Journal of Computer Science and Cybernetics, V.42, N.x (2026), 1–
DOI no. 10.15625/1813-9663/23599

E-STDN: EFFICIENT SPOOF TRACE DISENTANGLEMENT FOR FACE ANTI-SPOOFING SYSTEMS USING LIGHTWEIGHT MODELS

MINH PHAM NGOC, TRUNG PHAM DUY, LAM THU BUI*

Academy of Cryptography Techniques, No. 141 Chien Thang Street, Thanh Liet Ward, Ha Noi, Viet Nam

Crossref Similarity Check
Powered by iThenticate

Abstract. This paper introduces E-STDN, an efficient architecture for face anti-spoofing that enhances the original spoof trace disentanglement network (STDN) by addressing the trade-off between accuracy and computation cost. Inspired by MobileNet, our design remains lightweight while maintaining strong spoof trace representation capabilities. Experiments on the CelebA-Spoof dataset demonstrate that E-STDN achieves superior TDR@FDR=0.5 compared to previous baselines, while significantly reducing inference time relative to the original STDN. Overall, our work presents an improved model design that can contribute to the development of robust and practical face anti-spoofing systems.

Keyword. Face anti-spoofing, E-STDN, spoof trace learning, MobileNet.

1. INTRODUCTION

Face Anti-Spoofing (FAS) has seen substantial progress globally, with recent methods achieving near-perfect performance on standardized benchmarks. For instance, in the LivDet-Face 2021 competition, the leading algorithm attained an Attack Presentation Classification Error Rate (APCER) as low as 0.0074% [1], demonstrating the effectiveness of deep learning models that combine temporal and textural analysis. Similarly, on the Replay-Attack dataset [2], which includes over 1,300 spoofing videos captured under diverse conditions, state-of-the-art models have achieved a Half Total Error Rate (HTER) of 0% [3] successfully exploiting subtle motion discrepancies and reflectance anomalies. In the area of 3D mask detection, handcrafted methods based on Haralick features reached 100% accuracy on video-based spoofing detection on the 3DMAD dataset [4], suggesting that well-engineered traditional features can still rival end-to-end learning when the attack surface is well defined.

Despite these advancements, real-world deployments often reveal persistent limitations. A report by ID.me analyzing more than 80,000 spoofing attempts across 26 U.S. states between 2020 and 2021 highlighted the vulnerability of existing FAS systems when exposed to diverse user behaviors, environmental variability, and rapidly evolving spoofing techniques [5]. These findings underscore the need for more robust and theoretically grounded solutions capable of generalizing beyond controlled laboratory settings.

*Corresponding author.
E-mail addresses: phamngocminh1230@gmail.com (M.P. Ngoc); trungpd@actvn.edu.vn (T.P. Duy); lambt@actvn.edu.vn (L.T. Bui).

Computational efficiency is a critical factor in the design of robust face anti-spoofing systems. In real-world deployments, particularly on mobile devices and edge computing platforms (e.g., IoT security cameras, smart locks), resources such as memory, battery life, and processing power are strictly limited. Heavy deep learning models, despite their high performance, often suffer from high latency, which degrades user experience during authentication. Furthermore, offloading processing to the cloud introduces privacy concerns and dependency on stable network connections. Consequently, there is an urgent demand for FAS solutions that can deliver high security while maintaining the low latency and efficiency required for on-device execution.

Among modern approaches, the spoof trace disentanglement network (STDN) has attracted attention for its spoof trace regression mechanism, which explicitly models spoof-specific artifacts in the input space [6]. While effective, the original STDN framework is relatively heavy and computationally intensive, leading to slower inference, mainly due to its standard convolutional blocks and large model size. This makes it almost impossible to implement STDN for the practical cases.

Building on this, we propose the efficient spoof trace disentanglement network (E-STDN), a principled lightweight extension of STDN that explicitly targets architectural efficiency while preserving the core inductive bias of spoof trace disentanglement. Unlike mainstream lightweight anti-spoofing approaches that rely on generic backbones (e.g., EfficientNet or Vision Transformers) and formulate face anti-spoofing as a global binary classification problem, E-STDN is designed around the explicit modeling and disentanglement of spoof traces. This design choice enables the network to capture localized, fine-grained spoof artifacts, which are often attenuated or lost in generic lightweight models due to aggressive spatial pooling and the absence of domain-specific constraints. By embedding MobileNet-inspired efficiency mechanisms directly into the spoof trace disentanglement pipeline, E-STDN achieves a favorable balance between computational efficiency, inference speed, and representational fidelity, making it particularly suitable for resource-constrained and edge-deployed anti-spoofing scenarios.

It is worth noting that the primary objective of this work is to optimize the architectural efficiency of the STDN framework to enable feasible edge deployment, rather than pursuing absolute state-of-the-art performance on the public leaderboard. Consequently, our study focuses on quantifying the relative efficiency gains and trade-offs against the baseline STDN and standard lightweight backbones, rather than benchmarking against large-scale or multi-modal systems designed for server-side execution.

