penalty-reward criterion C, such that we have i) a penalty when AUCs are unstable and ii) a reward for high polarities (reduced by their instability). For a model $s \in \mathcal{S}$, $\mathcal{A} = \{\text{AUC}(s, d) \mid d \in \mathcal{D}\}$ its AUC scores across the datasets, and $\mathcal{P} = \{P_s(d) \mid d \in \mathcal{D}\}$ its polarities, this criterion is defined as:

$$C(\mathcal{A}, \mathcal{P}) = -\Phi(\mathcal{A}) + |\Psi(\mathcal{P}) - \Phi(\mathcal{P})|\,, \tag{6.7}$$

with $\Psi : \mathbb{R}_+^k \mapsto \mathbb{R}_+$ and $\Phi : \mathbb{R}_+^k \mapsto \mathbb{R}_+$ measure the central tendency and dispersion, respectively, and $k = \text{card}(\mathcal{D})$. The proposed Cross-AUC is then defined as an adjusted averaged AUC:

$$\text{Cross-AUC} = \Psi(\mathcal{A}) + \lambda \cdot C(\mathcal{A}, \mathcal{P}), \tag{6.8}$$

with λ a balancing parameter.

For our experiments, we found the combination of the harmonic mean for Ψ, the standard deviation for Φ and $\lambda = 0.5$ to provide the best fit. We vary Θ, the density estimation function, and provide an ablation on Ψ. We show experimentally in Section 6.4 that the proposed Cross-AUC is a good approximation of the AUC on the Combined dataset.

6.4 Experiments

6.4.1 Experimental Settings

Deepfake Detectors. To conduct our evaluation study, seven SOTA methods [23, 31, 32, 26, 27, 62, 28] are selected. The selection is based on the following criteria: (1) the official codes of the considered methods are available and open-source for reliable reproducibility; (2) they are diverse in the way they approach deepfake detection. To guarantee this diversity, we include in our study an end-to-end binary classifier (**Xception** [23]), a disentanglement learning approach (**RECCE** [32]), a method relying on data synthesis (**SBI** [26]), hybrid methods combining multi-task learning and data synthesis strategies (**SLADD** [31], **CADDM** [27]), a fine-grained approach (**LAA-Net** [62]), and a method leveraging CLIP [74] (**ForensicAdapter** [28]); (3) except [23] that was published in 2019, the considered baselines are very recent and have been accepted in top-tier conferences. The summary of compared

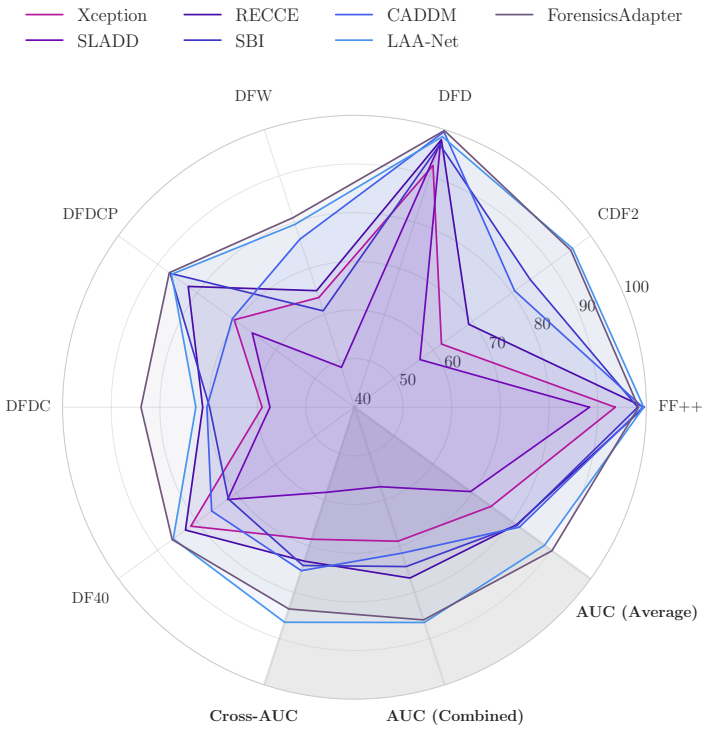


Figure 6.4 Methods results (AUC) for each dataset. Averaged AUC, AUC on the combined results, and the proposed Cross-AUC are also reported.

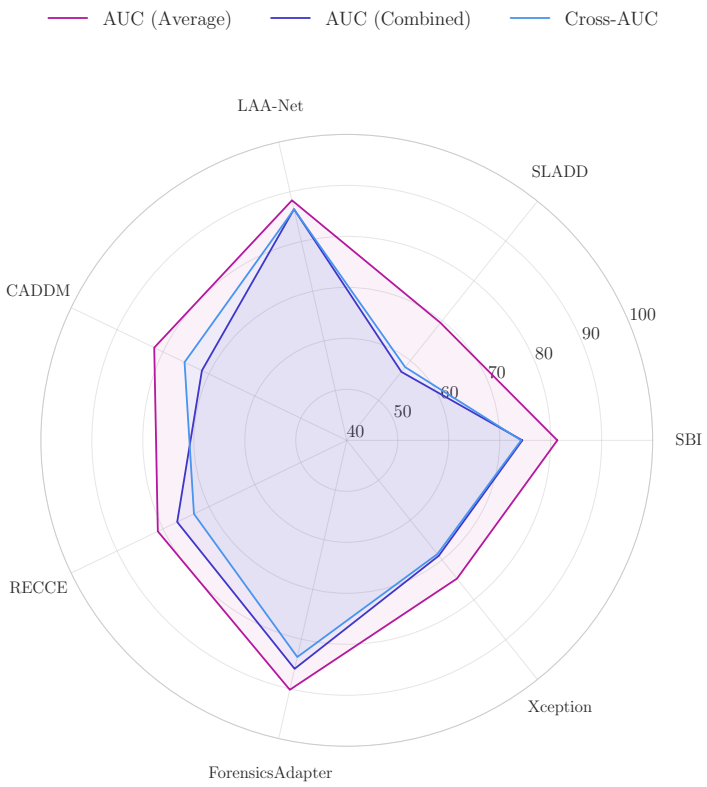


Figure 6.5 Direct comparison of average AUC, combined AUC, and the proposed Cross-AUC on the studied methods.

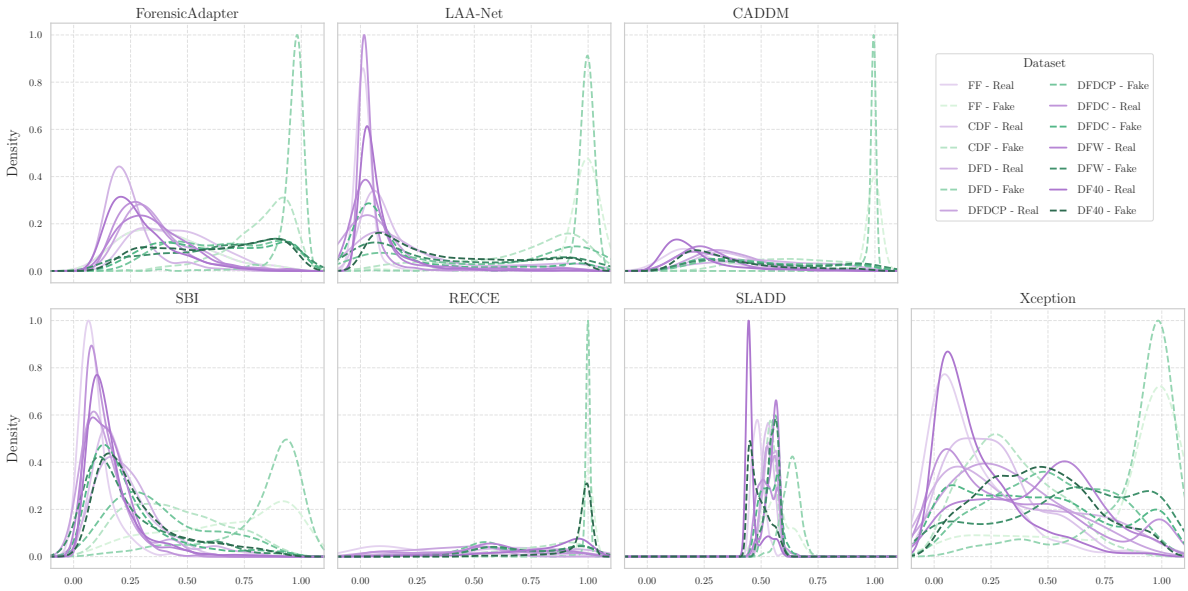


Figure 6.6 Prediction distributions of compared Deepfake detectors methods on seven datasets.

Detector	Backbone	Family	Respository	Weights Release	Venue
Xception [23]	Xception [46]	Binary Classifier	https://github.com/ondyari/FaceForensics	✓	ICCV'19
SLADD [31]	Modified Xception [46]	Multitask + Data synthesis	https://github.com/liangchen527/SLADD	×	CVPR'22
RECCE [32]	Designed Networks	Disentanglement Learning	https://github.com/VISION-SJTU/RECCE	✓	CVPR'22
SBI [26]	EfficientNet [45]	Data synthesis	https://github.com/mapooon/SelfBlendedImages	✓	CVPR'22
CADDM [27]	EfficientNet [45] + Multiscale	Multitask + Data synthesis	https://github.com/megvii-research/CADDM	✓	CVPR'23
LAA-Net [62]	EfficientNet [45] + E-FPN	Multitask + Data synthesis + Fine-grained	https://github.com/10Ring/LAA-Net	✓	CVPR'24
ForensicAdapter [28]	CLIP [74] + ViT [68]	Multitask + Data synthesis + Adapting CLIP	https://github.com/OUC-VAS/ForensicsAdapter	✓	CVPR'25

Table 6.1 Summary of compared deepfake detectors.

deepfake detectors is presented in Table 6.1. More details about these methods are provided in the supplementary materials.

Datasets. Seven standard and challenging datasets are used in our experiments: **FaceForensics++** (FF++) [23], **Celeb-DF** (CDF) [80], **Google Deepfake Detection** (DFD) [83], **WildDeepfake** (DFW) [82], **Deepfake Detection Challenge** (DFDC) [88], **Deepfake Detection Challenge Preview** (DFDCP) [81], and **DF40** [14]. FF++ is typically used for training, while the others serve as test sets in the cross-dataset evaluation protocol. These datasets are selected based on the following criteria: 1) they are widely used for evaluation in the research community; 2) they contain diverse source fakes, including those generated by black-box or undisclosed methods, for which no implementation details or prior knowledge are available; 3) they apply various perturbation methods. An overview of the selected benchmarks and further details are provided in the supplementary materials.

Setups. We extract 32 frames per video following the conventional test split. Facial regions are cropped using Face-RetinaNet [137]. These bounding boxes are slightly enlarged by a factor of 1.25 around the center of the face and then resized to a fixed resolution of 256×256. The open-source pre-trained models of the selected methods are used for all our experiments. However, since the pre-trained model of SLADD is not available, we retrain it on FF++ [23].

Evaluation Protocol. We consider both the conventional evaluation protocol, *i.e.*, cross-testing on individual datasets, and our suggested protocol for comparison. Specifically, to better simulate the anisotropic and multi-source nature of real-world data, we propose evaluating each model on a unified dataset, called Combined dataset, formed by aggregating all seven considered datasets. This reveals domain gaps that may be hidden in single-dataset evaluation settings.

Evaluation Metrics. The polarization is computed using the Wasserstein distance (W) between the estimated densities $R_{s,d}$ and $F_{s,d}$, corresponding to real and fake score distributions. To approximate these densities, we experimented with different density estimation functions Θ based on non-parametric and parametric approaches. For the former, we employ Kernel Density Estimation (KDE) and Quantile Cumulative Density Function (Q); for the latter, we consider two types of distributions, namely, Gaussian Mixture Model (GMM) and Beta Distribution (BD). To estimate the parameters of GMM and BD, Expectation-Maximization (EM) and numerical Maximum Likelihood Estimation (MLE) are used, respectively. When not specified, Q is chosen for its effectiveness. We mainly focus on comparing the averaged AUC (AUC_a), the AUC on the Combined dataset (AUC_c), and our Cross-AUC. Comparisons with additional evaluation metrics are provided in the supplementary materials for a more in-depth analysis, including Accuracy, Balanced Accuracy, Precision, Recall, Specificity, F1-Score, and Equal Error Rate.

6.4.2 Results and Discussions

Overall Performance Analysis. In Table 6.2, we evaluate seven deepfake detection models across seven datasets and a combined test set, using AUC, threshold stability (ϕ_τ), and distributional separability quantified by the polarities using four different kinds of estimated densities. It can be observed that ForensicAdapter [28] achieves the highest average AUC (90.18%) and a high Combined AUC (85.96%), indicating robust generalization across domains. It also consistently ranks among the best in distributional separability, reflecting reliable class boundaries. LAA-Net [62] attains the highest Combined AUC (86.51%) and leads in terms of polarity across most datasets, suggesting strong discriminative capacity. However, its higher threshold variance suggests less stable decision boundaries in cross-domain settings. RECCE [32] offers a strong balance between per-dataset AUC (81.13%), Combined AUC (76.91%), and low threshold variance, indicating consistent decision behavior across domains, though its class separability is slightly weaker. SBI [26] and CADDM [27] perform moderately well in terms of both average and Combined AUC, but show less consistent threshold behavior and lower score separability. Xception [46] and SLADD [31] exhibit lower AUCs

		FF++	CDF	DFD	DFW	DFDCP	DFDC	DF40	Average	**Combined**
Xception [46]	AUC (%)	93.6	62.1	92.24	63.71	70.56	58.98	81.59	74.68	68.95
	τ	0.4	0.24	0.7	0.57	0.4	0.3	0.24	0.4	0.31
	ϕ_τ	0.0004	0.0004	0.0002	0.0007	0.0010	0.0001	0.0001	0.0004	0.0001
	W_{KDE}	0.524	0.064	0.430	0.110	0.146	0.078	0.239	0.227	0.168
	W_{BD}	0.549	0.068	0.446	0.122	0.164	0.060	0.232	0.234	0.152
	W_{Q}	0.661	0.077	0.524	0.139	0.183	0.091	0.264	0.277	0.190
	W_{GMM}	0.616	0.069	0.496	0.149	0.118	0.081	0.249	0.254	0.180
SLADD [31]	AUC (%)	88.31	56.65	97.22	48.63	65.98	57.37	72.25	69.48	57.18
	τ	0.51	0.53	0.57	0.55	0.53	0.53	0.45	0.52	0.52
	ϕ_τ	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
	W_{KDE}	0.060	0.001	0.098	0.004	0.022	0.012	0.003	0.028	0.013
	W_{BD}	0.078	0.003	0.079	0.001	0.007	0.008	0.024	0.029	0.013
	W_{Q}	0.079	0.005	0.090	0.002	0.015	0.010	0.026	0.032	0.014
	W_{GMM}	0.079	0.005	0.089	0.004	0.015	0.010	0.027	0.033	0.014
RECCE [32]	AUC (%)	**99.56**	69.02	97.82	65.19	82.24	71.19	82.94	81.13	76.91
	τ	0.98	0.74	0.99	0.65	0.53	0.58	0.97	0.77	0.86
	ϕ_τ	0.00001	0.0002	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
	W_{KDE}	0.631	0.126	0.352	0.096	0.212	0.181	0.153	0.250	0.221
	W_{BD}	0.569	0.124	0.280	0.116	0.240	0.182	0.117	0.233	0.201
	W_{Q}	0.716	0.136	0.355	0.115	0.243	0.203	0.144	0.273	0.232
	W_{GMM}	0.692	0.134	0.349	0.111	0.227	0.203	0.142	0.265	0.228
SBI [26]	AUC (%)	98.23	84.65	96.78	60.83	86.83	69.77	72.05	81.3	74.43
	τ	0.21	0.28	0.46	0.16	0.19	0.14	0.17	0.23	0.17
	ϕ_τ	0.0006	0.0001	0.0008	0.0001	0.0002	0.0001	0.0001	0.0002	0.0001
	W_{KDW}	0.503	0.215	0.516	0.078	0.220	0.097	0.103	0.248	0.152
	W_{BD}	0.513	0.219	0.500	0.089	0.224	0.116	0.112	0.253	0.182
	W_{W}	0.576	0.238	0.577	0.085	0.244	0.108	0.116	0.278	0.172
	W_{GMW}	0.567	0.236	0.564	0.084	0.245	0.105	0.115	0.274	0.171
CADDM [27]	AUC (%)	99.26	80.7	99.52	76.31	71	70.33	76.28	81.91	71.53
	τ	0.97	0.45	0.93	0.37	0.44	0.37	0.22	0.53	0.31
	ϕ_τ	0.0253	0.0001	0.0001	0.0001	0.0011	0.0001	0.0001	0.0038	0.0001
	W_{KDE}	0.661	0.203	0.436	0.210	0.156	0.155	0.137	0.280	0.176
	W_{BD}	0.607	0.203	0.386	0.238	0.182	0.165	0.146	0.275	0.214
	W_{Q}	0.726	0.226	0.466	0.246	0.179	0.174	0.151	0.310	0.198
	W_{GMM}	0.714	0.224	0.459	0.241	0.174	0.173	0.150	0.305	0.197
LAA-Net [62]	AUC (%)	99.45	**95.45**	98.47	79.52	86.48	72.6	86.04	88.28	**86.51**
	τ	0.23	0.3	0.96	0.06	0.08	0.03	0.11	0.25	0.083
	ϕ_τ	0.0161	0.0004	0.0001	0.0001	0.0005	0.0001	0.0001	0.0024	0.0001
	W_{KDE}	**0.831**	**0.511**	0.633	**0.262**	**0.334**	0.169	0.281	**0.431**	0.302
	W_{BD}	**0.749**	**0.495**	0.547	0.248	**0.357**	0.189	0.272	**0.408**	0.296
	W_{Q}	**0.927**	**0.582**	0.708	**0.285**	**0.446**	0.156	0.299	**0.486**	0.318
	W_{GMM}	**0.911**	**0.579**	0.690	**0.292**	**0.417**	0.154	0.286	**0.475**	0.317
ForensicAdapter [28]	AUC (%)	98.39	94.94	**99.8**	**80.95**	**87.04**	**83.88**	**86.3**	**90.18**	85.96
	τ	0.76	0.64	0.65	0.5	0.43	0.4	0.38	0.53	0.43
	ϕ_τ	0.0013	0.0003	0.0027	0.0001	0.0007	0.0001	0.0001	0.0007	0.0001
	W_{KDE}	0.497	0.338	**0.659**	0.251	0.259	**0.257**	**0.291**	0.365	**0.304**
	W_{BD}	0.460	0.334	**0.624**	**0.253**	0.268	**0.266**	**0.294**	0.357	**0.306**
	W_{Q}	0.540	0.374	**0.713**	0.285	0.293	**0.286**	**0.322**	0.402	**0.336**
	W_{GMM}	0.521	0.371	**0.709**	0.284	0.288	**0.285**	**0.320**	0.397	**0.335**

Table 6.2 Performance of the selected models on all datasets. τ: Optimal threshold; ϕ_τ: variance of the top–K optimal thresholds; W_{KDE}, W_{Q}, W_{GMM}, W_{BD}: model polarities using Wasserstein distances on densities estimated with non-parametric approaches (Kernel Density Estimation (KDE), Quantile (Q)), and parametric approaches assuming two types of distributions, i.e., GMM and BD. Gray represents the datasets the model was trained on. Red highlights the best AUC and polarities.