This paper is organized as follows. Section 2 reviews related work, including a summary of the original STDN architecture. Section 3 presents the mathematical setup of spoof trace learning. Section 4 details the E-STDN architecture and its components. Section 5 provides experimental results. Finally, Section 6 concludes the paper and outlines future directions.

2. RELATED WORK

2.1. Deep learning based face anti-spoofing

In the field of face anti-spoofing, current methodologies can be broadly categorized into two main approaches: those based on conventional RGB cameras and those leveraging specialized sensors (see Figure 1). Each approach presents distinct advantages, challenges, and

research directions, reflecting the ongoing effort to balance practical deployability with detection accuracy under complex real-world conditions.

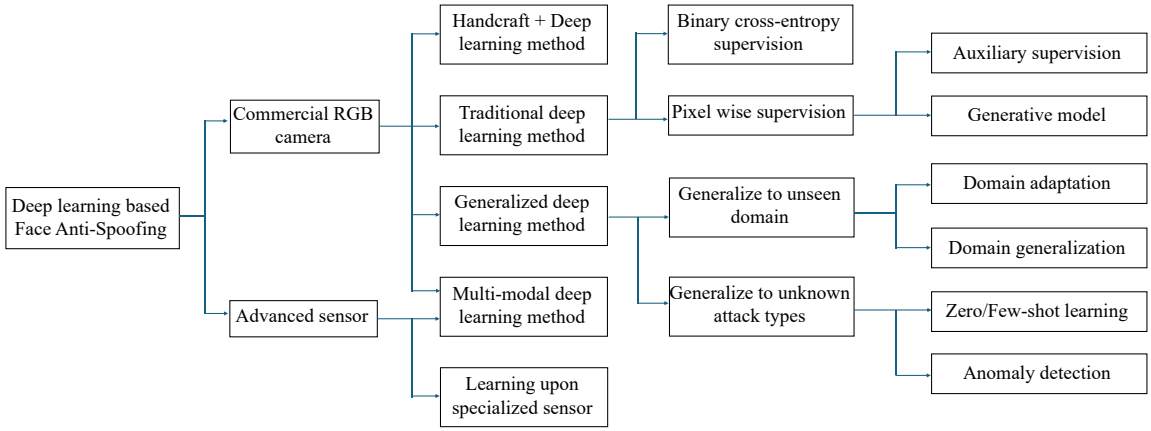

Figure 1: Different deep learning methods [7]

RGB-based face anti-spoofing methods are widely adopted due to their compatibility with existing devices, with early works relying on binary supervision and later approaches incorporating pixel-level labels, multi-task learning, and domain generalization to improve robustness [8, 9]. However, the limited modality of RGB inputs restricts their effectiveness against increasingly sophisticated attacks, motivating the use of specialized sensors such as depth, structured-light, and infrared imaging to capture richer geometric and physiological cues. These sensor-enhanced methods, often combined with anomaly detection, provide stronger generalization by identifying deviations from genuine biological patterns, particularly under unseen or zero-shot attack scenarios.

In the period of 2023–2025, the field has witnessed a significant shift towards advanced feature learning and multi-modal architectures. Recent studies like OC-SCMNet [10] have introduced spoof cue map guidance to learn latent features for one-class detection, addressing the limitations of binary supervision in CNNs. Similarly, multi-modal approaches such as [11] and [12] have focused on fusing RGB with infrared or depth signals to improve generalization against sophisticated attacks. Notably, He et al. [12] proposed a lightweight cross-fused network, validating the growing trend towards efficiency.

However, a critical gap remains between these advanced architectures and practical interpretability. While recent methods improve accuracy, they often function as "black boxes", potentially overfitting to dataset-specific biases (e.g., background context) rather than isolating the attack medium itself. In contrast, STDN-like architectures explicitly disentangle the spoof trace from the live face, offering a physically interpretable signal that generalizes well across domains. Our proposed E-STDN aims to bridge this gap by retaining the interpretability of trace disentanglement while adopting MobileNet-inspired efficiency to overcome the high computational cost of the original STDN.

2.2. Lightweight convolutional architectures

Over the past decade, CNN architectures have shifted from heavy, deep designs such as VGG and ResNet to lightweight models better suited for edge deployment. While deep

networks improve feature extraction, their high computational cost limits real-time face anti-spoofing on mobile devices. To address this, E-STDN adopts MobileNet-style efficiency by using depthwise separable convolutions, which decouple spatial and channel-wise filtering to reduce computation by approximately 8–9× compared to standard convolutions, while maintaining effective feature representation. This design enables E-STDN to meet practical latency and resource constraints, with the encoder configuration summarized in Table 1.

Table 1: MobileNetV1 architecture

Layer	Filter Shape	Stride	Output
Conv2D	$3 \times 3 \times 3 \rightarrow 32$	2	$112 \times 112 \times 32$
DW+PW Conv	$3 \times 3 \times 32,\ 1 \times 1 \times 32 \rightarrow 64$	1	$112 \times 112 \times 64$
DW+PW Conv	$3 \times 3 \times 64,\ 1 \times 1 \times 64 \rightarrow 128$	2	$56 \times 56 \times 128$
DW+PW Conv	$3 \times 3 \times 128,\ 1 \times 1 \times 128 \rightarrow 128$	1	$56 \times 56 \times 128$
DW+PW Conv	$3 \times 3 \times 128,\ 1 \times 1 \times 128 \rightarrow 256$	2	$28 \times 28 \times 256$
DW+PW Conv	$3 \times 3 \times 256,\ 1 \times 1 \times 256 \rightarrow 256$	1	$28 \times 28 \times 256$
DW+PW Conv	$3 \times 3 \times 256,\ 1 \times 1 \times 256 \rightarrow 512$	2	$14 \times 14 \times 512$
DW+PW Conv ×5	$3 \times 3 \times 512,\ 1 \times 1 \times 512 \rightarrow 512$	1	$14 \times 14 \times 512$
DW+PW Conv	$3 \times 3 \times 512,\ 1 \times 1 \times 512 \rightarrow 1024$	2	$7 \times 7 \times 1024$
DW+PW Conv	$3 \times 3 \times 1024,\ 1 \times 1 \times 1024 \rightarrow 1024$	1	$7 \times 7 \times 1024$
Global Avg Pool	GAP	-	$1 \times 1 \times 1024$
FC + Softmax	-	-	$1 \times 1 \times 1000$

Regarding the training setup for these baseline architectures, we employ the standard binary Cross-Entropy loss to supervise the classification task. The model parameters are optimized using the Adam algorithm [13] with default hyperparameters ($\beta_1 = 0.9, \beta_2 = 0.999$), ensuring consistent and stable convergence across all compared models.

2.3. Spoof trace disentanglement network

Inspired by adversarial learning paradigms, the spoof trace disentanglement network (STDN) [6] addresses limitations of conventional methods that fail to explicitly isolate spoof-specific patterns from facial features. STDN employs a disentanglement strategy to decouple spoof-related artifacts from identity-relevant information, thereby improving robustness against diverse spoofing attacks.

To provide a rigorous foundation for spoof trace learning, facial images are modeled as elements of a Hilbert space and formalize the decomposition of images into live components and spoof traces. This mathematical framework enables us to analyze the theoretical properties of the spoof trace disentanglement network (STDN) and establish conditions for stable learning and convergence (see Figure 2).

Definition 1. [Image space and norm] Consider the image space $\mathcal{I} = \mathbb{R}^{H \times W \times C}$, where H and W denote the image height and width, and C is the number of color channels. This space is equipped with the Frobenius inner product

$$\langle A, B \rangle_F = \sum_{i=1}^{H} \sum_{j=1}^{W} \sum_{k=1}^{C} A_{i,j,k} \cdot B_{i,j,k},$$

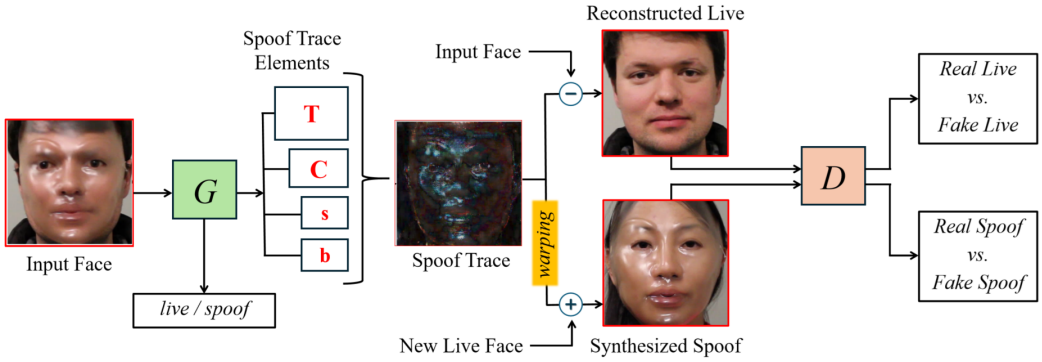

Figure 2: Framework of the spoof trace disentanglement network [6]

which induces the Frobenius norm $\|A\|_F = \sqrt{\langle A, A\rangle_F}$.

Definition 2. [Live and spoof image sets] We define:

$\mathcal{L} \subset \mathcal{I}$: the set of live (authentic) images.

$\mathcal{S} \subset \mathcal{I}$: the set of spoof (fake) images.

$\mathcal{D} = \mathcal{L} \cup \mathcal{S}$: the set of all images under consideration.