	AUC_c	AUC_a	Cross-AUC	H-score	Δ_a	$\Delta_{\text{Cross-AUC}}$	$\Delta_{\text{H-score}}$
Xception [46]	68.95	74.68	**68.52**	72.38	5.73	**0.42**	2.42
SLADD [31]	57.18	69.48	**58.37**	65.88	12.3	**1.19**	8.70
RECCE [32]	76.91	81.13	73.26	**79.22**	4.22	3.64	**2.31**
SBI [26]	74.43	81.3	**74.20**	79.10	6.87	**0.22**	4.67
CADDM [27]	71.53	81.91	**75.31**	80.43	10.38	**3.78**	8.90
LAA-Net [62]	86.51	88.28	**86.47**	87.25	1.77	**0.04**	0.74
ForensicsAdapter [28]	85.96	90.18	**83.59**	89.66	4.22	**2.36**	3.70

Table 6.3 Overview of our predicted AUC, H-score [159], and the average AUC compared to the AUC of the combined dataset, where AUC_c is the AUC of the combined dataset, AUC_a is the mean of AUC from all the datasets, and our method Cross-AUC. Δ_a, $\Delta_{\text{Cross-AUC}}$, and $\Delta_{\text{H-score}}$ are the distances from AUC_c to AUC_a, Cross-AUC, and H-score, respectively. The closest AUC and distance to the combined set is marked with **Red**.

and weak generalization, confirming their limitations in complex, cross-domain settings. For a visual comparison of performance between detectors across datasets, see Figure 6.4. A more comprehensive view of performance with more evaluation metrics is reported in the supplementary materials.

Polarization Robustness under Domain Shift. We compare the polarization scores of each model by looking at their average values across individual datasets and their value on the combined dataset, as shown in Table 6.2. In general, most models show a lower polarization score on the combined set compared to the average across separate datasets. LAA-Net has the highest average polarization (e.g., average W_Q: 0.486) and performs well on several datasets such as CDF, DFW, DFDCP. However, it also shows the largest drop in the combined setting, with its score decreasing to 0.318. ForensicsAdapter has a slightly lower average 0.402, but its combined score remains relatively high 0.336, showing less difference between the two. This suggests that its behavior is more consistent across domains. RECCE shows moderate scores in both settings (0.273 average, 0.232 combined) with only a small change. SBI, CADDM, and Xception achieve strong scores on specific datasets (e.g., FF++, DFD), but their combined scores are notably lower. SLADD has low scores in both average 0.032 and combined 0.014 cases, with very little variation. These observations are further supported by Figure 6.6.

Overall, most models show a drop in polarization when moving from individual datasets to the combined set. This gap provides a simple indication of how much a model's ability to separate real and fake predictions is affected by domain shift. These results suggest that evaluating the polarization on mixed-domain inputs is useful to understand the robustness of the model.

Conventional Average AUC is Overoptimistic. Table 6.3 compares the conventional average AUC across individual datasets (AUC_a) with the AUC measured on the combined test

set (AUC_c). Across all models, AUC_a is consistently higher than AUC_c, with gaps ranging from 1.77% (LAA-Net) to over 12% (SLADD). This consistent drop highlights that average AUC often overstates generalization, as it reflects performance under domain-specific conditions with implicit per-domain advantages. However, models in deployment encounter inputs from mixed or unknown distributions, more closely represented by the combined test set. The lower AUC_c values expose how model predictions degrade under such conditions, revealing weaknesses that per-domain averages mask. To address this, we include our proposed Cross-AUC. In all cases, Cross-AUC is significantly closer to AUC_c than AUC_a, often reducing the error by more than 80%. This demonstrates the effectiveness of the proposed metric in approximating real-world performance. These results confirm that relying solely on average AUC can be misleading. Evaluating on combined distributions or computing such a metric is essential for assessing generalization capabilities.

AUC Generalizability with Cross-AUC. To assess how well performance on individual datasets generalizes to realistic, multi-domain deployment, we analyze the gap between predicted Cross-AUC and the combined AUC (AUC_c), as shown in Table 6.3. Unlike the conventional average AUC (AUC_a), which consistently overestimates performance under distribution shift, Cross-AUC is explicitly designed to estimate generalization using only per-dataset AUCs and score-level polarization, measured from Table 6.2. Across all models, Cross-AUC consistently aligns more closely with AUC_c than AUC_a, achieving the smallest difference ($\Delta_{\text{Cross-AUC}}$) in all cases. An alternative radar-based view of the comparison is given in Figure 6.5. Occasional *rank reversals* are also observed (e.g., ForensicsAdapter vs. LAA-Net; see Supplementary), where a model with the highest average AUC underperforms on both Combined AUC and Cross-AUC. Overall, these results demonstrate that Cross-AUC, computed based on per-domain performance and polarization, offers a robust and practical estimator of cross-domain generalization.

Cross-AUC versus H-score. H-score [159] was originally designed for universal domain adaptation as the harmonic mean of accuracies on common and unknown classes. To align with our setting, we replace accuracy with per-dataset AUC. While this adaptation provides a useful baseline, H-score does not reflect the effect of domain-dependent threshold shifts. Consequently, its estimates remain farther from the combined AUC compared to Cross-AUC (Table 6.3).

Out-of-Distribution (OOD) Evaluation with MagicBrush. We further evaluate generalization with MagicBrush [2], which contains manipulation types unseen in prior benchmarks. In addition to testing detectors on MagicBrush alone, we also include it as an eighth dataset and recompute all metrics across the expanded pool. Table 6.4 shows that the gap between

		MagicBrush	AUC_c	AUC_a	Cross-AUC
Xception [46]	AUC	44.90	66.70	70.96	**65.54**
	W_Q	0.029	0.170	0.246	
SLADD [31]	AUC	38.25	56.98	65.58	**58.44**
	W_Q	0.007	0.013	0.029	
RECCE [32]	AUC	53.70	72.23	77.70	**73.20**
	W_Q	0.001	0.196	0.239	
SBI [26]	AUC	86.71	72.79	81.98	**78.65**
	W_Q	0.209	0.187	0.269	
CADDM [27]	AUC	53.24	69.60	78.33	**74.08**
	W_Q	0.012	0.172	0.272	
LAA-Net [62]	AUC	55.73	85.26	**84.21**	80.03
	W_Q	0.018	0.309	0.428	
ForensicsAdapter [28]	AUC	59.42	83.22	86.34	**83.01**
	W_Q	0.061	0.298	0.359	

Table 6.4 A true out-of-distribution scenario with MagicBrush [2]. **Red** highlights the closest performance.

Model	AUC_c	Cross-AUC					
		Ψ_{Harmonic}	Δ	$\Psi_{\text{Arithmetic}}$	Δ	$\Psi_{\text{Geometric}}$	Δ
Xception [46]	68.95	**68.52**	**0.42**	69.93	0.98	67.29	1.66
SLADD [31]	57.18	**58.38**	**1.19**	60.80	3.61	59.72	2.53
RECCE [32]	76.91	73.27	3.64	**77.40**	**0.49**	73.98	2.92
SBI [26]	74.43	**74.21**	**0.22**	77.45	3.02	73.18	1.24
CADDM [27]	71.53	**75.32**	**3.78**	80.61	9.07	77.54	6.00
LAA-Net [62]	86.51	**86.47**	**0.04**	82.52	3.98	85.37	1.13
ForensicsAdapter [28]	85.96	**83.60**	**2.36**	82.02	3.94	82.94	3.02

Table 6.5 Estimated Cross-AUC with different Ψ functions, i.e., harmonic, arithmetic, and geometric means. The difference (Δ) between the AUC_c and the resulting Cross-AUC is reported for each function. **Red** highlights the closest performance.

average AUC and the combined AUC becomes even more pronounced, while Cross-AUC remains consistently close to the combined AUC. This demonstrates that Cross-AUC provides a reliable estimate not only under controlled benchmarks but also when truly novel manipulations are integrated into evaluation.

6.4.3 Ablation Studies

Effect of Mean Function in Cross-AUC Estimation. Table 6.5 presents an ablation study comparing three averaging functions (Ψ), harmonic, arithmetic, and geometric, used in our in our Cross-AUC. We evaluate their effectiveness by measuring the absolute difference with combined AUC (AUC_c). The results show that the harmonic mean consistently produces closest results, achieving the lowest difference (Δ) in 6 out of 7 models. For example, it

	Xception	SLADD	RECCE	SBI	CADDM	LAA-Net	ForensicsAdapter	Avg.
AUC_c	68.95	57.18	76.91	74.43	71.53	86.51	85.96	74.49
Cross-AUC_{WD}	**68.52**	**58.37**	73.26	**74.20**	**75.31**	86.47	83.59	**74.24**
Cross-AUC_{KL}	68.29	60.07	**74.22**	73.88	76.80	**86.54**	**87.05**	75.26
Cross-AUC_{JS}	54.61	65.73	60.82	66.05	62.22	67.48	63.45	62.90

Table 6.6 Estimated Cross-AUC with different polarization estimators, i.e., Wasserstein Distance (WD), KL Divergence (KL), and Jensen-Shannon (JS) Distance. **Red** highlights the best estimator.

	AUC_c	Cross-AUC									
		λ=0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Xception [46]	68.95	71.61	70.83	70.06	**69.29**	68.52	67.75	66.98	66.20	65.43	64.66
SLADD [31]	57.18	64.37	62.87	61.37	59.87	58.37	**56.87**	55.37	53.87	52.37	50.86
RECCE [32]	76.91	78.03	**76.83**	75.64	74.45	73.26	72.07	70.88	69.69	68.50	67.30
SBI [26]	74.43	78.12	77.14	76.16	75.18	**74.20**	73.22	72.24	71.26	70.28	69.30
CADDM [27]	71.53	79.40	78.38	77.36	76.34	75.31	74.29	73.27	72.25	**71.22**	70.20
LAA-Net [62]	86.51	87.09	86.94	86.78	86.62	**86.47**	86.30	86.15	85.99	85.83	85.67
ForensicsAdapter [28]	85.96	88.45	87.23	**86.02**	84.81	83.59	82.38	81.16	79.95	78.74	77.52

Table 6.7 Effects of the balancing factor λ. **Red** highlights the most balancing λ value.

predicts within 0.42 points for Xception, 1.19 for SLADD, and only 0.04 for LAA-Net.

Cross-AUC with Different Polarization Estimators. We further examine the effect of replacing the Wasserstein Distance in Cross-AUC with alternative separability measures. As shown in Table 6.6, both Wasserstein and KL-based Cross-AUC remain closely aligned with the combined AUC, whereas the Jensen-Shannon (JS) variant yields less consistent estimates. This suggests that while Cross-AUC is generally robust to the choice of estimator, Wasserstein offers the most stable approximation under distribution shift.

Effect of Balancing Factor λ. We also examine the influence of the balancing factor λ in Cross-AUC, which controls the trade-off between mean AUC and polarization. As shown in Table 6.7, very small λ values reduce Cross-AUC to a simple average AUC and thus deviate from the combined AUC, while excessively large λ values overweight polarization and also degrade alignment. A mid-range choice ($\lambda \approx 0.5$) consistently yields the closest approximation to AUC_c, confirming that Cross-AUC is robust provided λ is set within a reasonable range.

Runtime Analysis. We report runtime for different polarization estimators in Table 6.8. Non-parametric variants (W_{KDE}, W_Q) incur only millisecond-level overhead, while the GMM-based estimator (W_{GMM}) is slower by orders of magnitude. Nevertheless, all variants remain practical for offline evaluation, and the default Wasserstein implementation achieves a very good balance between efficiency and fidelity.

	FF++	CDF	DFD	DFW	DFDCP	DFDC	DF40	Combined
W_{KDE}	0.0014	0.0012	0.0012	**0.0010**	0.0012	0.0015	0.0023	0.0036
W_{BD}	0.0013	0.0012	0.0011	0.0011	0.0014	**0.0011**	**0.0012**	**0.0013**
W_{Q}	**0.0008**	**0.0008**	**0.0008**	0.0013	**0.0007**	0.0129	0.0148	0.0847
W_{GMM}	2.2309	2.1118	2.2576	2.7162	2.0931	4.8132	5.2694	8.3977

Table 6.8 Computational complexity (results reported in seconds). **Red** highlights the most computationally efficient estimator.

6.4.4 Potential Impact on the Research Community

This work introduces a simple yet effective framework for assessing model generalization under domain shift. By moving beyond conventional per-domain metrics and incorporating score-level polarization, Cross-AUC can be used as a replacement for average AUC, which can obscure real-world failure cases. As such, it provides researchers with a more reliable estimate of performance under uncontrolled conditions. More broadly, this highlights the need for a shift in the community toward evaluation protocols that prioritize reliability and robustness across domains over isolated benchmarks.

6.5 Conclusion

This work addresses the need to more accurately assess the generalization capabilities of deepfake detectors, as current evaluation practices using separate AUC scores across datasets do not capture real-world scenarios involving mixed-domain data. To solve this, we propose Cross-dataset AUC (Cross-AUC), a metric that combines mean AUC with the polarization of prediction scores, measured using the Wasserstein Distance. This approach captures both ranking performance and cross-domain score separability, thereby explaining the gap between average and combined AUC and providing a more realistic evaluation of robustness under domain shift. Our experiments show that most state-of-the-art detectors do not generalize as well as suggested by average AUC, while Cross-AUC consistently provides estimates closer to the combined AUC. Beyond ranking models, Cross-AUC highlights the importance of prediction polarization as a measure of generalization stability.

Chapter 7

Conclusion

This chapter concludes this work on deepfake detection under realistic and challenging conditions. It first provides a concise summary of the main contributions and findings, and then discusses several possible future research directions. Together, these elements aim to highlight how the proposed frameworks advance the state of the art and outline how they could be further extended in future investigations.

7.1 Summary

This thesis has addressed the problem of reliable deepfake detection in the era of increasingly realistic generative models. Despite rapid progress in deep learning-based detectors, a clear performance gap persists between laboratory and real-world conditions. This is caused by limited generalization to unseen manipulations, lack of robustness to high-quality (HQ) forgeries and perturbations. Furthermore, current evaluation protocols do not fully reflect the effectiveness of existing methods in real-world conditions. Taking into account the typical deepfake generation process, data synthesis, multi-task learning strategies, multi-scale feature representations, transformer architectures have been developed towards generalizable, robust, interpretable deepfake detection. In addition, a new measure called cross-AUC has been proposed to better assess the suitability of current deepfake detectors in an unconstrained context.

The first contribution of this thesis is LAA-Net, a CNN-based multi-task learning framework for generalizable yet quality-agnostic deepfake image detection. LAA-Net introduces an explicit attention mechanism that focuses on vulnerable pixels, which are defined as the pixels that are more likely to exhibit blending artifacts, through two auxiliary branches: a heatmap branch and a self-consistency branch. These branches are trained using pseudo-fakes gener-

ated via blending-based data synthesis, while an Enhanced Feature Pyramid Network (E-FPN) aggregates multi-scale features without incurring redundancy. Together, these components allow the network to capture both subtle and global inconsistencies, leading to improved and more stable cross-dataset performance in terms of AUC and AP on several benchmarks. It also provides enhanced interpretability via localized output maps.

As a second contribution, we extend this vulnerability-aware design to transformer architectures. As ViT tend to favor global context at the expense of fine-grained cues, we propose to overcome that by introducing LAA-Former and its Swin-based counterpart, LAA-Swin, which integrate a lightweight Learning-based Local Attention (L2-Att) module. L2-Att generalizes the notion of vulnerable pixels to vulnerable patches, enabling transformers to explicitly attend to artifact-prone regions while preserving their ability to model long-range dependencies. Extensive experiments across multiple datasets show that these transformer-based variants not only enhance generalization and robustness to perturbations but also achieve competitive or superior performance compared to both CNN-based and transformer-based state-of-the-art methods. They also rely on a reduced model size and computational cost, without requiring large-scale training data.

As a third contribution, we extend the proposed vulnerability-driven deepfake detection framework to the video level. Specifically, we propose FakeSTormer, a multi-task learning framework for generalizable deepfake video detection that treats the problem as a fine-grained spatio-temporal detection task. FakeSTormer augments a standard classification head with two auxiliary branches: a temporal regression branch that localizes vulnerability-prone temporal locations, and a spatial branch that predicts frame-wise spatial vulnerabilities. To support these branches, a high-quality video-level data synthesis method, Self-Blended Video (SBV), is introduced to generate temporally coherent pseudo-fakes. Using a revisited version of TimeSformer with dedicated classification tokens, the resulting model is able to focus on a small number of informative spatial and temporal regions. As a result, it achieves superior cross-dataset performance and robustness to HQ forgeries compared to existing video-based detectors, while providing more interpretable saliency visualizations.

Finally, we propose to revisit the commonly used cross-dataset deepfake detection evaluation. While average AUC across isolated datasets is widely used, it fails to capture the behavior of detectors under realistic mixed-domain conditions, where data from heterogeneous sources coexist. To overcome this, this thesis introduces Cross-dataset AUC (Cross-AUC), a new metric that combines the mean AUC with the extent of probability prediction polarization estimated via the Wasserstein Distance between positive and negative score distributions. Cross-AUC accounts simultaneously for ranking performance and score separability across

domains and has been empirically shown to closely approximate the AUC obtained when all datasets are combined. This reveals that most state-of-the-art detectors still lack generalization capabilities, highlighting the need to consider prediction polarization as a key performance indicator.

7.2 Future Directions

Building on the contributions and findings discussed above, several promising research directions can be considered to further advance deepfake detection under realistic settings. While the proposed frameworks improve generalization, robustness, and interpretability across multiple datasets and modalities, a number of open challenges still exist, particularly in terms of explainability, universality across manipulation types, and deployment under real-world constraints. This section outlines possible extensions of this work along these axes and highlights how future methods could better bridge the gap between controlled benchmarks and practical forensic applications.

7.2.1 Interpretable Deepfake Detection and Localization

In this thesis, first steps towards interpretability are taken through the proposed vulnerability-based mechanisms. The latter provides localized maps that highlight artifact-prone regions. A natural extension is to move beyond visual explanations toward more explicit, human-understandable explanations. Future work could investigate detectors that not only localize forged regions more precisely in space and time, but also describe why a sample is considered fake. For instance, this might be done by linking detected artifacts to specific types of inconsistencies, e.g., blending seams, illumination mismatches, or temporal discontinuities.

7.2.2 Towards Universal Forgery Detection

A central challenge highlighted throughout this thesis is the limited generalization of current detectors across datasets, manipulation types, and acquisition conditions. Although the proposed vulnerability-aware frameworks improve robustness to unseen manipulations, they still operate on the same type of deepfakes, i.e face-swaps. A natural next step is to move towards more "universal" or manipulation-agnostic detectors that can handle a broader spectrum of forgeries, including emerging and unseen manipulation families, and potentially non-facial content.

Future work could explore architectures and training strategies explicitly designed for open-set scenarios, where detectors must maintain reliable performance when dealing with different deepfake types, or non-facial manipulations.

7.2.3 Efficient and Continually Adaptive Deepfake Detection in the Wild

While the proposed frameworks demonstrate strong performance under cross-dataset evaluation, their deployment in real-world environments raises additional constraints related to efficiency, latency, and continual adaptation. In many practical scenarios, deepfake detection must run on resource-limited devices (*e.g.*, mobile phones, embedded cameras, or edge nodes) and operate in (near) real time. Future work could therefore focus on designing lightweight, vulnerability-aware architectures that preserve the benefits of localized and spatio-temporal modeling while reducing memory and computational cost. This may involve model compression, pruning, quantization, or knowledge distillation tailored to deepfake detection, as well as exploring compact backbone architectures that remain robust under compression.

Beyond efficiency, real-world deployment also requires detectors to cope with continuous distribution shifts caused by new manipulation techniques, changing acquisition conditions, and evolving content platforms. Rather than relying solely on static, offline training, an important direction is to endow deepfake detectors with online or continual learning capabilities, enabling them to update their decision boundaries in response to new data with minimal supervision and without retraining from scratch.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger. Vol. 27. Curran Associates, Inc., 2014.

[2] K. Zhang, L. Mo, W. Chen, H. Sun, and Y. Su. "Magicbrush: A manually annotated dataset for instruction-guided image editing". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 31428–31449.

[3] Y. Alaluf, O. Patashnik, Z. Wu, A. Zamir, E. Shechtman, D. Lischinski, and D. Cohen-Or. "Third Time's the Charm? Image and Video Editing with StyleGAN3". In: *CoRR* abs/2201.13433 (2022).

[4] T.-J. Fu, X. E. Wang, S. T. Grafton, M. P. Eckstein, and W. Y. Wang. "M3L: Language-Based Video Editing via Multi-Modal Multi-Level Transformers". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 10513–10522.

[5] OpenAI. *ChatGPT (Large Language Model)*. Version June 14 2024. 2024.

[6] G. LLC. *Gemini (Large Language Model)*. Version 1.0. 2023.

[7] xAI. *Grok (Large Language Model)*. Version 4.0. 2025.

[8] L. French. *Deepfake face swap attacks on ID verification systems up 704% in 2023*. Online access 7-Feb-2024. 2024. URL: https://www.scworld.com/news/deepfake-face-swap-attacks-on-id-verification-systems-up-704-in-2023.

[9] E. H. Schwartz. *Samsung's New MegaPortrait Generates Deepfake Videos from Still Images*. Online access 1-Aug-2022. 2022. URL: https://voicebot.ai/2022/08/01/samsungs-new-megaportrait-generates-deepfake-videos-from-still-images/.

[10] R. Shao, T. Wu, and Z. Liu. "Detecting and recovering sequential deepfake manipulation". In: *European Conference on Computer Vision*. Springer. 2022, pp. 712–728.

[11] J. West and C. Bergstrom. *Which Face Is Real?* Retrieved from "Which Face Is Real?" website. 2019. URL: https://www.whichfaceisreal.com/methods.html.

[12] Y. Nirkin, Y. Keller, and T. Hassner. "FSGANv2: Improved Subject Agnostic Face Swapping and Reenactment". In: *TPAMI*. IEEE, 2022.

[13] W. Zhao, Y. Rao, W. Shi, Z. Liu, J. Zhou, and J. Lu. "DiffSwap: High-Fidelity and Controllable Face Swapping via 3D-Aware Masked Diffusion". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 8568–8577.

[14] Z. Yan et al. "DF40: Toward Next-Generation Deepfake Detection". In: *Advances in Neural Information Processing Systems*. Ed. by A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang. Vol. 37. Curran Associates, Inc., 2024, pp. 29387–29434.

[15] V. Danry, J. Leong, P. Pataranutaporn, P. Tandon, Y. Liu, R. Shilkrot, P. Punpongsanon, T. Weissman, P. Maes, and M. Sra. "AI-generated characters: putting deepfakes to good use". In: *CHI conference on human factors in computing systems extended abstracts*. 2022, pp. 1–5.

[16] E. Lundberg and P. Mozelius. "The potential effects of deepfakes on news media and entertainment". In: *AI & SOCIETY* 40.4 (2025), pp. 2159–2170.

[17] J. Wakefield. *Deepfake presidents used in Russia-Ukraine war*. https://www.bbc.com/news/technology-60780142. [Online; accessed 7-March-2023]. 2022.

[18] S. Cahlan. *How misinformation helped spark an attempted coup in Gabon*. https://wapo.st/3KZARDF. [Online; accessed 7-March-2023]. 2020.

[19] H. Chen and K. Magramo. *Finance worker pays out $25 million after video call with deepfake "chief financial officer"*. https://edition.cnn.com/2024/02/04/asia/deepfake-cfo-scam-hong-kong-intl-hnk/index.html. [Online; accessed 4-February-2024]. 2024.

[20] Reuters. *South Korea to criminalize watching or possessing sexually explicit deepfakes*. https://edition.cnn.com/2024/09/26/asia/south-korea-deepfake-bill-passed-intl-hnk/index.html. [Online; accessed 26-September-2024]. 2024.

[21] M. S. Rana, B. Murali, and A. H. Sung. "Deepfake detection using machine learning algorithms". In: *2021 10th International Congress on Advanced Applied Informatics (IIAI-AAI)*. IEEE. 2021, pp. 458–463.

[22] A. Mitra, S. P. Mohanty, P. Corcoran, and E. Kougianos. “A machine learning based approach for deepfake detection in social media through key video frame extraction”. In: *SN Computer Science* 2.2 (2021), p. 98.

[23] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner. “FaceForensics++: Learning to Detect Manipulated Facial Images”. In: *International Conference on Computer Vision (ICCV)*. 2019.

[24] Y. Qian, G. Yin, L. Sheng, Z. Chen, and J. Shao. “Thinking in frequency: Face forgery detection by mining frequency-aware clues”. In: *European conference on computer vision*. Springer. 2020, pp. 86–103.

[25] L. Li, J. Bao, T. Zhang, H. Yang, D. Chen, F. Wen, and B. Guo. “Face X-Ray for More General Face Forgery Detection”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[26] K. Shiohara and T. Yamasaki. “Detecting Deepfakes with Self-Blended Images”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 18720–18729.

[27] S. Dong, J. Wang, R. Ji, J. Liang, H. Fan, and Z. Ge. “Implicit Identity Leakage: The Stumbling Block to Improving Deepfake Detection Generalization”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 3994–4004.

[28] X. Cui, Y. Li, A. Luo, J. Zhou, and J. Dong. “Forensics Adapter: Adapting CLIP for Generalizable Face Forgery Detection”. In: *arXiv preprint arXiv:2411.19715* (2024).

[29] Z. Yan, Y. Zhao, S. Chen, M. Guo, X. Fu, T. Yao, S. Ding, Y. Wu, and L. Yuan. “Generalizing Deepfake Video Detection with Plug-and-Play: Video-Level Blending and Spatiotemporal Adapter Tuning”. In: *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*. June 2025, pp. 12615–12625.

[30] K. Lin, Y. Lin, W. Li, T. Yao, and B. Li. “Standing on the shoulders of giants: Reprogramming visual-language model for general deepfake detection”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 39. 5. 2025, pp. 5262–5270.

[31] L. Chen, Y. Zhang, Y. Song, L. Liu, and J. Wang. “Self-Supervised Learning of Adversarial Example: Towards Good Generalizations for Deepfake Detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 18710–18719.

[32] J. Cao, C. Ma, T. Yao, S. Chen, S. Ding, and X. Yang. “End-to-End Reconstruction-Classification Learning for Face Forgery Detection”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 4103–4112. DOI: 10.1109/CVPR52688.2022.00408.

[33] E. Zhao, X. Xu, M. Xu, H. Ding, Y. Xiong, and W. Xia. “Learning self-consistency for deepfake detection”. In: *ICCV 2021*. 2021.

[34] J. Wang, Z. Wu, W. Ouyang, X. Han, J. Chen, Y.-G. Jiang, and S.-N. Li. “M2tr: Multi-modal multi-scale transformers for deepfake detection”. In: *Proceedings of the 2022 international conference on multimedia retrieval*. 2022, pp. 615–623.

[35] A. Khormali and J.-S. Yuan. “DFDT: An End-to-End DeepFake Detection Framework Using Vision Transformer”. In: *Applied Sciences* (2022).

[36] Y. Xu, J. Liang, G. Jia, Z. Yang, Y. Zhang, and R. He. “TALL: Thumbnail Layout for Deepfake Video Detection”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2023, pp. 22658–22668.

[37] X. Dong, J. Bao, D. Chen, T. Zhang, W. Zhang, N. Yu, D. Chen, F. Wen, and B. Guo. “Protecting Celebrities From DeepFake With Identity Consistency Transformer”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 9468–9478.

[38] Y. Zhang, B. Colman, X. Guo, A. Shahriyari, and G. Bharaj. “Common sense reasoning for deepfake detection”. In: *European conference on computer vision*. Springer. 2024, pp. 399–415.

[39] H. Zhao, W. Zhou, D. Chen, T. Wei, W. Zhang, and N. Yu. “Multi-attentional deepfake detection”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 2185–2194.

[40] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen. “MesoNet: a Compact Facial Video Forgery Detection Network”. In: *CoRR* abs/1809.00888 (2018).

[41] R. Wang, L. Ma, F. Juefei-Xu, X. Xie, J. Wang, and Y. Liu. “FakeSpotter: A Simple Baseline for Spotting AI-Synthesized Fake Faces”. In: *CoRR* abs/1909.06122 (2019).

[42] H. H. Nguyen, J. Yamagishi, and I. Echizen. “Capsule-Forensics: Using Capsule Networks to Detect Forged Images and Videos”. In: *CoRR* abs/1810.11215 (2018).

[43] Y. Li and S. Lyu. “Exposing DeepFake Videos By Detecting Face Warping Artifacts”. In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2019.

[44] Z. Yan, Y. Zhang, Y. Fan, and B. Wu. "UCF: Uncovering Common Features for Generalizable Deepfake Detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2023, pp. 22412–22423.

[45] M. Tan and Q. V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: *CoRR* abs/1905.11946 (2019).

[46] F. Chollet. "Xception: Deep learning with depthwise separable convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1251–1258.

[47] K. He, X. Zhang, S. Ren, and J. Sun. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015).

[48] Y. Wang, K. Yu, C. Chen, X. Hu, and S. Peng. "Dynamic Graph Learning With Content-Guided Spatial-Frequency Relation Reasoning for Deepfake Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 7278–7287.

[49] Z. Xu, Z. Hong, C. Ding, Z. Zhu, J. Han, J. Liu, and E. Ding. "MobileFaceSwap: A Lightweight Framework for Video Face Swapping". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2022.

[50] Z. Gu, Y. Chen, T. Yao, S. Ding, J. Li, F. Huang, and L. Ma. "Spatiotemporal Inconsistency Learning for DeepFake Video Detection". In: *Proceedings of the 29th ACM International Conference on Multimedia* (2021).

[51] I. Ganiyusufoglu, L. M. Ngô, N. Savov, S. Karaoglu, and T. Gevers. "Spatio-temporal Features for Generalized Detection of Deepfake Videos". In: *CoRR* abs/2010.11844 (2020).

[52] Y. Zheng, J. Bao, D. Chen, M. Zeng, and F. Wen. "Exploring Temporal Coherence for More General Video Face Forgery Detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 15044–15054.

[53] C. Zhao, C. Wang, G. Hu, H. Chen, C. Liu, and J. Tang. "ISTVT: Interpretable Spatial-Temporal Video Transformer for Deepfake Detection". In: *IEEE Transactions on Information Forensics and Security* 18 (2023), pp. 1335–1348. DOI: 10.1109/TIFS.2023.3239223.

[54] A. Haliassos, K. Vougioukas, S. Petridis, and M. Pantic. "Lips Don't Lie: A Generalisable and Robust Approach to Face Forgery Detection". In: *CoRR* abs/2012.07657 (2020).

[55] D. Zhang, F. Lin, Y. Hua, P. Wang, D. Zeng, and S. Ge. “Deepfake video detection with spatiotemporal dropout transformer”. In: *Proceedings of the 30th ACM international conference on multimedia*. 2022, pp. 5833–5841.

[56] Z. Wang, J. Bao, W. Zhou, W. Wang, and H. Li. “AltFreezing for More General Video Face Forgery Detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 4129–4138.

[57] W. Guan, W. Wang, B. Peng, J. Dong, and T. Tan. “ST-SBV: Spatial-Temporal Self-Blended Videos for Deepfake Detection”. In: *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*. Springer. 2024, pp. 274–288.

[58] M. Li, X. Li, K. Yu, C. Deng, H. Huang, F. Mao, H. Xue, and M. Li. “Spatio-temporal catcher: A self-supervised transformer for deepfake video detection”. In: *Proceedings of the 31st ACM International Conference on Multimedia*. 2023, pp. 8707–8718.

[59] P. Yue, B. Chen, and Z. Fu. “Local region frequency guided dynamic inconsistency network for deepfake video detection”. In: *Big Data Mining and Analytics* 7.3 (2024), pp. 889–904.

[60] L. Jiang, R. Li, W. Wu, C. Qian, and C. C. Loy. “DeeperForensics-1.0: A Large-Scale Dataset for Real-World Face Forgery Detection”. In: *CVPR*. 2020.

[61] B. M. Le and S. S. Woo. “Quality-Agnostic Deepfake Detection with Intra-model Collaborative Learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2023, pp. 22378–22389.

[62] D. Nguyen, N. Mejri, I. P. Singh, P. Kuleshova, M. Astrid, A. Kacem, E. Ghorbel, and D. Aouada. “LAA-Net: Localized Artifact Attention Network for Quality-Agnostic and Generalizable Deepfake Detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2024, pp. 17395–17405.

[63] S. Chen, T. Yao, Y. Chen, S. Ding, J. Li, and R. Ji. “Local Relation Learning for Face Forgery Detection”. In: *AAAI Conference on Artificial Intelligence*. 2021.

[64] L. Chen, Y. Zhang, Y. Song, J. Wang, and L. Liu. “OST: Improving Generalization of DeepFake Detection via One-Shot Test-Time Training”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh. Vol. 35. Curran Associates, Inc., 2022, pp. 24597–24610.

[65] W. Bai, Y. Liu, Z. Zhang, B. Li, and W. Hu. “AUNet: Learning Relations Between Action Units for Face Forgery Detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 24709–24719.

[66] M. Astrid, E. Ghorbel, and D. Aouada. “Statistics-aware Audio-visual Deepfake Detector”. In: *arXiv preprint arXiv:2407.11650* (2024).

[67] M. Astrid, E. Ghorbel, and D. Aouada. “Detecting Audio-Visual Deepfakes with Fine-Grained Inconsistencies”. In: *arXiv preprint arXiv:2408.06753* (2024).

[68] A. Dosovitskiy et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *CoRR* abs/2010.11929 (2020).

[69] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. “Training data-efficient image transformers & distillation through attention”. In: *CoRR* abs/2012.12877 (2020).

[70] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. “End-to-End Object Detection with Transformers”. In: *CoRR* abs/2005.12872 (2020).

[71] Y. Xu, J. Zhang, Q. Zhang, and D. Tao. “ViTPose: Simple Vision Transformer Baselines for Human Pose Estimation”. In: *Advances in Neural Information Processing Systems*. 2022.

[72] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. “Masked Autoencoders Are Scalable Vision Learners”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 16000–16009.

[73] 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*. Ed. by S. Sclaroff, C. Distante, M. Leo, G. M. Farinella, and F. Tombari. Cham: Springer International Publishing, 2022, pp. 219–229.

[74] A. Radford et al. “Learning Transferable Visual Models From Natural Language Supervision”. In: *International Conference on Machine Learning*. 2021.

[75] J. Li, D. Li, C. Xiong, and S. Hoi. “Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation”. In: *International conference on machine learning*. PMLR. 2022, pp. 12888–12900.

[76] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al. “Lora: Low-rank adaptation of large language models.” In: *ICLR* 1.2 (2022), p. 3.

[77] Z. Xu, X. Zhang, R. Li, Z. Tang, Q. Huang, and J. Zhang. “FakeShield: Explainable Image Forgery Detection and Localization via Multi-modal Large Language Models”. In: *International Conference on Learning Representations*. 2025.

[78] Z. Gu, Y. Chen, T. Yao, S. Ding, J. Li, and L. Ma. "Delving into the local: Dynamic inconsistency learning for deepfake video detection". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 36. 1. 2022, pp. 744–752.

[79] S. Banerjee, W. J. Scheirer, K. Bowyer, and P. J. Flynn. "On Hallucinating Context and Background Pixels from a Face Mask using Multi-scale GANs". In: *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2018), pp. 289–298.

[80] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu. "Celeb-DF: A Large-Scale Challenging Dataset for DeepFake Forensics". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[81] B. Dolhansky, R. Howes, B. Pflaum, N. Baram, and C. Canton-Ferrer. "The Deepfake Detection Challenge (DFDC) Preview Dataset". In: *CoRR* abs/1910.08854 (2019).

[82] B. Zi, M. Chang, J. Chen, X. Ma, and Y.-G. Jiang. "WildDeepfake: A Challenging Real-World Dataset for Deepfake Detection". In: *Proceedings of the 28th ACM International Conference on Multimedia* (2020).

[83] N. Dufour and A. Gully. *Contributing data to deepfake detection research*. `https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html`. 2019.

[84] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, and C. Shen. "Twins: Revisiting the Design of Spatial Attention in Vision Transformers". In: *Neural Information Processing Systems*. 2021.

[85] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows". In: *CoRR* abs/2103.14030 (2021).

[86] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao. "Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 548–558.

[87] A. Hassani, S. Walton, J. Li, S. Li, and H. Shi. "Neighborhood Attention Transformer". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 6185–6194.

[88] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. Canton-Ferrer. "The DeepFake Detection Challenge Dataset". In: *CoRR* abs/2006.07397 (2020).

[89] G. Bertasius, H. Wang, and L. Torresani. “Is Space-Time Attention All You Need for Video Understanding?” In: *Proceedings of the International Conference on Machine Learning (ICML)*. July 2021.

[90] D. Nguyen, M. Astrid, A. Kacem, E. Ghorbel, and D. Aouada. “Vulnerability-Aware Spatio-Temporal Learning for Generalizable Deepfake Video Detection”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2025, pp. 10786–10796.

[91] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros. “CNN-generated images are surprisingly easy to spot... for now”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 8695–8704.

[92] S. Ren, K. He, R. Girshick, and J. Sun. “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *Advances in neural information processing systems* 28 (2015).