We assume that $\mathcal{L} \cap \mathcal{S} = \emptyset$ and that $\mathcal{L}$ is a closed and convex subset in the normed space $(\mathcal{I}, \|\cdot\|_F)$.

Definition 3. [Orthogonal projection onto the live set] Let $\mathcal{L} \subset \mathcal{I}$ denote the set of live (authentic) images, assumed to be closed and convex. For any $I \in \mathcal{I}$, we define the orthogonal projection of I onto $\mathcal{L}$ as

$$\hat{I} = \text{proj}_{\mathcal{L}}(I) := \arg\min_{X \in \mathcal{L}} \|I - X\|_F^2.$$

Definition 4. [Spoof trace operator] Given an image $I \in \mathcal{I}$, we define its spoof trace as the residual component

$$T(I) := I - \text{proj}_{\mathcal{L}}(I),$$

which is orthogonal to the live set $\mathcal{L}$.

2.3.1. Problem formulation

Given live face images $\mathcal{L} \subset \mathbb{R}^{N\times N\times 3}$ and spoofed images $\mathcal{S} \subset \mathbb{R}^{N\times N\times 3}$, the objective is to distinguish between live and spoof samples while estimating spoof traces. This is formulated as

$$\arg\min_{\hat{\mathbf{I}}} \|\mathbf{I} - \hat{\mathbf{I}}\|_F \quad \text{s.t. } \mathbf{I} \in (\mathcal{S} \cup \mathcal{L}),\ \hat{\mathbf{I}} \in \mathcal{L}, \tag{1}$$

where $\mathbf{I}$ is the input image, $\hat{\mathbf{I}}$ is the target live image, and $\mathbf{I} - \hat{\mathbf{I}}$ represents the spoof trace. Spoof traces are categorized by hierarchical levels: global features (color deviation), low-level features (makeup patterns), and high-level features (sharp edges, Moiré noise).

The spoof trace separation function is decomposed into components $\{s, b, C, T\}$ as

$$G(\mathbf{I}) = \mathbf{I} - \hat{\mathbf{I}} \tag{2}$$

$$= \mathbf{sI} + \mathbf{b} + \lfloor \mathbf{C} \rfloor_N + \mathbf{T}, \tag{3}$$

where $\mathbf{s} \in \mathbb{R}^{1\times 1\times 3}$ denotes the color deviation coefficient, $\mathbf{b} \in \mathbb{R}^{1\times 1\times 3}$ represents color balance deviation, $\mathbf{C} \in \mathbb{R}^{L\times L\times 3}$ contains smooth content features, and $\mathbf{T} \in \mathbb{R}^{N\times N\times 3}$ captures high-level features.

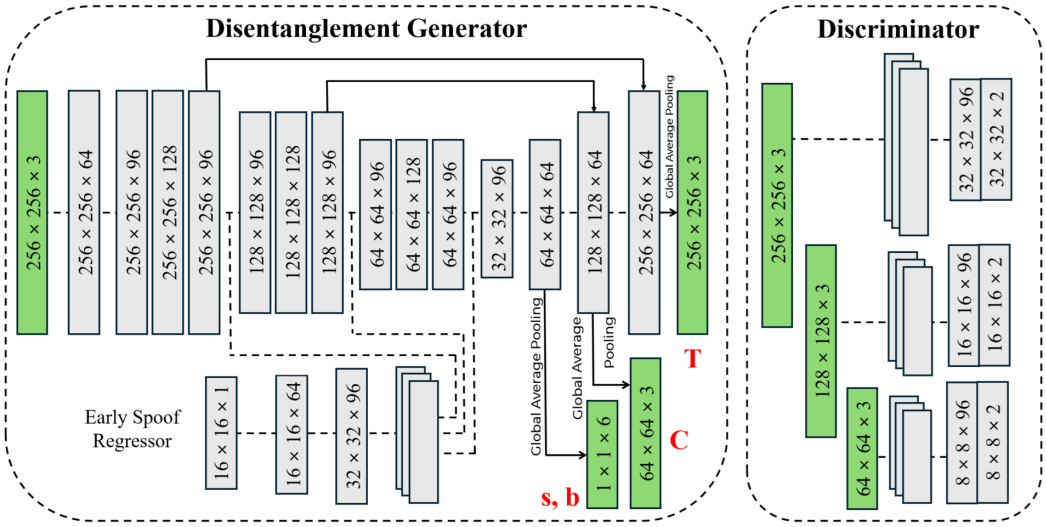

Figure 3: The architecture of the spoof trace disentanglement network [6]

2.3.2. Spoof trace disentanglement and training strategy

The framework integrates spoof trace disentanglement, reconstruction–synthesis, multi-scale discrimination, and supervised learning into a unified adversarial training pipeline. The generator follows a UNet-style architecture, exploiting the decoder upsampling path to disentangle spoof components at multiple resolutions (see Figure 3). Specifically, global average pooling is applied to the first, middle, and final decoder blocks to extract low-level $(\mathbf{s}, \mathbf{b})$, mid-level $(\mathbf{C})$, and high-level $(\mathbf{T})$ spoof components, respectively. Skip connections preserve fine spatial details, while depthwise separable convolutions ensure computational efficiency.