[93] C. Wang and W. Deng. “Representative Forgery Mining for Fake Face Detection”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021.

[94] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. “Feature pyramid networks for object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2117–2125.

[95] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár. “Focal Loss for Dense Object Detection”. In: *CoRR* abs/1708.02002 (2017).

[96] S. S. Seferbekov, V. I. Iglovikov, A. V. Buslaev, and A. A. Shvets. “Feature Pyramid Network for Multi-Class Land Segmentation”. In: *CoRR* abs/1806.03510 (2018).

[97] M. Tan, R. Pang, and Q. V. Le. “EfficientDet: Scalable and Efficient Object Detection”. In: *CoRR* abs/1911.09070 (2019).

[98] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. “Emerging Properties in Self-Supervised Vision Transformers”. In: *CoRR* abs/2104.14294 (2021).

[99] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, pp. 4171–4186.

[100] C.-F. (Chen, Q. Fan, and R. Panda. “CrossViT: Cross-Attention Multi-Scale Vision Transformer for Image Classification”. In: *International Conference on Computer Vision (ICCV)*. 2021.

[101] X. Chu, Z. Tian, B. Zhang, X. Wang, and C. Shen. “Conditional Positional Encodings for Vision Transformers”. In: *International Conference on Learning Representations*. 2021.

[102] M. Rodrigo, C. Cuevas, and N. García. “Comprehensive comparison between vision transformers and convolutional neural networks for face recognition tasks”. In: *Scientific reports* 14.1 (2024), p. 21392.

[103] B. O. Ayinde, T. Inanc, and J. M. Zurada. “Regularizing deep neural networks by enhancing diversity in feature extraction”. In: *IEEE transactions on neural networks and learning systems* 30.9 (2019), pp. 2650–2661.

[104] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool. “Pose guided person image generation”. In: *Advances in neural information processing systems* 30 (2017).

[105] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: 10.1109/TIP.2003.819861.

[106] F. Matern, C. Riess, and M. Stamminger. “Exploiting Visual Artifacts to Expose Deepfakes and Face Manipulations”. In: *2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*. 2019, pp. 83–92. DOI: 10.1109/WACVW.2019.00020.

[107] T. T. Nguyen, C. M. Nguyen, D. T. Nguyen, D. T. Nguyen, and S. Nahavandi. “Deep Learning for Deepfakes Creation and Detection”. In: *CoRR* abs/1909.11573 (2019).

[108] I. P. Singh, N. Mejri, v. D. Nguyen, E. Ghorbel, and D. Aouada. “Multi-label deepfake classification”. In: *IEEE Workshop on Multimedia Signal Processing* (2023).

[109] A. Haliassos, R. Mira, S. Petridis, and M. Pantic. “Leveraging real talking faces via self-supervision for robust forgery detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 14950–14962.

[110] H. Law and J. Deng. “CornerNet: Detecting Objects as Paired Keypoints”. In: *International Journal of Computer Vision* 128 (2018), pp. 642–656.

[111] Z. Tian, C. Shen, H. Chen, and T. He. “FCOS: Fully Convolutional One-Stage Object Detection”. In: *CoRR* abs/1904.01355 (2019).

[112] Z. Sun, Y. Han, Z. Hua, N. Ruan, and W. Jia. “Improving the Efficiency and Robustness of Deepfakes Detection through Precise Geometric Features”. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 3608–3617.

[113] B. Liu, B. Liu, M. Ding, T. Zhu, and X. Yu. “TI2Net: Temporal Identity Inconsistency Network for Deepfake Detection”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Jan. 2023, pp. 4691–4700.

[114] B. Huang, Z. Wang, J. Yang, J. Ai, Q. Zou, Q. Wang, and D. Ye. “Implicit Identity Driven Deepfake Face Swapping Detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 4490–4499.

[115] Y. Guo, C. Zhen, and P. Yan. “Controllable Guide-Space for Generalizable Face Forgery Detection”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2023, pp. 20818–20827.

[116] Deepfakes. *FaceSwapDevs*. `https://github.com/deepfakes/faceswap`. 2019.

[117] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner. “Face2Face: Real-time Face Capture and Reenactment of RGB Videos”. In: *CoRR* abs/2007.14808 (2020).

[118] M. Kowalski. *FaceSwap*. `https://github.com/MarekKowalski/FaceSwap`. 2018.

[119] J. Thies, M. Zollhöfer, and M. Nießner. “Deferred Neural Rendering: Image Synthesis using Neural Textures”. In: *CoRR* abs/1904.12356 (2019).

[120] D. E. King. “Dlib-Ml: A Machine Learning Toolkit”. In: *J. Mach. Learn. Res.* 10 (Dec. 2009), pp. 1755–1758.

[121] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. “ImageNet: A large-scale hierarchical image database”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition* (2009), pp. 248–255.

[122] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur. “Sharpness-Aware Minimization for Efficiently Improving Generalization”. In: *CoRR* abs/2010.01412 (2020).

[123] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. “Random Erasing Data Augmentation”. In: *CoRR* abs/1708.04896 (2017).

[124] R. Müller, S. Kornblith, and G. E. Hinton. “When Does Label Smoothing Help?” In: *CoRR* abs/1906.02629 (2019).

[125] Z. Sheng, Z. Yu, X. Liu, S.-Y. Cao, Y. Liu, H.-L. Shen, and H. Zhang. “Structure Aggregation for Cross-Spectral Stereo Image Guided Denoising”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 13997–14006.

[126] Y. Mansour and R. Heckel. “Zero-Shot Noise2Noise: Efficient Image Denoising Without Any Data”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 14018–14027.

[127] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra. “Grad-CAM: Why did you say that? Visual Explanations from Deep Networks via Gradient-based Localization”. In: *CoRR* abs/1610.02391 (2016).

[128] J. Sun, X. Wang, Y. Zhang, X. Li, Q. Zhang, Y. Liu, and J. Wang. “FENeRF: Face Editing in Neural Radiance Fields”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 7672–7682.

[129] G.-S. Hsu, C.-H. Tsai, and H.-Y. Wu. “Dual-Generator Face Reenactment”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 642–650.

[130] B. Bayar and M. C. Stamm. “A Deep Learning Approach to Universal Image Manipulation Detection Using a New Convolutional Layer”. In: *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security* (2016).

[131] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. “MS-Celeb-1M: A Dataset and Benchmark for Large-Scale Face Recognition”. In: *European Conference on Computer Vision*. 2016.

[132] S. Abnar and W. Zuidema. “Quantifying Attention Flow in Transformers”. In: *Annual Meeting of the Association for Computational Linguistics*. 2020.

[133] A. Luo, R. Cai, C. Kong, Y. Ju, X. Kang, J. Huang, and A. C. K. Life. “Forgery-aware adaptive learning with vision transformer for generalized face forgery detection”. In: *IEEE Transactions on Circuits and Systems for Video Technology* (2024).

[134] J. Liang, H. Shi, and W. Deng. “Exploring disentangled content information for face forgery detection”. In: *European conference on computer vision*. Springer. 2022, pp. 128–145.

[135] B.-S. Hua, M.-K. Tran, and S.-K. Yeung. “Pointwise convolutional neural networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 984–993.

[136] J. Guan, H. Zhou, Z. Hong, E. Ding, J. Wang, C. Quan, and Y. Zhao. "Delving into sequential patches for deepfake detection". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 4517–4530.

[137] J. Deng, J. Guo, Y. Zhou, J. Yu, I. Kotsia, and S. Zafeiriou. "RetinaFace: Single-stage Dense Face Localisation in the Wild". In: *CoRR* abs/1905.00641 (2019).

[138] I. Loshchilov and F. Hutter. "Decoupled Weight Decay Regularization". In: *International Conference on Learning Representations*. 2017.

[139] Z. Yan, Y. Luo, S. Lyu, Q. Liu, and B. Wu. "Transcending Forgery Specificity with Latent Space Augmentation for Generalizable Deepfake Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2024, pp. 8984–8994.

[140] D. Nguyen, M. Astrid, E. Ghorbel, and D. Aouada. "FakeFormer: Efficient Vulnerability-Driven Transformers for Generalisable Deepfake Detection". In: *arXiv preprint arXiv:2410.21964* (2024).

[141] L. Li, J. Bao, H. Yang, D. Chen, and F. Wen. "Advancing High Fidelity Identity Swapping for Forgery Detection". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 5073–5082. DOI: 10.1109/CVPR42600.2020.00512.

[142] Y. Zhu, Q. Li, J. Wang, C. Xu, and Z. Sun. "One Shot Face Swapping on Megapixels". In: *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. June 2021, pp. 4834–4844.

[143] J. Choi, T. Kim, Y. Jeong, S. Baek, and J. Choi. "Exploiting Style Latent Flows for Generalizing Deepfake Video Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2024, pp. 1133–1143.

[144] T. DeVries and G. W. Taylor. *Improved Regularization of Convolutional Neural Networks with Cutout*. 2017.

[145] J. Choi and Y. Kim. *Colorful Cutout: Enhancing Image Data Augmentation with Curriculum Learning*. 2024.

[146] C. Gong, D. Wang, M. Li, V. Chandra, and Q. Liu. "Keepaugment: A simple information-preserving data augmentation approach". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 1055–1064.

[147] G. Song and W. Chai. "Collaborative Learning for Deep Neural Networks". In: *Neural Information Processing Systems*. 2018.

[148] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz. “mixup: Beyond Empirical Risk Minimization”. In: *CoRR* abs/1710.09412 (2017).

[149] A. Haliassos, R. Mira, S. Petridis, and M. Pantic. “Leveraging Real Talking Faces via Self-Supervision for Robust Forgery Detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 14950–14962.

[150] Y. Luo, Y. Zhang, J. Yan, and W. Liu. “Generalizing Face Forgery Detection With High-Frequency Features”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 16317–16326.

[151] K. Hara, H. Kataoka, and Y. Satoh. “Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 6546–6555.

[152] Q. Lv, Y. Li, J. Dong, S. Chen, H. Yu, H. Zhou, and S. Zhang. “DomainForensics: Exposing Face Forgery Across Domains via Bi-Directional Adaptation”. In: *IEEE Transactions on Information Forensics and Security* 19 (2024), pp. 7275–7289. DOI: 10.1109/TIFS.2024.3426317.

[153] D. Zhang, Z. Xiao, S. Li, F. Lin, J. Li, and S. Ge. “Learning natural consistency representation for face forgery video detection”. In: *European Conference on Computer Vision*. Springer. 2024, pp. 407–424.

[154] L. Lin, X. He, Y. Ju, X. Wang, F. Ding, and S. Hu. “Preserving Fairness Generalization in Deepfake Detection”. In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2024), pp. 16815–16825.

[155] A. Kumagai, T. Iwata, H. Takahashi, T. Nishiyama, and Y. Fujiwara. “AUC Maximization under Positive Distribution Shift”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang. Vol. 37. Curran Associates, Inc., 2024, pp. 36071–36096.

[156] J. Huang and C. Ling. “Using AUC and accuracy in evaluating learning algorithms”. In: *IEEE Transactions on Knowledge and Data Engineering* 17.3 (2005), pp. 299–310. DOI: 10.1109/TKDE.2005.50.

[157] N. Ueda and A. Fujino. “Partial auc maximization via nonlinear scoring functions”. In: *arXiv preprint arXiv:1806.04838* (2018).

[158] K. Kwegyir-Aggrey, M. Gerchick, M. Mohan, A. Horowitz, and S. Venkatasubramanian. “The Misuse of AUC: What High Impact Risk Assessment Gets Wrong”. In: *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency* (2023).

[159] B. Fu, Z. Cao, M. Long, and J. Wang. “Learning to detect open classes for universal domain adaptation”. In: *European conference on computer vision*. Springer. 2020, pp. 567–583.

[160] I. J. Goodfellow, J. Shlens, and C. Szegedy. *Explaining and Harnessing Adversarial Examples*. 2015.

[161] A. Kurakin, I. J. Goodfellow, and S. Bengio. “Adversarial Machine Learning at Scale”. In: *ArXiv* abs/1611.01236 (2016).

[162] X. Zhang, Q. Wang, J. Zhang, and Z. Zhong. “Adversarial AutoAugment”. In: *ArXiv* abs/1912.11188 (2019).

Appendix A

Appendices

A.1 Supplementary Materials on LAA-Net

This document provides supplementary material complementing the main chapter 3. It is structured as follows. First, the computation of the self-consistency loss and the ground truth generation of heatmaps are described. Second, more quantitative and qualitative results are provided. In particular, additional metrics are reported for both in-dataset and cross-dataset settings. Moreover, qualitative results comparing E-FPN and FPN are shown.

A.1.1 Self-Consistency Loss

To clarify the calculation of the self-consistency loss, we show Figure A.1, which illustrates the generation process of the predicted and the ground-truth, $\hat{\mathbf{C}}$ and $\mathbf{C}$, respectively. The self-consistency loss is a binary cross entropy loss between $\hat{\mathbf{C}}$ and $\mathbf{C}$.

A.1.2 Ground Truth Generation of Heatmaps

In this section, we provide more details regarding the generation of ground-truth heatmaps, described in Section 3.3.1. Firstly, a k-th vulnerable point, denoted as $\mathbf{p}^k$, is selected, as shown in Figure A.2 (i). Secondly, we measure the height and the width of the blending mask $\mathbf{B}$ at the point $\mathbf{p}^k$ shown as orange lines in Figure A.2 (ii). Using the calculated distances,

Method	Training Set		FF++ [23]				
	Real	Fake	ACC	AUC	AP	AR	mF1
Ours w/ BI [25]	✓		99.03	99.95	99.99	99.21	99.60
Ours w/ SBI [26]	✓		99.04	99.96	99.99	99.29	99.64

Table A.1 In-dataset evaluation on FF++ [23] reported by ACC, AUC, AP, AR, and mF1.

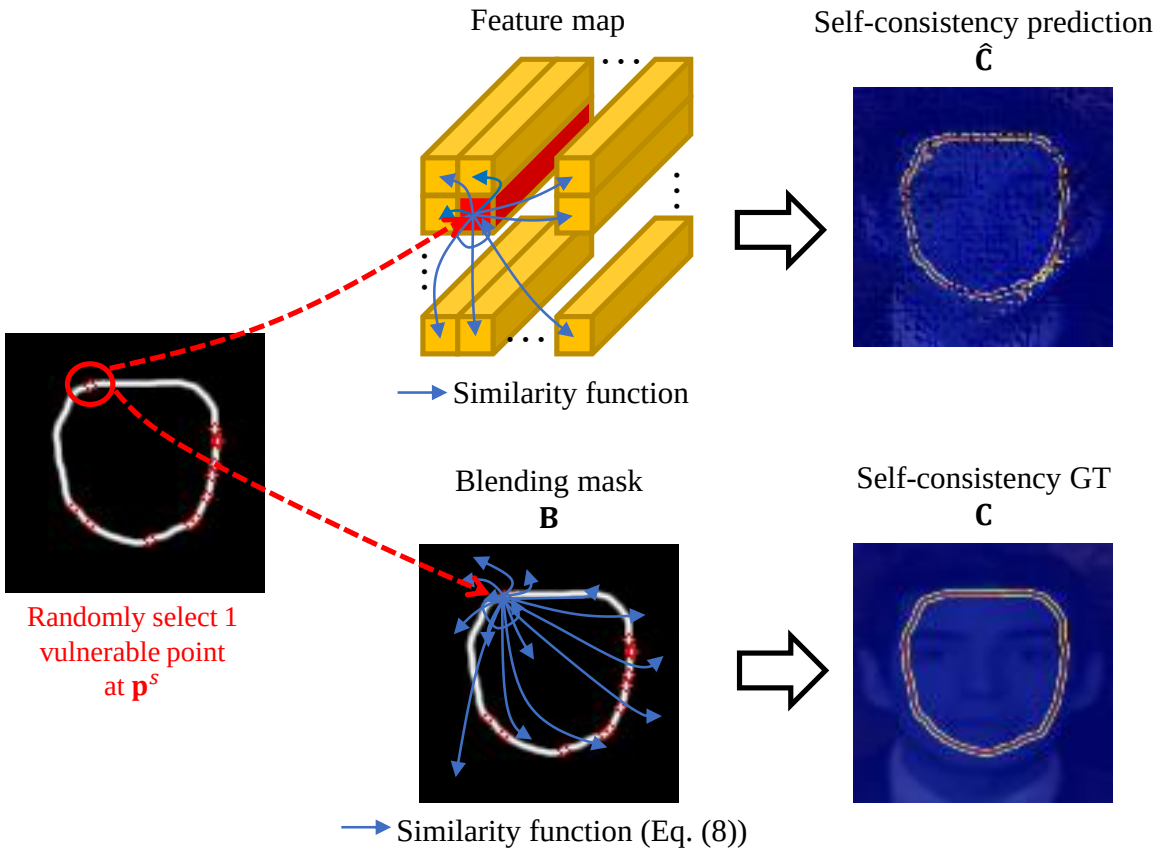


Figure A.1 In order to generate the consistency map prediction $\hat{\mathbf{C}}$ as well as the associated ground truth $\mathbf{C}$, we first randomly select a vulnerable point located at $\mathbf{p}^s$. For computing $\hat{\mathbf{C}}$, we measure the similarity between the feature at $\mathbf{p}^s$ (red block) and the features generated from every point. Namely, we use the similarity function in [33]. As for $\mathbf{C}$, we measure the consistency values between the pixel at the $\mathbf{p}^s$ and all pixels in $\mathbf{B}$, as also described in (Eq. 3.5) of the chapter.