An Early Spoof Regressor (ESR) operates on bottleneck features to predict a spoof mask $\mathbf{M} \in \mathbb{R}^{K\times K}$ $(K = 16)$. The final spoof prediction combines ESR response and spoof trace intensity

$$\text{score} = \frac{1}{2K^2}\|\mathbf{M}\|_1 + \frac{\alpha_0}{2N^2}\|\mathbf{G}(\mathbf{I})\|_1, \tag{4}$$

where $N = 256$ is the input resolution and α_0 controls the contribution of spoof intensity.

Spoof traces are utilized in two ways: (i) *reconstruction*, which recovers authentic faces by subtracting estimated traces, $\mathbf{I}_{\text{real}} = \mathbf{I} - \mathbf{G}(\mathbf{I})$; and (ii) *synthesis*, which transfers disentangled traces onto clean faces to generate novel spoof samples. To ensure geometric consistency during synthesis, dense offset fields derived from 3DMM landmarks [14] are employed. A 3D Morphable Model (3DMM) is a statistical parametric model that represents facial geometry using a low-dimensional set of coefficients learned from aligned 3D face scans [14]

$$\mathbf{G}_i = g(\mathbf{I}_i \mid \boldsymbol{\theta}_g), \quad \widetilde{\mathbf{G}}_{i\to j} = g(\mathbf{I}_i \mid \boldsymbol{\theta}_g + \Delta\mathbf{p}_{i\to j}), \tag{5}$$

where $\Delta\mathbf{p}_{i\to j}$ is computed via Delaunay triangulation-based interpolation,

$$\Delta\mathbf{p}_{i\to j} = \text{Tri}(\mathbf{p}_0, \mathbf{s}_i, \Delta\mathbf{s}_{i\to j}). \tag{6}$$

To enforce realism at different spatial scales, three PatchGAN discriminators $\{D_1, D_2, D_3\}$ operate on resolutions 256^2, 128^2, and 64^2, capturing fine textures, mid-level patterns, and global structure, respectively.

Training is formulated as a multi-objective optimization with several loss terms. The ESR loss supervises spoof localization

$$L_{\text{ESR}} = \frac{1}{K^2} \left| E_{x\sim\mathcal{L}}[\|\mathbf{M}\|_1] + E_{x\sim\mathcal{S}}[\|\mathbf{M}_s - \mathbf{1}\|_1] \right|. \tag{7}$$

The generator adversarial loss encourages realistic reconstruction and synthesis under multi-scale discrimination

$$L_G = \sum_{n=1}^{3} \left| E_{x\sim\mathcal{S}}[(D_n^1(\mathbf{I}_r - G_r) - 1)^2] + E_{x\sim\mathcal{L},\mathcal{S}}[(D_n^2(\mathbf{I}_r + G_{j\to i}) - 1)^2] \right|_2. \tag{8}$$

The discriminator loss ensures balanced adversarial learning

$$\begin{aligned} L_D = \sum_{n=1}^{3} \Big(& E_{x\sim\mathcal{L}}[(D_n^1(\mathbf{I}_r) - 1)^2] + E_{x\sim\mathcal{S}}[(D_n^1(\mathbf{I}_r - G_r))^2] \\ & + E_{x\sim\mathcal{S}}[(D_n^2(\mathbf{I}_r + G_{j\to i}) - 1)^2] + E_{x\sim\mathcal{L}}[(D_n^2(\mathbf{I}_r + G_{j\to i}))^2] \Big). \end{aligned} \tag{9}$$

Trace regularization constrains spoof intensity while preserving domain variation

$$L_R = \beta\, E_{x\sim\mathcal{L}}[\|\mathbf{G}(\mathbf{I}_r)\|_2^2] + E_{x\sim\mathcal{S}}[\|\mathbf{G}(\mathbf{I}_s)\|_2^2], \tag{10}$$

and the supervised pixel loss enforces consistency for synthesized samples

$$L_P = E_{x\sim\mathcal{L},\mathcal{S}}\big[\|\mathbf{G}(\mathbf{I}_r + G_{j\to i}) - \text{sg}(|G_{j\to i}|)\|_1\big]. \tag{11}$$

Following [6], training proceeds in three sequential phases. The *generation phase* optimizes spoof extraction, reconstruction, and synthesis

$$L_{\text{Gen}} = \alpha_1 L_G + \alpha_2 L_{\text{ESR}} + \alpha_3 L_R. \tag{12}$$