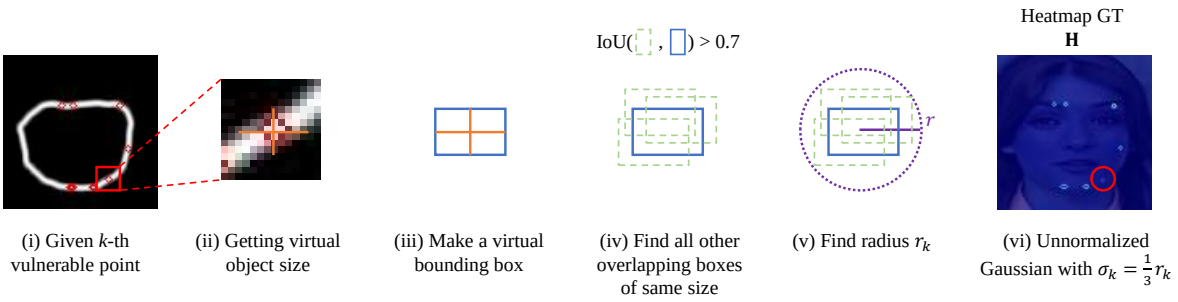


Figure A.2 The generation process of ground truth heatmaps by producing using an *Unnormalized Gaussian Distribution* given a selected vulnerable point.

a virtual bounding box is created, indicated by the blue box in Figure A.2 (iii). Then, we identify overlapping boxes, illustrated by dashed-line green boxes in Figure A.2 (iv), with the Intersection over Union (IoU) greater than a threshold ($t = 0.7$) compared to the virtual bounding box. A radius r_k (solid purple line in Figure A.2 (v)) is calculated by forming a tight circle encompassing all these boxes. Finally, an *Unnormalized Gaussian Distribution*, shown as a red circle in Figure A.2 (vi), is generated with a standard deviation $\sigma_k = \frac{1}{3}r_k$ (Eq. (3.2)) of

Method	Fake	Test set (%)															
		CDF2				DFW				DFD				DFDC			
		AUC	AP	AR	mF1	AUC	AP	AR	mF1	AUC	AP	AR	mF1	AUC	AP	AR	mF1
Xception [23]	✓	61.18	66.93	52.40	58.78	65.29	55.37	57.99	56.65	89.75	85.48	79.34	82.29	69.90	91.98	67.07	77.57
FaceXRay+BI [25]	✓	79.5	-	-	-	-	-	-	-	95.40	93.34	-	-	65.5	-	-	-
LRNet [112]	✓	53.20	-	-	-	-	-	-	-	52.29	-	-	-	-	-	-	-
LocalRL [63]	✓	78.26	-	-	-	-	-	-	-	89.24	-	-	-	76.53	-	-	-
TI^{2}Net [113]	✓	68.22	-	-	-	-	-	-	-	72.03	-	-	-	-	-	-	-
Multi-attentional [39]	✓	68.26	75.25	52.40	61.78	73.56	73.79	63.38	68.19	92.95	96.51	60.76	74.57	63.02	-	-	-
RECCE [32]	✓	70.93	70.35	59.48	64.46	68.16	54.41	56.59	55.48	98.26	79.42	69.57	74.17	-	-	-	-
SFDG [48]	✓	75.83	-	-	-	69.27	-	-	-	88.00	-	-	-	73.63	-	-	-
EIC+IIE [114]	✓	83.80	-	-	-	-	-	-	-	93.92	-	-	-	81.23	-	-	-
AltFreezing [56]	✓	89.50	-	-	-	-	-	-	-	98.50	-	-	-	-	-	-	-
CADDM [27]	✓	<u>93.88</u>	91.12	77.00	83.46	<u>74.48</u>	<u>75.23</u>	<u>65.26</u>	<u>69.89</u>	99.03	<u>99.59</u>	82.17	90.04	-	-	-	-
UCF [44]	✓	82.4	-	-	-	-	-	-	-	94.5	-	-	-	80.5	-	-	-
Controllable GS [115]	✓	84.97	-	-	-	-	-	-	-	-	-	-	-	81.65	-	-	-
PCL+I2G [33]		90.03	-	-	-	-	-	-	-	99.07	-	-	-	74.27	-	-	-
SBI [26]		93.18	85.16	<u>82.68</u>	<u>83.90</u>	67.47	55.87	55.82	55.85	97.56	92.79	<u>89.49</u>	91.11	86.15	93.24	<u>71.58</u>	<u>80.99</u>
AUNet [65]		92.77	-	-	-	-	-	-	-	<u>99.22</u>	-	-	-	<u>86.16</u>	-	-	-
Ours (w/ BI)		86.28	<u>91.93</u>	50.01	64.78	57.13	56.89	50.12	53.29	**99.51**	**99.80**	**95.47**	**97.59**	69.69	<u>93.67</u>	50.12	65.30
Ours (w/ SBI)		**95.40**	**97.64**	**87.71**	**92.41**	**80.03**	**81.08**	**65.66**	**72.56**	98.43	99.40	88.55	<u>93.64</u>	**86.94**	**97.70**	**73.37**	**83.81**

Table A.2 Cross-dataset evaluation in terms of AUC, AP, AR, and mF1 (%) on CDF2 [80], DFW [82], DFD [83], and DFDC [81]. **Bold** and <u>underlined</u> highlight the best and the second-best performance, respectively. ✓symbol is used to depict methods that utilized both Real data and Fake data for training.

the main chapter). The steps are repeated for every vulnerable point $k \in [\![1, \text{card}(\mathcal{P})]\!]$. The final $\mathbf{H}$ is the superimposition of all g_{ij}^k.

A.1.3 Additional Results

In addition to AUC, we provide results using additional metrics, namely, Average Precision (AP), Average Recall (AR), Accuracy (ACC), and mean F1-score (mF1).

Table A.1 and Table A.2 report the results under the in-dataset and the cross-dataset settings, respectively. Overall, it can be seen that LAA-Net achieves better performances than other state-of-the-art methods.

A.1.4 Qualitative Results: E-FPN versus FPN

A qualitative comparison between the proposed E-FPN and the traditional FPN with different fusion settings is reported in Figure A.3. Using EfficientNet-B4 [45] (EFNB4) as our backbone, the $\mathbf{F}^{(6)}$ refers to the features extracted from the last convolution block in the backbone. In other words, this means that no FPN design is integrated. By gradually aggregating features from lower to higher resolution layers, we can observe the improvement of the forgery localization ability for both E-FPN and FPN. More notably, E-FPN produces more precise activations on the blending boundaries as compared to FPN. This can be explained by the fact that the E-FPN integrates a filtering mechanism for learning less noise. In contrast, FPN seems to consider regions outside the blending boundary, which results in lower performance as previously shown in Table 3.4 - Section 3.4.4 of the main chapter 3.

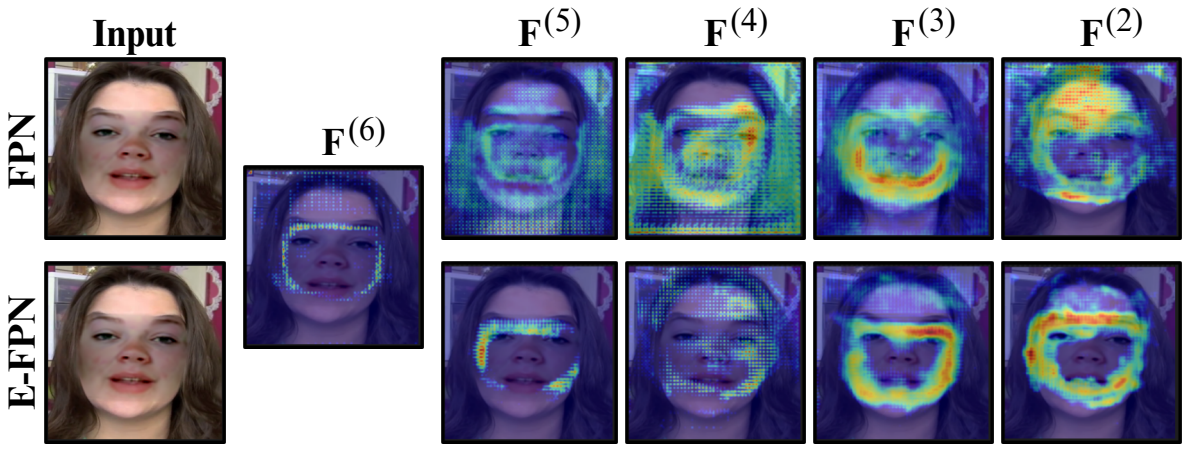


Figure A.3 Feature visualization by GradCAM [127] between *E-FPN* and *FPN* with different integration of multi-scale layers. It shows that E-FPN can focus better on artifacts as compared to FPN. The setup details are provided in Table 3.4 as shown in the chapter.

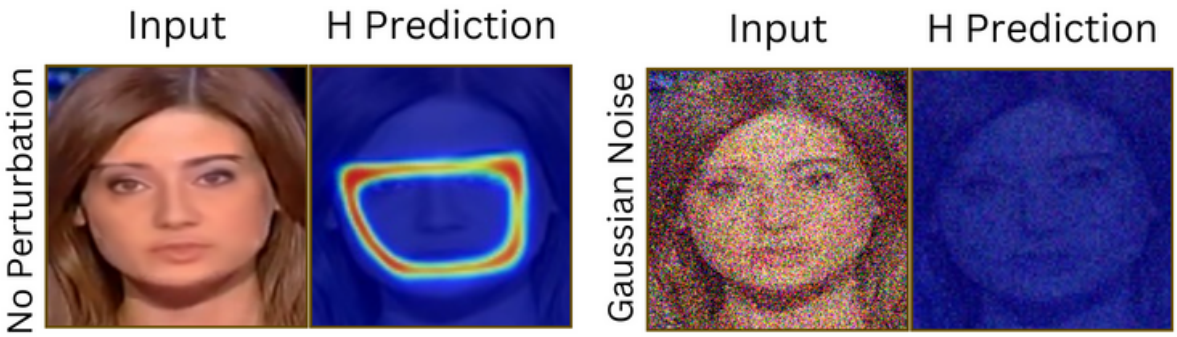


Figure A.4 Detection of vulnerable points w/o and w/ Gaussian noise.

A.1.5 Qualitative Results: Gaussian Noise

In Table 3.2 of the main chapter, the performance of LAA-Net declined significantly when encountering Gaussian Noise perturbations. One possible reason is that the introduction of noise elevates the difficulty of detecting the vulnerable points. To confirm that, we report the inference of the heatmap before and after applying a Gaussian Noise on a facial image in Figure A.4. As it can be observed, the detection of vulnerable points is highly impacted with the introduction of a Gaussian noise.

A.1.6 Robustness to Compression

To assess the robustness of LAA-Net to compression, we test LAA-Net on the c23 version of FF++, and the overall AUC is equal to 89.30%.

A.2 Supplementary Materials on LAA-Former

A.2.1 More Details regarding the Datasets

Datasets. FF++ contains 1000 original videos and 4000 deepfake videos. The types of deepfakes include Deepfakes (DF) [128], Face2Face (F2F) [117], FaceSwap (FS) [118], and NeuralTextures (NT) [119]. Moreover, while CDF [80] and DFD [83] represent datasets including high-quality deepfakes [80, 62], DFW [82] is recognized as one of the most challenging benchmarks. DFW is collected entirely from the internet without any prior knowledge regarding the used manipulation techniques. Additionally, it includes images with occlusions, illumination changes, different head poses, and diverse backgrounds. Meanwhile, DFDC [88], including its preview version, DFDCP [81], comprises more than $128,000$ videos and is therefore one of the largest and most challenging datasets in the field of deepfake detection.

Data Processing. The RetinaFace [137] algorithm is employed to extract facial bounding boxes. These boxes are enlarged by a factor of 1.25 and then resized to a specific resolution (H, W) to match the different backbone configurations. To generate pseudo-fakes, the two images are first aligned. For this purpose, the Dlib [120] library is utilized to extract 68 and 81 facial landmarks for the two considered blending synthesis algorithms, BI [25] and SBI [26], respectively. Finally, the processed pseudo-fakes are generated for each training iteration.

A.2.2 Architecture Details

We describe in detail the hyperparameters of the two considered LAA-Former variants as follows:

- LAA-Former-S: $H = W = 112$, $P = 8$, $L = 12$, $D = 384$, MLP size $= 1536$, No. Heads $= 6$, #Params$=$ 23M, FLOPs$=$ 8.9G.
- LAA-Former-B: $H = W = 224$, $P = 16$, $L = 12$, $D = 768$, MLP size $= 3072$, No. Heads $= 12$, #Params$=$ 91M, FLOPs$=$ 35.8G.

where the *MLP size* represents the dimension of hidden layers in MLP, the *No. Heads* denotes the number of heads in MHSA, *#Params* is the number of parameters, and *FLOPs* represents the computational cost in terms of floating point operations per second. For LAA-Swin architecture, we adopt these two backbone variants from Swin [85], namely:

- LAA-Swin-S: $H = W = 224$, $P = 4$, $M = 7$, $d = 32$, $\alpha = 4$, $C_h = 96$, Layer Numbers = {2, 2, 18, 2}, No. Heads = {3, 6, 12, 24}, #Params$=$ 55M, FLOPs$=$ 6.5G.

- LAA-Swin-B: $H = W = 224$, $P = 4$, $M = 7$, $d = 32$, $\alpha = 4$, $C_h = 128$, Layer Numbers = {2, 2, 18, 2}, No. Heads = {4, 8, 16, 32}, #Params= 91M, FLOPs= 11.5G.

where M is the window size, d is the query dimension of each head, the expansion layer of each MLP is α, and C_h denotes the channel number in the hidden layers during the first stage.

A.2.3 More Details regarding the training Setup in Section 4.3

In this section, we provide additional details on the experimental settings used in Section 4.3 of chapter 4.

Specifically, in Figure 4.3b, we train vanilla ViT [68], Swin [85], and Xception [46] with the SBI data synthesis algorithm [26], using the same training settings as in [62, 26]. The detection performance of CNN-based methods across different ranges of Mask-SSIM [104] on CDF2 [80], *i.e.*, LAA-Net [62], CADDM [27], and SBI [26], is directly taken from [62], while the results of specialized transformer-based methods, *i.e.*, ForensicsAdapter [28] and FaViT [133], are reproduced using the officially released pretrained weights. Note that in SBI [26], the authors focus solely on data synthesis; therefore, the detection architecture used is a standard CNN (EfficientNet-B4 (EFNB4) [45]).

Secondly, in Figure 4.3a, all models, including CNNs and ViT variants, are trained on FF++ [23] with both real and fake data for 50 epochs. Following the conventional splitting in [23], we uniformly extract 128 and 32 frames from each video for training and validation, respectively, resulting in a total of 460800 and 22400 frames for the corresponding tasks. All models are initialized with ImageNet-pretrained weights [121]. We employ different optimizers, with Adam commonly used for CNNs [31, 64, 27, 25, 39, 48] and AdamW for ViTs [85, 84, 69, 86]. The learning rate is initially set to 10^{-4} and linearly decays to 0 over the training period. All experiments are carried out on a NVIDIA A100 GPU.

Function	Test set AUC (%)						
	CDF	DFD	DFDCP	DFDC	DFW	DiffSwap	Avg.
MinMax	91.6	<u>95.9</u>	<u>88.1</u>	71.1	70.1	<u>92.1</u>	84.8
Unnormalized 3D Gaussian	**92.5**	91.8	86.4	<u>72.4</u>	**76.3**	91.7	<u>85.2</u>
MeanStd	<u>92.4</u>	**98.5**	**90.0**	**74.6**	<u>74.2</u>	**96.9**	**87.8**

Table A.3 Comparison of different normalization functions. We consider three normalization functions, i.e., Standardization (MeanStd), MinMax, and Unnormalized 3D Gaussian. Among these, Standardization gives the best overall performance.

A.3 Supplementary Materials on FakeSTormer

A.3.1 Details of the Ground-Truth Derivative

Calculation formula. To generate the ground truth data for the temporal branch h, we compute the derivative on $\tilde{\mathbf{B}} = (\tilde{\mathbf{B}}(t))_{t\in[[1,T]]}$ with respect to the temporal dimension. Specifically, we calculate the absolute value of the difference between two consecutive patch-level vulnerability values $\tilde{\mathbf{B}}(t)$ and $\tilde{\mathbf{B}}(t-1)$ with $t \geq 2$ such that,

$$\mathbf{D}(t) = \left|\tilde{\mathbf{B}}(t) - \tilde{\mathbf{B}}(t-1)\right|. \tag{A.1}$$

The process is iterated for every pair of consecutive frames of $(\tilde{\mathbf{B}}(t))_{t\in[[1,T]]}$ to obtain a derivative matrix $\mathbf{D} \in \mathbb{R}^{T\times\sqrt{N}\times\sqrt{N}}$. For $t = 1$, we insert a matrix $\mathbf{0}$, i.e., $\mathbf{D}(1) = \mathbf{0}$, indicating no temporal change at the first frame.

Normalization functions. Employing a normalization function is important for stabilizing the training of our temporal branch h. Therefore, we consider three different normalization functions including Standardization (MeanStd), MinMax, and Unnormalized 3D Gaussian. Specifically, for the Standardization and MinMax, we respectively compute the std($\mathbf{D}$)-mean($\mathbf{D}$), and min($\mathbf{D}$)-max($\mathbf{D}$), while we follow the work of [140] to adapt an unnormalized Gaussian map from 2D to 3D for normalization. We report in Table A.3 cross-evaluation results on six datasets [80, 83, 81, 88, 82, 13] with the use of the three investigated functions, using a model trained on FF++ [23]. It can be noted that the model is robust to various types of normalization functions with the best performance recorded for the Standardization approach.

A.3.2 SBV: Pseudo-code and Visual Samples

Algorithm. To enhance the clarity and reproducibility of the SBV generation process, we provide the overall algorithm in the form of pseudo-code, as detailed in Algorithm 1.

Visual Samples. To visually demonstrate the benefits of the Consistent Synthesized Pa-

Algorithm 1: Pseudo-code for SBV Generation

Input: Real video $\mathbf{X}_i^r \in \mathcal{V}^r$ of size (C, T, H, W), facial landmarks $\mathbf{L}_i \triangleq \cup_{t=1}^{T}\{\mathbf{l}_{ij}(t)\}_{1\leq j\leq n}$ of size $(T, n, 2)$, a distance $\bar{d}$, a threshold τ

Output: Self-blended video $\mathbf{X}_i^{\tilde{r}} \in \mathcal{V}^{\tilde{r}}$ of size (C, T, H, W), blending mask $\mathbf{M}_i$ of size (T, H, W)

Initialize $\theta^{(sbi)}$ as an empty dictionary
Initialize $\mathbf{X}_i^{\tilde{r}}$ as an empty array
Initialize $\mathbf{M}_i$ as an empty array
for $j = 1$ **to** T **do**
    if $j = 1$ **then**
        $\mathbf{X}_i^{\tilde{r}}(t_0), \mathbf{M}_i(t_0), \{\theta^{(c)}, \theta^{(m)}, \theta^{(b)}, \ldots\} \leftarrow$ SBI($\mathbf{X}_i^r(t_0), \mathbf{l}_i(t_0)$)
        $\mathbf{X}_i^{\tilde{r}} \leftarrow \mathbf{X}_i^{\tilde{r}} \cup \{\mathbf{X}_i^{\tilde{r}}(t_0)\}$
        $\mathbf{M}_i \leftarrow \mathbf{M}_i \cup \{\mathbf{M}_i(t_0)\}$
        $\theta^{(sbi)} \leftarrow \{\theta^{(c)}, \theta^{(m)}, \theta^{(b)}, \ldots\}$
    end
    else
        $\mathbf{l}_i(t) \leftarrow$ LandmarkInterpolation($\mathbf{l}_i(t), \mathbf{l}_i(t-1), \bar{d}, \tau$)
        $\mathbf{X}_i^{\tilde{r}}(t), \mathbf{M}_i(t), \_ \leftarrow$ SBI($\mathbf{X}_i^r(t), \mathbf{l}_i(t), \theta^{(sbi)}$)
        $\mathbf{X}_i^{\tilde{r}} \leftarrow \mathbf{X}_i^{\tilde{r}} \cup \{\mathbf{X}_i^{\tilde{r}}(t)\}$
        $\mathbf{M}_i \leftarrow \mathbf{M}_i \cup \{\mathbf{M}_i(t)\}$
    end
end
return $\mathbf{X}_i^{\tilde{r}}, \mathbf{M}_i$

rameters (CSP) and the Landmark Interpolation (LI) module (Section 5.3.1 in chapter 5) in generating high-quality pseudo-fake videos, we show some SBV samples, their blending boundaries, original landmarks, and those modified by the proposed modules in Figure A.5. In the top part of the figure, we compare data generated using only CSP to data generated with both CSP and the Landmark Interpolation module. We observe that the Landmark Interpolation module ensures smooth transitions of facial landmarks between consecutive frames ($t \rightarrow t+1$). In the bottom part of the figure, we compare data generated with only CSP to data generated without any of the proposed SBV components. We observe significant variations in the manipulated facial areas when CSP is omitted. Therefore, the proposed CSP and Landmark Interpolation module effectively enhances the temporal coherence of the generated SBV.