The *discrimination phase* updates multi-scale discriminators using L_D. Finally, the *supervision phase* refines spoof trace learning with labeled data

$$L_{\text{Sup}} = \alpha_4 L_{\text{ESR}} + \alpha_5 L_P. \tag{13}$$

3. EFFICIENT SPOOF TRACE DISENTANGLEMENT NETWORK

Inspired by the idea in Subsection 2.2 (about the lightweight architecture), Subsection 2.3 (about the STDN architecture) and practical insights during training, several limitations of the original STDN architecture have been identified, especially when considering deployment in real-world environments requiring low-latency and resource-efficient inference:

- The network design remains suboptimal, relying on conventional bottleneck-style CNN blocks rather than more recent lightweight convolutional architectures such as EfficientNet or MobileNet. These modern blocks offer a better trade-off between accuracy and model complexity, enabling significant reduction in parameters and computational cost.

- Training and inference are time-consuming due to the presence of high-cost synthesis and face reconstruction operations, as well as the use of an encoder–decoder structure with skip connections inspired by U-Net. These connections, while helpful for preserving spatial details, have not been optimized for computational efficiency.

To address the aforementioned challenges, this section introduces an improved version of STDN, aiming to reduce the overall number of parameters while preserving competitive detection accuracy by incorporating lightweight architectural components.

3.1. Architecture overview

The original U-Net backbone is replaced with EfficientUNET, which leverages mobile-optimized building blocks to reduce computational complexity. The core innovation lies in the EfficientBlock design, which combines depthwise separable convolutions with residual connections forming depthwise separable bottlenecks that significantly enhance computational efficiency.

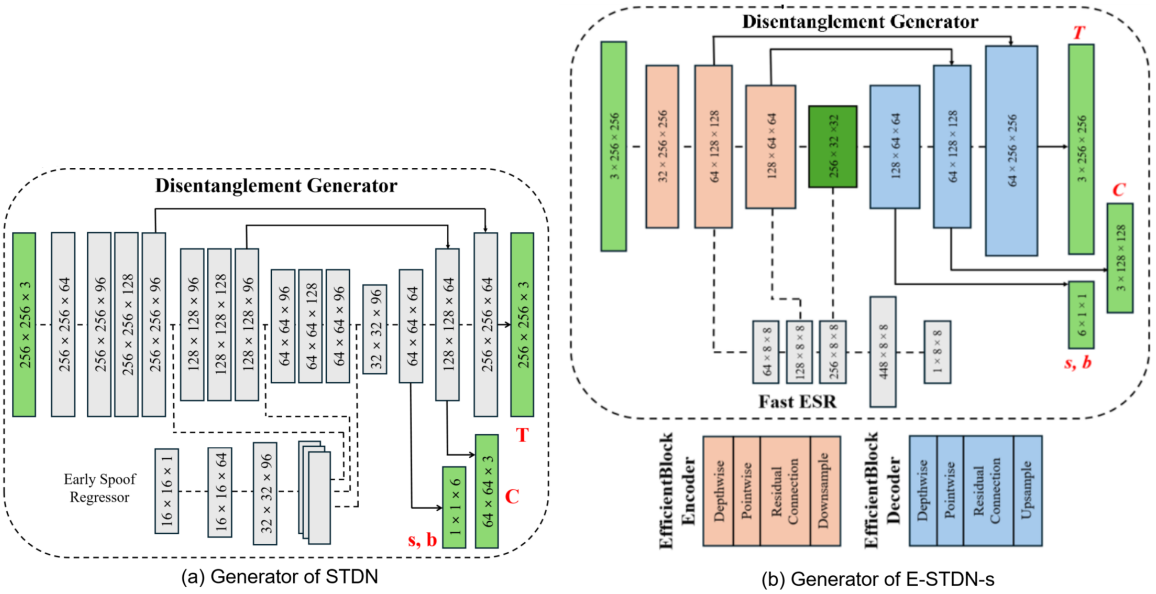

Figure 4: Schematic comparison of the disentanglement generator architectures. (a) The original STDN generator [6] relies on standard convolutional blocks. (b) The proposed E-STDN-s generator improved by replacing standard blocks with lightweight EfficientBlocks. For detailed architectural illustrations, please refer to Fig. 3 for (a) and Fig. 5 for (b).

To visually highlight the architectural optimizations, Figure 4 presents a side-by-side schematic comparison between the generator of the original STDN and our proposed E-STDN-s. As illustrated in Figure 4(a), the baseline STDN is built upon computationally intensive standard convolutional blocks and employs a complex, multi-branch Early Spoof Regressor to extract traces. In contrast, Figure 4(b) demonstrates the structural transformation in E-STDN-s, where these heavy components are systematically replaced. Specifically, the standard blocks are substituted by EfficientBlocks, which utilize depthwise separable convolutions within an inverted residual structure to decouple spatial filtering from channel

mixing. Furthermore, the original heavy regressor is streamlined into a FastESR module, which replaces redundant convolutional branches with efficient adaptive pooling and feature concatenation mechanisms ($448 \times 8 \times 8$) to aggregate multi-scale cues with minimal overhead.

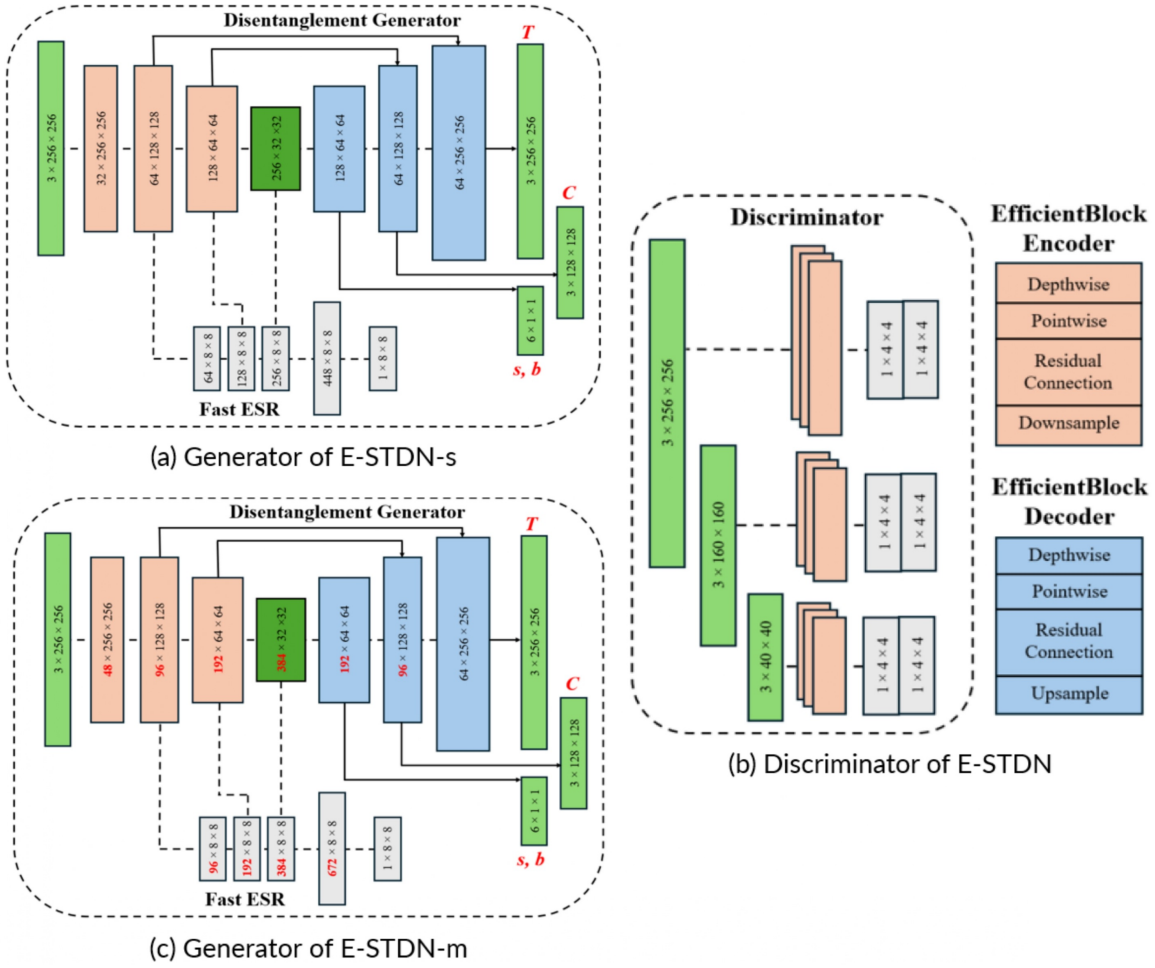

Figure 5: The efficient spoof trace disentanglement network architectures

Each EfficientBlock adopts an inverted residual design inspired by MobileNet, comprising an expansion phase (1×1 pointwise convolution), a depthwise 3×3 convolution for spatial filtering, and a projection phase (1×1 pointwise convolution) to restore channel dimensionality. This depthwise separable bottleneck significantly reduces computational cost (approximately 8–9×) compared to standard convolutions while preserving representational capacity. When input and output dimensions match, a residual connection is applied to improve gradient flow and support deeper network training.