A.3.3 Impact of SBV

To verify the advantage of using SBV for improving the generalization of detectors, we conduct several experiments using different binary classifiers trained either with SBV or with one of the four types of forgeries forming FF++ [23] (DF [116], F2F [117], FS [118], NT [119]). For a fair comparison, a widely-used CNN-based Resnet3D [151] and a Transformer-based

Figure A.5 Illustration of the facial landmarks, the generated SBV, and the blending boundaries with and without applying the Consistent Synthesized Parameters (CSP) module and the Landmark Interpolation (LI) module. The lack of applied CSP and LI indicates simply stacked SBIs (BottomRight).

Method	Pretrain	Training set					Test set AUC (%)						Avg.
		Real	DF	FS	F2F	NT	FF++	CDF	DFD	DFDCP	DFDC	DFW	
ResNet3D [151]	×	✓	✓	×	×	×	72.5	58.5	51.3	53.4	59.4	65.0	60.0
TimeSFormer [89]	×	✓	✓	×	×	×	65.4	59.3	66.1	53.5	61.4	57.5	60.5
ResNet3D [151]	×	✓	×	✓	×	×	70.6	61.1	50.6	59.2	55.8	51.5	58.1
TimeSFormer [89]	×	✓	×	✓	×	×	76.4	51.7	43.7	44.6	54.5	43.9	52.5
ResNet3D [151]	×	✓	×	×	✓	×	78.0	63.8	54.5	63.4	55.7	50.1	60.9
TimeSFormer [89]	×	✓	×	×	✓	×	81.1	64.4	60.1	64.5	52.0	50.5	62.1
ResNet3D [151]	×	✓	×	×	×	✓	72.7	63.7	75.6	69.1	59.6	62.6	67.2
TimeSFormer [89]	×	✓	×	×	×	✓	75.5	65.8	84.7	70.3	62.7	65.5	70.8
ResNet3D [151] + SBV	×	✓	×	×	×	×	90.2	85.9	85.0	82.8	66.4	67.5	79.6(↑12.4)
TimeSFormer [89] + SBV	×	✓	×	×	×	×	**94.7**	**89.5**	**95.6**	**88.6**	**72.5**	**70.9**	**85.3**(↑14.5)

Table A.4 Cross-dataset generalization. Performance comparison in terms of AUC (%) on multiple datasets of different binary classification models [89, 151] trained using our video synthesis (SBV) and normal fake data [23]. All models are trained on FF++(c23) [23] from ***Scratch (S)*** and are tested on other datasets [80, 83, 81, 88, 82]. Gray indicates the use of normal fake data for training. **Bold** and underline highlight the best and the second-best performance, respectively.

Method	Real	Fake	Contrast	Saturation	Gaussian Blur	Gaussian Noise	JPEG Compression	Block Wise	Avg
DSP-FWA [43]	✓	✓	80.7	79.6	67.3	61.8	68.0	76.6	72.3
FaceXray [25]	✓	✓	88.9	96.0	70.0	58.0	62.2	94.7	78.3
SBI [26]	✓	×	92.3	92.0	72.7	62.2	79.1	92.2	81.7
LAA-Net [62]	✓	×	95.0	**97.0**	73.2	57.5	75.2	94.9	82.1
FakeSTormer	✓	×	**97.6**	96.3	**81.6**	**65.4**	**92.9**	**96.8**	**88.4**

Table A.5 Robustness to unseen perturbations. Average AUC scores (%) across all levels for each degradation type.

TimeSformer [89] are employed. We note that both selected models are trained from *Scratch (S)* without pretrained initialization. Table A.4 presents the generalization performance in terms of AUC (%) on five datasets [80, 83, 81, 88, 82] respectively when trained with different manipulation methods from FF+. Notably, training with SBV significantly increases the overall generalizability capability of binary models as compared to those trained on using one specific manipulation. This indicates the importance of highly realistic, naturally consistent generated pseudo-fake videos.

A.3.4 Robustness to Unseen Perturbations

In the chapter 5, we report in Figure 5.3 the "Average" performance under different corruptions. This section complements this experiment by reporting the mean performance across different severity levels for each degradation type, as detailed in Table A.5. Except for a slight decrease in effectiveness under "Change Saturation" compared to LAA-Net [62], FakeSTormer is generally more robust to the unseen perturbations as compared to other augmented-based methods [62, 26, 25, 43].

No. shots	Test set AUC (%)					
	CDF	DFD	DFDCP	DFDC	DFW	Avg.
1	92.35	98.47	90.02	74.56	74.19	85.92
2	92.34	<u>98.51</u>	<u>90.11</u>	<u>74.60</u>	74.25	85.96(↑0.04)
3	**92.38**	**98.52**	**90.13**	**74.61**	<u>74.28</u>	<u>85.98</u>(↑0.06)
4	**92.38**	**98.52**	**90.13**	**74.61**	**74.30**	**85.99**(↑0.07)
5	<u>92.36</u>	**98.52**	**90.13**	**74.61**	**74.30**	<u>85.98</u>(↑0.06)

Table A.6 Multi-shot inferences. AUC (%) comparison of our model using different numbers of inference shots in the cross-dataset setup. The AUC slightly increases with a higher number of shots.

A.3.5 Multi-shot Inferences

Models can sometimes be overconfident in their predictions, which negatively impacts the generalizability aspect [148, 124]. To address this issue, we explore the possibility of regularizing the input during testing. Specifically, we propose multi-shot inference, leveraging Vulnerability-Driven Cutout Augmentation by utilizing the temporal head output $\tilde{\mathbf{D}}$. We use $\tilde{\mathbf{D}}$ because the most significant temporal changes in vulnerable areas over time, from $t \rightarrow (t+1)$, are likely to occur at the spatial locations corresponding to the highest values of $\bar{\mathbf{B}}$ (Eq. (5.2) in the chapter 5).

In particular, given a test video, after the first shot of inference, the prediction map $\tilde{\mathbf{D}}$ can be leveraged to generate a new masked video through the proposed Cutout augmentation for the second inference shot. Specifically, we select $\tilde{\mathbf{D}}$ at $t = 2$ (capturing the temporal transition from the first $\rightarrow$ the second frame) to define the set $\mathcal{P}$ (Section 5.3.1 of the chapter 5) for determining Cutout positions. This iterative process can be repeated for multiple inference shots.

Table A.6 presents the cross-dataset evaluation results with five shots of inference on five unseen datasets [80, 82, 83, 81, 88] using the model trained on FF++ [23]. The results indicate a gradual improvement in generalization performance after each iteration. This suggests that the prediction outputs are not only interpretable but also can be used potentially to enhance the model performance.

A.3.6 Visualization of Auxiliary Branches' Outputs

In addition to the probability output of the standard classification branch, FakeSTormer can provide more valuable insights from our auxiliary branches that might be conductive to prediction's post-analyses. Specifically, the spatial and temporal branches output the intensity of the spatial artifacts encoded in each video frame and the vulnerability change over time, respectively. The spatial branch provides frame-level scores, while the temporal branch offers

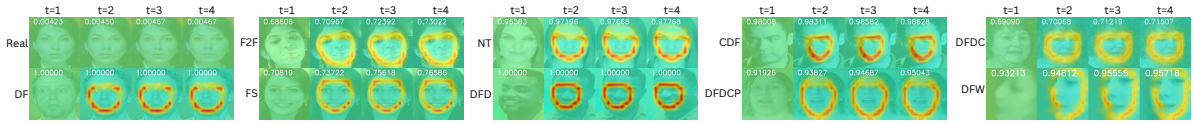


Figure A.6 Visualization of Auxiliary Branches' Outputs. We visualize the additional auxiliary spatial and temporal branches' outputs on different unseen datasets. As shown, the number on each frame denotes the output of the spatial branch g, while the heatmap visualizes the output of the temporal branch h.

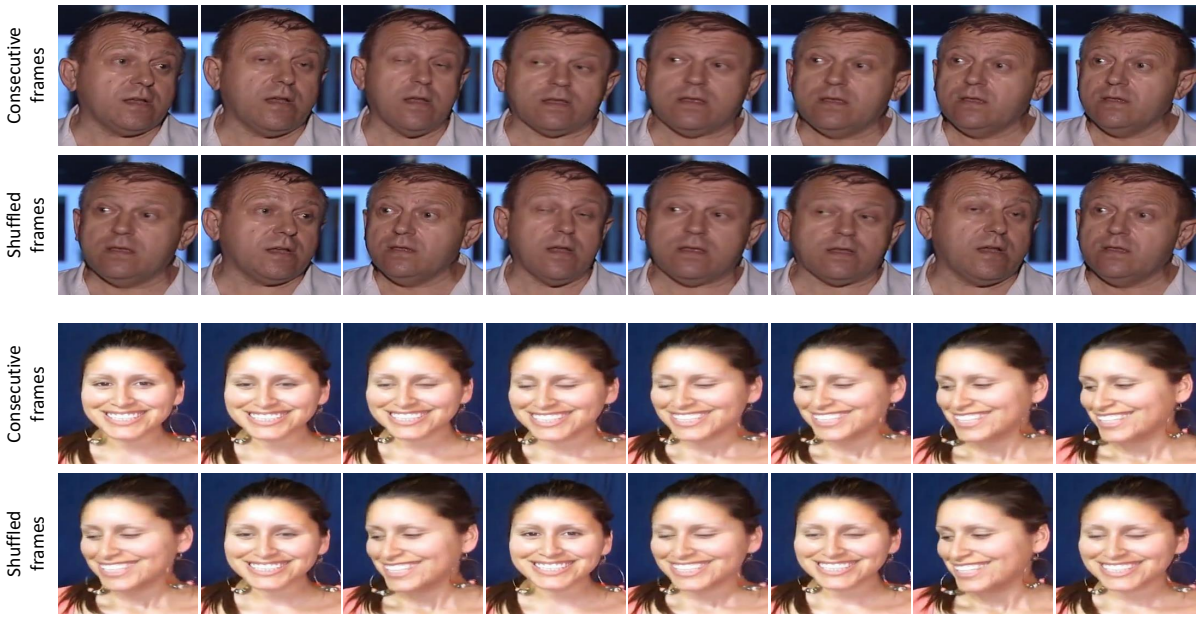


Figure A.7 Shuffled frames can produce obvious temporal inconsistencies.

more fine-grained insights. As shown in Figure A.6, the spatial outputs (denoted by the numbers in each frame) denote high values for fake data and low values for real data. For the temporal outputs, the heatmaps show the change in vulnerability between the instant frame t and the previous one. It can be observed that it primarily focuses around the blending boundaries. We note that the change between $t-1$ and t is visualized at the t^{th} frame; hence, there is a *blank* heatmap at the 1^{st} frame.

A.3.7 STC [58]: Shuffled Frames can produce obvious Temporal Inconsistencies

We propose SBV to generate subtler artifacts for pseudo-fakes compared to the STC approach used in [58]. We believe that STC may produce obvious (low-quality) temporal artifacts, as it shuffles frames in the temporal domain, leading to significant inconsistencies. Figure A.7 illustrates how shuffling creates noticeable discrepancies between frames. In contrast, our SBV leverages consecutive frames to produce subtler temporal artifacts while simulating these artifacts in a different manner (as detailed in Section 5.3.1 of the chapter 5).

A.3.8 Details on the Datasets

Datasets. For our experiments, we select datasets that haven typically used as benchmarks in previous works [56, 52, 36, 53, 143, 44, 48, 32]. For both training and validation, we employ **FaceForensics++**(FF++) [23], which consists of 1,000 real videos and 4,000 fake videos generated using four manipulation methods: (Deepfakes (DF) [116], FaceSwap (FS) [118], Face2Face (F2F) [117], and NeuralTextures (NT) [119]). It can be noted that, for training, we use only the real videos and generate pseudo-fake data using our synthesized method, SBV. By default, the c23 version of FF++ is adopted, following the recent literature [52, 56, 53, 36, 143].

For further validation, we also evaluate on the following datasets: (1) **Celeb-DFv2** (CDF) [80], a well-known benchmark with high-quality deepfakes; (2) **DeepfakeDetection** (DFD) [83], which includes 3,000 forged videos featuring 28 actors in various scenes; (3) **Deepfake Detection Challenge Preview** (DFDCP) [81] and (4) **Deepfake Detection Challenge** (DFDC) [88], a large-scale dataset containing numerous distorted videos with issues such as compression and noise; (5) **WildDeepfake** (DFW) [82], a dataset fully sourced from the internet, without prior knowledge of manipulation methods; (6) **DiffSwap** generated in the similar protocol as in LFGDIN [59] by using a recent diffusion-based swapping method [13] on 250 real videos selected from CDF [80]; and (7) **DF40** [14], a highly diverse and large-scale dataset comprising 40 distinct deepfake techniques, enables more comprehensive evaluations for the next generation of deepfake detection.

Data Pre-processing. Following the splitting convention [23], we extract 256, 32, and 32 consecutive frames for training, validation, and testing, respectively. Facial regions are cropped using Face-RetinaNet [137]. These bounding boxes are conservatively enlarged by a factor of 1.25 around the center of the face and then resized to a fixed resolution of 224×224. Additionally, we store 81 facial landmarks for each frame, extracted using Dlib [120]. Finally, the preserved landmark keypoints are utilized to dynamically generate pseudo-fakes during each training iteration.

A.3.9 Revisited TimeSformer: Implementation Details

We choose TimeSformer [89] as our feature extractor given its ability to effectively capture separate long-range temporal information and spatial features. First, given a video $\mathbf{X} \in \mathbb{R}^{C \times T \times H \times W}$, its frames in each time step are split into N number of non-overlapping patches of size $P \times P$, i.e., $N = \frac{H \times W}{P^2}$. Each patch is flatten as $\mathbf{x}_{(t,p)} \in \mathbb{R}^{C.P^2}$, and is then linearly mapped into D-dimensional embedding vector $\mathbf{z}^0_{(t,p)} \in \mathbb{R}^D$ by means of a learnable matrix $E \in \mathbb{R}^{D \times C.P^2}$ where $t = [[1, T]]$ indexes temporal positions, and $p = [[1, N]]$ indexes

spatial positions. The process results in an input patch embedding matrix $\mathbf{Z}^0 \in \mathbb{R}^{T \times N \times D}$.

In TimeSformer, a global class token $\mathbf{z}_{cls}$ attends to all patches and then is used for classification. This mechanism implicitly captures mixed spatial-temporal features at the same time, which might lead to overfitting on a specific type of domain artifacts [52, 56]. We revisit it slightly in order to decouple the spatial and temporal information by considering two sorts of additional tokens (one spatial and one temporal).

For that purpose, we attach in each dimension of $\mathbf{Z}^0$, a spatial token $\mathbf{z}_s^0 \in \mathbb{R}^D$ and a temporal token $\mathbf{z}_t^0 \in \mathbb{R}^D$, respectively. These tokens will independently interact only with patch embeddings belonging to their dimension axis by leveraging the decomposed SA [89]. This mechanism not only facilitates the disentanglement learning process of spatio-temporal features but is also beneficial to optimize the computational complexity of $\mathcal{O}(T^2 + N^2)$ as compared to $\mathcal{O}(T^2 \cdot N^2)$ in vanilla SA. Those tokens will then be fed into L ($L = 12$ as default) transformer encoder blocks in which each block contains a multi-head temporal SA (TSA), a multi-head spatial SA (SSA), LayerNorm (LN), and a multi-layer perception (MLP). Note that, for the sake of matrix compatibility, a placeholder embedding $\mathbf{z}_{(0,0)}^0$ is attached. Formally, the feature extraction process can be summarized as follows,

$$[\mathbf{Z}^L, \mathbf{z}_s^L, \mathbf{z}_t^L] = \Phi(\mathbf{X}), \tag{A.2}$$

where $\mathbf{Z}^L$ is the final patch embedding matrix, $\mathbf{z}_s^L$ the resulting set of spatial tokens, and $\mathbf{z}_t^L$ the resulting set of temporal tokens that will be respectively sent to the temporal head h, the spatial head g, and the classification head f. Our overall framework is illustrated in Figure 5.2-I of the chapter 5.

A.4 Supplementary Materials on Cross-AUC

Our technical appendix is organized as follows: (1) detailed descriptions of the seven deepfake detection models used in our experiments; (2) dataset information and selection criteria; (3) definitions of evaluation metrics beyond AUC and polarization; and (4) additional cross-dataset results, ablation studies, and extended analyses that complement Section 6.4 of chapter 6.

A.4.1 More Details of Deepfake Detectors

In this section, we describe the seven state-of-the-art (SOTA) deepfake detection models [46, 31, 32, 26, 27, 62, 28] evaluated under our proposed protocol (see Section 6.4.1 in the chapter 6).

XceptionNet [1]**:** In this work, the well-known benchmark FF++ [23] is introduced. The Xception architecture proposed in [46] was trained in an end-to-end manner using both real and fake data from FF++ [23].

SLADD [2]**:** In [31], the authors introduce an approach called SLADD [31] that leverages adversarial training [160, 161, 162]. In particular, it relies on two components: a generator and a discriminator (detector). While the generator attempts to synthesize facial images based on splicing operations, the detector aims to discriminate between fake and real faces. SLADD is trained using the generated pseudo-fakes as well as real and deepfake images from FF++. This approach can be seen as a hybrid method since it simultaneously takes advantage of a data synthesis strategy as well as a multi-task learning framework. This approach is claimed to be generic, given the high AUC achieved under the cross-dataset protocol.

RECCE [3]**:** This method, referred to as RECCE [32], is based on a reconstruction-classification schema. While the reconstruction learning over real images allows extracting forgery-aware features, the classification learning mines the main discrepancy between real and fake images. Both real and fake data from FF++ [23] are used during training. This multi-task learning framework is shown to achieve competitive AUC performance under the cross-dataset setting.

SBI [4]**:** This approach, called SBI [26], is based on a novel data synthesis. This data synthesis produces pseudo-fakes by blending source and target images from a single pristine image. The main idea is that such an augmentation generates pseudo-fake images with very subtle artifacts, thereby pushing the classifier to learn more generic representations. Only real data

[1] https://github.com/ondyari/FaceForensics
[2] https://github.com/liangchen527/SLADD
[3] https://github.com/VISION-SJTU/RECCE
[4] https://github.com/mapooon/SelfBlendedImages

from FF++ [23] and pseudo-fakes are used for training a binary classifier. Similar to [32], an impressive AUC is registered for the cross-dataset setting.

CADDM [5]**:** The proposed method, ID-unaware Deepfake Detection Model, aims to improve the generalization of deepfake detectors by addressing the issue of Implicit Identity Leakage (IIL), the unintended reliance on identity-specific features. To mitigate this, the authors introduce an Artifact Detection Module (ADM) that encourages the model to focus on local visual artifacts rather than global identity features. ADM uses a multi-scale anchor-based detection approach to identify regions in an image that may contain manipulation traces, helping the model distinguish between real and fake content at a finer, more localized level. To support ADM training, they design a Multi-scale Facial Swap (MFS) technique that generates fake images with labeled artifact areas by blending regions of source and fake images using different window sizes and blending strategies. This synthetic data provides ground-truth artifact locations, allowing the model to learn more generalized features. By shifting attention away from identity and toward manipulation cues, this method significantly improves cross-dataset performance and robustness to unseen forgery techniques.

LAA-Net [6]**:** LAA-Net (Localized Artifact Attention Network) is a fine-grained deepfake detection framework designed to effectively detect high-quality and unseen manipulations by focusing on localized visual inconsistencies. At its core, LAA-Net introduces an explicit attention mechanism within a multi-task learning framework composed of three parallel branches: a binary classification branch to distinguish real from fake images, a heatmap branch to localize regions with potential blending artifacts (termed "vulnerable points"), and a self-consistency branch that evaluates similarity between vulnerable pixels and their surroundings. These branches are trained using pseudo-fake data generated through blending-based synthesis from real images, eliminating the need for large-scale fake datasets. Additionally, LAA-Net integrates an Enhanced Feature Pyramid Network (E-FPN) to capture and preserve discriminative low-level features across multiple scales while minimizing redundancy. By combining precise attention guidance with robust feature representation, LAA-Net significantly improves the generalizability and accuracy of deepfake detection, even under challenging cross-dataset and quality-agnostic scenarios.

ForensicAdapter [7]**:** Forensics Adapter is a lightweight and task-specific adapter network designed to transform CLIP [74] into a powerful and generalizable face forgery detector. Unlike previous CLIP-based approaches that treat CLIP as a frozen feature extractor, this method introduces a dedicated adapter module that operates in parallel with CLIP to detect subtle

[5] https://github.com/megvii-research/CADDM
[6] https://github.com/10Ring/LAA-Net
[7] https://github.com/OUC-VAS/ForensicsAdapter

Dataset	Venue	Real videos	Fake videos	FS	FR	EFS	FE	Methods	Perturbs	Unknown
FF++ [23]	ICCV'19	1,000	4,000	2	2	-	-	4	2	×
DFD [83]	None	363	3,068	5	-	-	-	5	-	×
DFDCP [81]	ArXiv'19	1,131	4,113	-	-	-	-	2	3	×
CDF [80]	CVPR'20	590	5,639	1	-	-	-	1	-	×
DFW [82]	ACMMM'20	3,805	3,509	-	-	-	-	-	-	✓
DFDC [88]	ArXiv'20	23,564	104,500	5	1	2	-	8	19	×
DF40 [14]	NeurIPS'24	1428	0.1M+	10	13	12	5	40	6	×

Table A.7 Overview of datasets. The number of "Methods" is categorized into four subsets: Face-Swapping (FS), Face-Reenactment (FR), Entire Face Synthesis (EFS), and Face Editing (FE). "Perturbs" denotes the number of perturbations applied, while "Unknown" indicates whether the source contains prior knowledge of manipulations.

manipulation traces, especially blending boundaries characteristic of forged faces. The adapter is trained using a combination of objectives: masked blending boundary detection, patch-wise contrastive learning, and sample-wise contrastive learning, each tailored to enhance sensitivity to forgery cues. A novel interaction mechanism is also proposed to enable bidirectional knowledge exchange between the adapter and CLIP: the adapter absorbs low-level CLIP visual tokens to enrich its learning, and in return, uses attention bias to guide CLIP's focus toward forgery-specific features without altering CLIP's core parameters. Despite having only $5.7M$ trainable parameters, ForensicsAdapter achieves state-of-the-art performance across multiple benchmarks and proves highly robust to dataset shifts and image perturbations, making it a strong baseline for future CLIP-based forgery detection work.

A.4.2 More Details on the Datasets

Seven standard and challenging datasets are used in our experiments, including FF++ [23], Celeb-DF (CDF) [80], Google Deepfake Detection (DFD) [83], WildDeepfake (DFW) [82], Deepfake Detection Challenge (DFDC) [88], its preview version (DFDCP) [81], and DF40 [14]. These datasets are selected based on the following criteria: 1) they are widely used for evaluation in the research community; 2) they contain diverse source fakes, including those generated by black-box or undisclosed methods, for which no implementation details or prior knowledge are available; 3) they apply various perturbation methods. An overview of the selected benchmarks is presented in Table A.7.

Specifically, FF++ is typically used for training, while the others serve as test sets in the cross-dataset evaluation protocol. It consists of 1000 real videos and 4000 fakes videos with different compression levels, which are generated by four manipulation techniques, i.e., Deepfakes (DF) [116], Face2Face (F2F) [117], FaceSwap (FS) [118], and NeuralTextures (NT) [119]. In our experiments, we adopt the raw version of FF++. CDF is one of the most common and challenging benchmarks typically used to assess the generalizability of deepfake

detectors under the cross-dataset evaluation setting [25, 37, 32, 31, 26, 65, 62, 28]. It contains highly realistic deepfakes. DFD includes more than 3000 forged videos featuring 28 actors in various scenes. DFW is fully sourced from the internet, without prior knowledge of generation methods. DFDC and its preview version (DFDCP) is a challenging, large-scale dataset that contains numerous distorted videos applied by several perturbation techniques such as compression, noise, etc. The recent DF40 benchmark, a highly diverse and large-scale dataset comprising 40 distinct deepfake techniques, enables more comprehensive evaluations for the next generation of deepfake detection.

A.4.3 More Evaluation Metrics

In addition to score polarization measured by Wasserstein Distance (W), AUC, and our proposed Cross-AUC, we report additional metrics to support a more in-depth analysis. These include Accuracy (ACC), Balanced Accuracy (BACC), Precision (P), Recall (R), Specificity (S), F1-score (F1), and Equal Error Rate (EER). We consider the *Fake* and *Real* classes as the *Positive* and *Negative* classes, respectively. While some of these metrics are standard, we include them for completeness and improved presentation.

Equal Error Rate (EER): is a common metric for evaluating the performance of biometric authentication systems, such as speaker verification or facial recognition. It represents the point at which the False Acceptance Rate (FAR) and the False Rejection Rate (FRR) are equal. At this threshold, the system is equally likely to incorrectly accept an unauthorized user (false positive) as it is to incorrectly reject an authorized user (false negative). A lower EER indicates better overall system performance, as it reflects a more optimal trade-off between FAR and FRR.

In the context of Deepfake Detection, EER is used to evaluate the ability of a detector to distinguish between real and manipulated (fake) content. Here, the FAR corresponds to the rate at which fake content is incorrectly classified as real, and the FRR corresponds to the rate at which real content is incorrectly classified as fake. EER provides a single value that reflects the detector's robustness and balance between these two types of errors.

Accuracy (ACC): It is computed based on the percentage of correct predictions as follows,

$$\text{ACC} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}, \tag{A.3}$$

where TP, FP, TN, and FN denote True Positives, False Positives, True Negatives, and False Negatives, respectively.

While commonly used, ACC can be misleading when the dataset is imbalanced [155, 157].

For instance, in the context of deepfake detection, standard datasets such as FF++ contain four times more fake than real samples (see Table A.7). As a result, a model biased toward predicting the majority class (Fake) may achieve high ACC without truly distinguishing between real and fake content. To address this, we also report Balanced Accuracy (BACC), which provides a more informative view under class imbalance.

Balanced Accuracy (BACC): It is calculated based on the average recall values for both positive and negative classes.

$$\text{B-ACC} = \frac{1}{2}\left(\frac{\text{TP}}{\text{TP}+\text{FN}} + \frac{\text{TN}}{\text{TN}+\text{FP}}\right). \tag{A.4}$$

Specificity (S): It describes how well the model detects the negative samples. It is calculated as the proportion of correct negative predictions over the actual total number of negative samples. The formula is defined below,

$$\text{S} = \frac{\text{TN}}{\text{TN}+\text{FP}}. \tag{A.5}$$

Precision (P): It describes how correct the positive predictions are. It is calculated as the proportion of correctly predicted positive samples over the total number of predictions classified as positive. The formula is defined below,

$$\text{P} = \frac{\text{TP}}{\text{TP}+\text{FP}}. \tag{A.6}$$

Recall (R): It describes how well the model detects the positive samples. It is calculated as the proportion of correct positive predictions over the actual total number of positive samples. The formula is defined below,

$$\text{R} = \frac{\text{TP}}{\text{TP}+\text{FN}}. \tag{A.7}$$

F1-score (F1): It is often used to assess the overall performance of a classification model. It is computed as follows,

$$\text{F1} = 2.\frac{\text{P.R}}{\text{P}+\text{R}}. \tag{A.8}$$

Table A.8 Performance of the selected models on all datasets. τ: Optimal threshold; ϕ_τ: variance of the top–K optimal thresholds in each dataset; W_{KDE}, W_{Q}, W_{GMM}, W_{BD}: model polarities using Wasserstein distances on densities estimated with non-parametric approaches (Kernel Density Estimation (KDE), Quantile (Q)), and parametric approaches assuming two types of distributions, *i.e.*, Gaussian Mixture Model (GMM) and Beta Distribution (BD). Gray represents the dataset the model was trained on. Red highlights the best performance and polarities.

		FF++	CDF	DFD	DFW	DFDCP	DFDC	DF40	Average	**Combined**
Xception [46]	ACC (%)	85.32	43.19	82.59	58.06	53.44	55.92	57.56	62.29	58.53
	BACC (%)	87.31	53.55	79.79	57.95	63.7	55.84	67.17	66.47	61.6
	AUC (%)	93.6	62.1	92.24	63.71	70.56	58.98	81.59	74.68	68.95
	P (%)	96.03	74.48	83.24	57.8	94.54	57.71	41.95	72.25	75.2
	R (%)	81.34	21.72	90.58	64.62	50.48	42.39	92.49	63.37	47.26
	S (%)	93.28	85.38	69	51.28	76.92	69.29	92.4	76.79	75.94
	F1 (%)	88.08	33.64	86.76	61.02	65.82	48.88	57.72	63.13	58.04
	EER (%)	13.43	42.69	15.5	40.51	33.33	43.58	25.6	30.66	35.68
	τ	0.4	0.24	0.7	0.57	0.4	0.3	0.24	0.4	0.31
	ϕ_τ	0.0004	0.0004	0.0002	0.0007	0.0010	0.0001	0.0001	0.0004	0.0001
	W_{KDE}	0.524	0.064	0.430	0.110	0.146	0.078	0.239	0.227	0.168
	W_{BD}	0.549	0.068	0.446	0.122	0.164	0.060	0.232	0.234	0.152
	W_{Q}	0.661	0.077	0.524	0.139	0.183	0.091	0.264	0.277	0.190
	W_{GMM}	0.616	0.069	0.496	0.149	0.118	0.081	0.249	0.254	0.180
SLADD [31]	ACC (%)	81.84	64.49	65.92	51.15	**87.06**	53.75	50.9	65.01	55.77
	BACC (%)	80.03	50.95	54	50.45	53.51	53.9	59.55	57.48	53.38
	AUC (%)	88.31	56.65	97.22	48.63	65.98	57.37	72.25	69.48	57.18
	P (%)	87.07	66.73	64.8	51.05	89.52	52.3	82.25	70.53	63.29
	R (%)	85.44	92.55	**100**	**93.65**	**96.76**	79.14	36.86	83.48	64.56
	S (%)	74.62	9.35	8	7.25	10.25	28.65	82.24	31.48	42.2
	F1 (%)	86.25	77.55	78.7	66.08	**93**	62.98	50.91	73.63	63.92
	EER (%)	18.65	44.44	7.5	52.22	41.02	45.21	29.59	34.09	46.4
	τ	0.51	0.53	0.57	0.55	0.53	0.53	0.45	0.52	0.52
	ϕ_τ	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
	W_{KDE}	0.060	0.001	0.098	0.004	0.022	0.012	0.003	0.028	0.013
	W_{BD}	0.078	0.003	0.079	0.001	0.007	0.008	0.024	0.029	0.013
	W_{Q}	0.079	0.005	0.090	0.002	0.015	0.010	0.026	0.032	0.014
	W_{GMM}	0.079	0.005	0.089	0.004	0.015	0.010	0.027	0.033	0.014
RECCE [32]	ACC (%)	92.28	70.01	75.18	54.6	75.28	63.4	71.21	71.7	68.12
	BACC (%)	88.8	58.57	66.5	54.05	70.4	63.52	53.89	65.1	61.4
	AUC (%)	**99.56**	69.02	97.82	65.19	82.24	71.19	82.94	81.13	76.91
	P (%)	90.16	70.62	71.72	53.2	94.42	59.25	70.78	72.87	67.17
	R (%)	**99.25**	93.75	**100**	88.43	76.69	**84.46**	**99.31**	**91.69**	**92.82**
	S (%)	78.35	23.39	33	19.67	64.1	42.58	8.48	38.51	29.98
	F1 (%)	94.49	80.56	83.53	66.43	84.64	69.64	82.65	80.27	77.94
	EER (%)	**1.49**	36.25	8	39.81	28.2	34.43	24.48	24.66	31.85
	τ	0.98	0.74	0.99	0.65	0.53	0.58	0.97	0.77	0.86
	ϕ_τ	0.00001	0.0002	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
	W_{KDE}	0.631	0.126	0.352	0.096	0.212	0.181	0.153	0.250	0.221
	W_{BD}	0.569	0.124	0.280	0.116	0.240	0.182	0.117	0.233	0.201
	W_{Q}	0.716	0.136	0.355	0.115	0.243	0.203	0.144	0.273	0.232
	W_{GMM}	0.692	0.134	0.349	0.111	0.227	0.203	0.142	0.265	0.228
SBI [26]	ACC (%)	80.59	58.57	90.92	55.06	35.91	55.12	40.14	59.47	51.7
	BACC (%)	85.26	68.17	91.35	55.76	63.91	54.86	55.98	67.89	59.95
	AUC (%)	98.23	84.65	96.78	60.83	86.83	69.77	72.05	81.3	74.43
	P (%)	**99.47**	97.01	95.61	**91.8**	**100**	**97.9**	92.86	**96.37**	95.84
	R (%)	71.26	38.69	89.7	12.69	27.83	9.94	14.44	37.79	21.34
	S (%)	**99.25**	**97.66**	93	**98.82**	**100**	**99.78**	97.52	**98**	**98.57**
	F1 (%)	83.04	55.31	92.56	22.31	43.54	18.05	24.99	48.54	34.9
	EER (%)	6.71	24.56	9	46.83	20.51	36.11	33.36	25.29	32.57
	τ	0.21	0.28	0.46	0.16	0.19	0.14	0.17	0.23	0.17
	ϕ_τ	0.0006	0.0001	0.0008	0.0001	0.0002	0.0001	0.0001	0.0002	0.0001
	W_{KDE}	0.503	0.215	0.516	0.078	0.220	0.097	0.103	0.248	0.152
	W_{BD}	0.513	0.219	0.500	0.089	0.224	0.116	0.112	0.253	0.182
	W_{Q}	0.576	0.238	0.577	0.085	0.244	0.108	0.116	0.278	0.172
	W_{GMM}	0.567	0.236	0.564	0.084	0.245	0.105	0.115	0.274	0.171

		FF++	CDF	DFD	DFW	DFDCP	DFDC	DF40	Average	**Combined**
CADDM [27]	ACC (%)	95.27	72.19	80.74	69.01	52.01	63.72	43.7	68.09	58.55
	BACC (%)	93.47	72.56	74	69.35	68.5	63.6	57.41	71.27	63.32
	AUC (%)	99.26	80.7	99.52	76.31	71	70.33	76.28	81.91	71.53
	P (%)	94.31	84.21	76.58	84.13	97.33	72.92	87.83	85.33	81.51
	R (%)	98.88	71.43	**100**	48.07	47.25	42.97	21.46	61.43	41.01
	S (%)	88.06	73.68	48	90.63	89.75	84.22	93.36	81.1	85.64
	F1 (%)	96.54	77.3	86.73	61.18	63.61	54.08	34.49	67.7	54.56
	EER (%)	2.24	27.49	3.5	30.91	38.46	36.06	30.48	24.16	35.97
	τ	0.97	0.45	0.93	0.37	0.44	0.37	0.22	0.53	0.31
	ϕ_τ	0.0253	0.0001	0.0001	0.0001	0.0011	0.0001	0.0001	0.0038	0.0001
	W_{KDE}	0.661	0.203	0.436	0.210	0.156	0.155	0.137	0.280	0.176
	W_{BD}	0.607	0.203	0.386	0.238	0.182	0.165	0.146	0.275	0.214
	W_{Q}	0.726	0.226	0.466	0.246	0.179	0.174	0.151	0.310	0.198
	W_{GMM}	0.714	0.224	0.459	0.241	0.174	0.173	0.150	0.305	0.197
LAA-Net [62]	ACC (%)	**96.67**	84.72	92.5	64.52	59.74	57.68	37.79	70.51	45.35
	BACC (%)	**97.14**	**87.71**	88.67	65.01	72.36	57.37	65.99	76.32	66.15
	AUC (%)	99.45	**95.45**	98.47	79.52	86.48	72.6	86.04	88.28	**86.51**
	P (%)	99.26	**98.15**	91.71	85.22	97.4	90.99	**99.66**	94.62	**98.27**
	R (%)	95.71	78.24	98.19	36.59	55.83	16.33	33.63	59.21	35.59
	S (%)	98.57	97.18	79.15	93.43	88.89	98.41	**98.35**	93.42	96.71
	F1 (%)	**97.45**	87.07	94.84	51.19	70.98	27.69	50.29	68.5	52.25
	EER (%)	2.86	**11.86**	5.96	27.27	**18.52**	33.91	**22.2**	17.51	**21.4**
	τ	0.23	0.3	0.96	0.06	0.08	0.03	0.11	0.25	0.083
	ϕ_τ	0.0161	0.0004	0.0001	0.0001	0.0005	0.0001	0.0001	0.0024	0.0001
	W_{KDE}	**0.831**	**0.511**	0.633	**0.262**	**0.334**	0.169	0.281	**0.431**	0.302
	W_{BD}	**0.749**	**0.495**	0.547	0.248	**0.357**	0.189	0.272	**0.408**	0.296
	W_{Q}	**0.927**	**0.582**	0.708	**0.285**	**0.446**	0.156	0.299	**0.486**	0.318
	W_{GMM}	**0.911**	**0.579**	0.690	**0.292**	**0.417**	0.154	0.286	**0.475**	0.317
ForensicAdapter [28]	ACC (%)	89.3	**85.21**	**97.96**	**76.61**	70.11	**75.46**	**71.96**	**80.94**	**76.19**
	BACC (%)	84.89	79.94	**97.56**	**76.72**	**75.33**	**75.39**	**77.12**	**80.99**	**78.15**
	AUC (%)	98.39	94.94	**99.8**	**80.95**	**87.04**	**83.88**	**86.3**	**90.18**	85.96
	P (%)	87.38	83.9	**97.68**	81.32	96.8	84.2	93.82	89.3	89.36
	R (%)	98.13	**96.13**	99.11	70.07	68.61	62.34	63.6	79.71	68.99
	S (%)	71.64	63.74	**96**	83.37	82.05	88.43	90.64	82.26	87.32
	F1 (%)	92.44	**89.6**	**98.39**	**76.27**	80.3	**71.64**	75.8	**83.49**	77.86
	EER (%)	5.22	12.28	**1.5**	**25.53**	23.08	**24.03**	22.32	**16.28**	22.39
	τ	0.76	0.64	0.65	0.5	0.43	0.4	0.38	0.53	0.43
	ϕ_τ	0.0013	0.0003	0.0027	0.0001	0.0007	0.0001	0.0001	0.0007	0.0001
	W_{KDE}	0.497	0.338	**0.659**	0.251	0.259	**0.257**	**0.291**	0.365	**0.304**
	W_{BD}	0.460	0.334	**0.624**	**0.253**	0.268	**0.266**	**0.294**	0.357	**0.306**
	W_{Q}	0.540	0.374	**0.713**	0.285	0.293	**0.286**	**0.322**	0.402	**0.336**
	W_{GMM}	0.521	0.371	**0.709**	0.284	0.288	**0.285**	**0.320**	0.397	**0.335**

A.4.4 Additional Results and Discussions

In Table A.8, we evaluate seven deepfake detection models [23, 31, 32, 26, 27, 62, 28], all trained on FF++ [23], across six unseen datasets [80, 83, 82, 88, 81, 14] and our proposed Combined test set. We report standard classification metrics (ACC, BACC, P, R, S, F1), EER, AUC, optimal thresholds (τ), threshold variance (ϕ_τ), and polarization scores based on Wasserstein distance (W) using KDE, BD, Q, and GMM estimators. Note that a fixed threshold of 0.5 is used to compute the classification metrics.

Cross-Dataset Generalization Performance

This setup directly tests cross-dataset generalization, a key challenge in deepfake detection, where models often fail when faced with manipulations or domains not seen during training.

It can be observed that results show substantial performance variation across datasets. ForensicAdapter consistently performs well across all test sets, maintaining strong accuracy, AUC, and F1, which reflects robust generalization. RECCE also performs well, especially in R, achieving high values on datasets like DFD and DF40. However, its AUC drops below 70% on CDF and DFW, indicating that its generalization is not uniformly strong across all domains. LAA-Net achieves very high AUC on some datasets, such as CDF (95.45%) and DFD (98.47%), but its performance decreases on others like DFDC (72.60%) and DF40 (86.04%), showing it is more sensitive to certain shifts. CADDM performs moderately overall, with solid AUC and accuracy on several datasets, but lower R and higher EER on DFDC and DF40 indicate weaker transfer to more challenging domains.

In contrast, SLADD and Xception experience sharp performance drops outside of FF++. Xception, for example, drops from 85.32% ACC on FF++ to below 60% on most other datasets, showing poor adaptability. SLADD performs better in R but lacks overall consistency in F1 and AUC. This is partly explained by their weak score separability: both models show low polarity, meaning they struggle to distinguish real and fake samples confidently. Even though SLADD maintains stable thresholds, its overlapping score distributions lead to unreliable decisions when tested on unfamiliar data (Figure 6.6 in the chapter 6). The relationship between Polarization and classification metrics is further discussed in Section A.4.4.

The AUC trends highlight these generalization patterns more clearly. ForensicAdapter is the only model that maintains AUC above 80% across all test datasets, indicating consistent class separation under domain shifts. Other models, including RECCE, LAA-Net, SBI, and Xception, perform well on specific datasets but show noticeable drops on others, reflecting varying sensitivity to the target domain. Xception achieves competitive results on datasets like DFD and DF40, though its performance varies more widely across others. In comparison, SLADD shows the most limited generalization, with the lowest Combined AUC and consistently low polarity scores, suggesting that it struggles to separate real and fake samples reliably across domains.

Combined Test Set as a Realistic Generalization Benchmark

Our suggested Combined set, created by merging predictions from all test datasets, reflects real-world use cases where the content is highly mixed. It offers a challenging test of generalization, i.e., models are exposed to many manipulation types, compression levels, and domain shifts simultaneously.

ForensicAdapter again stands out with the highest Combined F1 (83.49%), low EER (16.28%), and strong AUC (85.96%). Its performance is both accurate and stable across

conditions. LAA-Net reaches the highest Combined AUC (86.51%) and excels in P and S but suffers from low R, leading to a lower Combined F1 (52.25%). RECCE maintains strong R and F1, showing consistent behavior even in the most varied testing condition. In contrast, models like SLADD, SBI, and Xception show noticeable performance drops on the Combined set. SLADD's high R leads to many false positives, while SBI and Xception exhibit more conservative behavior or limited score separability, affecting their reliability under mixed-domain conditions.

This comparison also highlights the limits of using per-dataset averages to assess generalization. While averages may suggest overall stability, they can mask poor performance on specific domains. For example, SBI shows a decent average AUC but fails on DFDC due to very low R. SLADD has high average R, but low S on several datasets leads to frequent false positives. In contrast, ForensicAdapter performs consistently across both average and Combined evaluations, confirming its robustness. In general, the Combined set provides a more realistic and reliable measure of performance under diverse conditions.

Threshold Stability and Confidence Behavior

We examine threshold behavior using ϕ_τ and τ, which reflect how consistently a model applies its decision boundary and how confident it is in its predictions.

RECCE and SLADD show the most stable threshold behavior. SLADD has identical ϕ_τ values of 0.0001 across all datasets, and RECCE maintains similarly low values between 0.00001 and 0.0002. Their τ values are also consistent. RECCE operates at higher τ (0.77 on average), indicating more cautious predictions. This stability is supported by its moderately high polarity on the Combined set ($W_{\text{Q}} = 0.232$), suggesting reasonably well-separated score distributions. SLADD, on the other hand, has the lowest polarity ($W_{\text{Q}} = 0.014$), showing that even with stable thresholds, the model struggles to confidently separate real and fake predictions.

SBI and Xception also show low ϕ_τ values, but their lower polarity scores suggest that threshold consistency does not translate into well-separated scores. Their decisions are more stable than confident. ForensicAdapter shows slightly more variation in threshold placement (e.g., 0.0027 on DFD), including the highest ϕ_τ on the Combined set (0.0007), but its high polarity ($W_{\text{Q}} = 0.336$) supports confident decisions despite small shifts in threshold.

LAA-Net and CADDM show less consistent threshold behavior, with noticeably higher ϕ_τ values on FF++ (0.0161 and 0.0253), and more fluctuation in τ across datasets. While both achieve high polarity, the instability in thresholds suggests more erratic decision behavior under domain shifts.

Overall, RECCE is the only model that combines both stable thresholds and clear score separability. Other models show strength in one aspect but not both, which may impact their reliability under unseen conditions.

Polarity and Its Relationship to Classification Metrics

We analyze polarity W using Wasserstein distance between the score distributions of real and fake samples, as a way to quantify how well a model separates the two classes. Since classification metrics like AUC, EER, and F1 are directly influenced by how confidently a model scores real versus fake inputs, we expect models with higher polarity to generally exhibit stronger performance. In our evaluation, we examine how this relationship holds across models under cross-domain conditions.

On the Combined test set, we observe that models with higher polarity, such as ForensicAdapter ($W_{\text{Q}} = 0.336$), LAA-Net ($W_{\text{Q}} = 0.318$), and RECCE ($W_{\text{Q}} = 0.232$), also achieve strong classification results. ForensicAdapter pairs the highest polarity with an AUC of 85.96% and F1 of 83.49%, indicating that well-separated score distributions contribute directly to confident and reliable decisions. RECCE follows a similar trend, where moderate-to-high polarity aligns with high R and competitive F1, further supported by its stable thresholds across domains. LAA-Net shows high polarity and the highest AUC (86.51%), which suggests good score separation. However, its R remains low, leading to a reduced F1 score (52.25%). This gap indicates that while the model distinguishes real and fake scores well, it tends to under-predict the fake class, possibly due to misaligned threshold behavior or domain sensitivity. As a result, high polarity does not always guarantee balanced classification outcomes.

In contrast, SLADD and Xception exhibit much lower polarity, with $W_{\text{Q}} = 0.014$ and 0.190 respectively. SLADD shows stable thresholds across datasets, but the score overlap between real and fake classes results in high false positive rates and low F1 on the Combined set. This confirms that stable thresholds alone do not guarantee good performance. Without clear separation in output scores, the model cannot make confident decisions under domain shift. Xception, while having slightly higher polarity than SLADD, struggles with both generalization and score separability. Its performance drops sharply outside of FF++, and the weak polarization across domains contributes to high error rates and poor adaptability. SBI, meanwhile, offers a different failure mode. Despite moderate polarity ($W_{\text{Q}} = 0.172$), it achieves very high P and S, but extremely low R. Its behavior reflects the use of conservative thresholds rather than confident score separation, a pattern supported by its relatively low τ values and narrow decision margins. While this approach minimizes false positives, it also prevents the model from detecting many fake samples, limiting its utility in general-purpose

settings.

These findings confirm that polarity is a strong indicator of how confidently a model distinguishes real and fake content, and it generally correlates with AUC and EER. However, it must be considered alongside threshold behavior and recall–precision balance to fully understand a model's generalization capacity. Models that combine high polarity with consistent thresholds and balanced decision strategies, such as ForensicAdapter and RECCE, are more reliable under cross-domain conditions.

Recall-Precision Trade-offs Across Models

While earlier sections analyze model performance from score separability and threshold stability perspectives. In this section, we focus on the practical trade-offs between R and P. Models like RECCE and SLADD tend to favor R, detecting more fake samples but at the cost of increased false positives. In contrast, SBI and LAA-Net show conservative behavior, prioritizing P and S while often missing actual deepfakes. ForensicAdapter offers the most balanced trade-off, maintaining high values for both R and P. These differences highlight how models vary not just in accuracy, but also in how cautiously or aggressively they make decisions, an important consideration for real-world deployment.

Failure Case Analysis

We observe consistent performance drops on certain datasets, revealing where generalization tends to fail. These cases highlight the sensitivity of some models to domain-specific shifts not reflected in overall averages.

In particular, the DFDC dataset poses a common challenge. Models such as LAA-Net, RECCE, and CADDM show significant declines in performance. For example, LAA-Net drops to 16.33% R and 27.69% F1, despite achieving top AUCs on other datasets. RECCE and CADDM also report high EERs (34.43% and 36.06%) and reduced F1, indicating difficulty adapting to DFDC's unique characteristics, such as higher compression or different manipulation styles. DF40 also reveals failure points. Xception and SBI, in particular, suffer from very low R (14.44% and 21.46%, respectively), despite maintaining high S. These patterns suggest that overly conservative models fail to detect fakes when the score distributions shift, even if their decision thresholds remain stable.

Such failures emphasize the limits of models that rely heavily on training distribution priors. While models like ForensicAdapter maintain robust results even under domain variation, others show clear breakdowns when faced with unseen manipulation types or domain properties not covered by FF++.

No. of Domains	Method	AUC_c	AUC_a	Cross-AUC
7 (All)	Xception [23]	68.95	74.68	**68.52**
	SLADD [31]	57.18	69.48	**58.37**
	RECCE [32]	76.91	81.13	**73.26**
	SBI [26]	74.43	81.30	**74.20**
	CADDM [27]	71.53	81.91	**75.31**
	LAA-Net [62]	86.51	88.28	**86.47**
	ForensicsAdapter [28]	85.96	90.18	**83.59**
6 (w/o DF40)	Xception [23]	65.13	73.53	**68.53**
	SLADD [31]	62.55	69.02	**57.62**
	RECCE [32]	74.16	80.83	**72.70**
	SBI [26]	75.62	82.84	**75.38**
	CADDM [27]	76.42	82.85	**76.88**
	LAA-Net [62]	84.79	88.66	**86.33**
	ForensicsAdapter [28]	87.36	90.83	**85.20**
5 (w/o DF40, DFDC)	Xception [23]	73.32	76.44	**73.06**
	SLADD [31]	67.58	71.35	**64.09**
	RECCE [32]	76.76	82.76	**78.51**
	SBI [26]	82.19	85.46	**80.81**
	CADDM [27]	83.13	85.35	**82.60**
	LAA-Net [62]	94.52	91.87	**93.77**
	ForensicsAdapter [28]	91.45	**92.22**	93.53
4 (w/o DF40, DFDC, DFDCP)	Xception [23]	74.58	77.91	**73.56**
	SLADD [31]	68.31	72.70	**65.87**
	RECCE [32]	81.91	**82.89**	79.33
	SBI [26]	82.09	85.12	**81.36**
	CADDM [27]	85.76	88.94	**87.57**
	LAA-Net [62]	95.72	93.22	**93.25**
	ForensicsAdapter [28]	92.75	93.52	**92.12**

Table A.9 Ablation study of different numbers of domains. **Red** highlights the highest or closest performance.

	Xception	SLADD	RECCE	SBI	CADDM	LAA-Net	ForensicsAdapter	Avg.
AUC_c	68.95	57.18	76.91	74.43	71.53	86.51	85.96	74.49
Cross-AUC_{WD}	**68.52**	**58.37**	73.26	**74.20**	**75.31**	86.47	83.59	**74.24**
Cross-AUC_{KL}	68.29	60.07	**74.22**	73.88	76.80	**86.54**	**87.05**	75.26
Cross-AUC_{JS}	54.61	65.73	60.82	66.05	62.22	67.48	63.45	62.90

Table A.10 Estimated Cross-AUC with different polarization estimators, i.e., Wasserstein Distance (WD), KL Divergence (KL), and Jensen-Shannon (JS) Distance. **Red** highlights the best estimator.

Ablation Study of Different Numbers of Domains

In Table A.9, we conduct an ablation study by progressively reducing the number of test domains from seven to four and measuring how Cross-AUC, average AUC (AUC_a), and Combined AUC (AUC_c) varies across different detectors. We found that Cross-AUC consistently remains close to AUC_c, even as domain diversity decreases, whereas AUC_a tends to overestimate the performance. This supports the stability of Cross-AUC under varying domain coverage, suggesting that it reflects real-world generalization risk more reliably than per-domain average.

Rank Reversal of Models

While Cross-AUC generally preserves the ranking implied by average AUC, exceptions can occur. For instance, Table 6.2 of the chapter 6 shows a reversal between ForensicsAdapter and LAA-Net: although ForensicsAdapter attains the highest average AUC, LAA-Net achieves higher Combined AUC and Cross-AUC. Such cases highlight that Cross-AUC is not only a tool for model comparison, but more importantly a practical measure of generalization stability across domains, an aspect crucial for real-world deployment under unpredictable data distributions.

Why Cross-AUC is Necessary

Evaluating on the Combined dataset reflects realistic deployment, as deepfake detectors are expected to face mixed-domain inputs. However, relying solely on the Combined AUC makes it difficult to interpret the gap between the average AUC and Combined AUC. Cross-AUC addresses this limitation by explicitly quantifying prediction polarization, thereby offering a clearer explanation of generalization behavior under domain shift. In other words, our results indicate that the degree of separability between real and fake predictions across domains is critical. This perspective opens a new direction for building robust deepfake detectors that account not only for accuracy ranking but also for prediction polarization across domains.

Method	Estimator	Polarization Score								
		FF++	CDF	DFD	DFW	DFDCP	DFDC	DF40	Average	Combined
Xception [23]	WD	0.661	0.077	0.524	0.139	0.183	0.091	0.264	0.277	0.190
	KL	23.597	17.919	16.039	15.340	20.217	4.925	11.431	15.638	2.999
	JS	0.797	0.712	0.794	0.699	0.792	0.413	0.554	0.680	0.345
SLADD [31]	WD	0.079	0.005	0.090	0.002	0.015	0.010	0.026	0.032	0.014
	KL	17.695	4.914	21.158	2.073	14.469	0.553	2.764	9.089	0.744
	JS	0.686	0.374	0.767	0.303	0.593	0.199	0.375	0.471	0.185
RECCE [32]	WD	0.716	0.136	0.355	0.115	0.243	0.203	0.144	0.273	0.232
	KL	26.704	17.120	4.266	12.087	20.226	4.199	1.922	12.361	1.187
	JS	0.824	0.701	0.755	0.628	0.791	0.464	0.518	0.669	0.418
SBI [26]	WD	0.576	0.238	0.577	0.085	0.244	0.108	0.116	0.278	0.172
	KL	21.895	18.545	22.352	9.984	20.633	5.346	6.802	15.080	5.859
	JS	0.813	0.738	0.813	0.557	0.809	0.381	0.426	0.648	0.389
CADDM [27]	WD	0.726	0.226	0.466	0.246	0.179	0.174	0.151	0.310	0.198
	KL	25.675	18.646	24.948	15.572	19.638	5.993	7.800	16.896	3.873
	JS	0.830	0.736	0.825	0.673	0.773	0.441	0.492	0.681	0.365
LAA-Net [62]	WD	0.927	0.582	0.708	0.285	0.446	0.156	0.299	0.486	0.318
	KL	25.423	21.371	23.247	19.313	19.500	6.680	12.617	18.307	9.745
	JS	0.822	0.800	0.804	0.742	0.749	0.397	0.562	0.696	0.528
ForensicsAdapter [28]	WD	0.540	0.374	0.713	0.285	0.293	0.286	0.322	0.402	0.336
	KL	23.222	21.466	24.085	17.018	20.781	11.225	13.890	18.812	10.989
	JS	0.819	0.798	0.830	0.708	0.815	0.557	0.590	0.731	0.535

Table A.11 Detailed polarization scores for different estimators, which are used to compute the Cross-AUC results reported in Table 6.6 of chapter 6 and Table A.10 in the supplementary.

Extended Analysis: Cross-AUC with Different Polarization Estimators

To assess the influence of the polarization component, we evaluate Cross-AUC with three alternative estimators: Wasserstein Distance, KL Divergence, and Jensen-Shannon (JS) Distance. Table A.10 reports the resulting Cross-AUC values, while Table A.11 details the underlying polarization scores from each estimator. The results show that both Wasserstein and KL yield polarization magnitudes that scale proportionally with the degree of domain mismatch, leading to Cross-AUC values that remain close to the combined AUC. In contrast, JS produces less discriminative polarization scores (Table A.11), which translate into weaker alignment in Cross-AUC (Table A.10). This suggests that although Cross-AUC is generally robust to the choice of estimator, the Wasserstein formulation offers the most stable and interpretable behavior, with KL as a competitive alternative.