Crucially, we employ group normalization (GN) instead of batch normalization (BN). This choice is theoretically grounded in the instability of BN when training with small batch sizes (e.g., $N = 8$ in our setup). While BN computes statistics across the batch dimension, GN divides the channels into G groups and computes the mean μ_i and standard deviation σ_i for a group i within a single sample as

$$\mu_i = \frac{1}{m}\sum_{k\in\mathcal{S}_i} x_k, \quad \sigma_i = \sqrt{\frac{1}{m}\sum_{k\in\mathcal{S}_i}(x_k-\mu_i)^2 + \epsilon}, \tag{14}$$

where x_k represents the feature values and $\mathcal{S}_i$ denotes the set of pixel indices within the group [15]. Since the set $\mathcal{S}_i$ is defined within a single sample, the normalization is independent of the batch size N. This eliminates the estimation noise inherent to BN in memory-constrained training, ensuring stable convergence for our E-STDN.

Group normalization layers are strategically integrated within each bottleneck to enhance training stability and ensure consistent performance across varying batch sizes, contributing to the theoretical robustness of the architecture. The overall architecture of the eficient spoof trace disentanglement network (E-STDN) is illustrated in Figure 5. We come up with two versions of E-STDN, namely small (s) and medium (m) ones.

3.1.1. E-STDN-s (small variant for edge deployment)

The small variant is optimized for deployment in resource-constrained environments where computational efficiency and memory footprint are critical. As summarized in Table 2, E-STDN-s adopts an aggressive channel reduction strategy, reducing the dimensionality of each encoder stage by approximately 50% relative to the original STDN. Combined with the EfficientBlock design based on depthwise separable convolutions, this configuration achieves substantial parameter and computational savings.

To quantify this efficiency, consider a single EfficientBlock comprising a depthwise convolution (3×3) followed by a pointwise convolution (1×1). For an input-output channel pair $(C_{\text{in}}, C_{\text{out}})$, the parameters are

$$P_{\text{block}} = 9\cdot C_{\text{in}} + C_{\text{in}}\cdot C_{\text{out}}. \tag{15}$$

A standard 3×3 convolution requires $9\cdot C_{\text{in}}\cdot C_{\text{out}}$ parameters, so the separable block reduces parameter count per layer by a factor of

$$\text{Compression} = \frac{9C_{\text{in}}C_{\text{out}}}{9C_{\text{in}} + C_{\text{in}}C_{\text{out}}} = \frac{9C_{\text{out}}}{9 + C_{\text{out}}}, \tag{16}$$

which is roughly 8–9$\times$ for typical channel widths. Following the α^2 parameter scaling law in depthwise-separable CNNs [16], if channels are halved uniformly across the network ($C_{\text{s}} = 0.5\,C_{\text{STDN}}$), then since 1×1 pointwise convolutions dominate parameters, the total parameter count reduces approximately by

$$R_{\text{params}} \approx 1 - \left(\frac{C_{\text{s}}}{C_{\text{STDN}}}\right)^2 = 1 - (0.5)^2 = 0.75, \tag{17}$$

corresponding to a theoretical 75% saving in parameters and FLOPs.

Despite its compact design, E-STDN-s retains the inverted residual bottleneck structure and group normalization to preserve representational stability. These elements mitigate the loss of feature expressiveness, maintaining sensitivity to common spoof artifacts while trading off some capacity for subtle or novel attack patterns.

3.1.2. E-STDN-m (medium variant for balanced performance)

Building upon the efficiency-oriented design of E-STDN-s, the medium variant relaxes the compression ratio to restore representational capacity while retaining most architectural

simplifications. It targets scenarios where moderate computational increases are acceptable in exchange for improved robustness and discriminative strength.

Compared to E-STDN-s, E-STDN-m scales channel capacity by 1.5× across all encoder stages, reducing parameters theoretically by 44% within the generator relative to the original STDN. Although less compact, it offers richer intermediate representations. Since parameter count scales quadratically with channel width, this expansion results in approximately $(1.5)^2 = 2.25\times$ more parameters than E-STDN-s within the generator, reflecting a balanced performance–efficiency trade-off.

The expanded capacity enhances the network's ability to capture fine-grained textural and frequency-domain cues essential for detecting high-quality print and replay attacks exhibiting subtle anti-aliasing or localized spoof artifacts. Improved feature propagation through skip connections further benefits spatial localization in the decoder, particularly for partially manipulated regions. Consequently, E-STDN-m is suited for server-side or forensic deployments where accuracy is prioritized over latency.

3.1.3. Core architectural components

Both E-STDN-s and E-STDN-m share the same fundamental architectural design, consisting of an encoder-decoder structure with skip connections, a FastESR module for early spoof regression, and multi-scale discriminators for adversarial training. The only difference between the two variants lies in their channel configurations, as summarized in Table 2. The following part details the core components shared across both variants.

E-STDN adopts a symmetric encoder–decoder architecture with three downsampling and three upsampling stages to extract and reconstruct hierarchical features. Both encoder and decoder blocks are built upon EfficientBlocks using an inverted residual depthwise separable bottleneck (1×1 expansion, 3×3 depthwise convolution, and 1×1 projection), combined with group normalization and residual connections to ensure stable gradient propagation and high computational efficiency (achieving an 8–9× parameter reduction). Starting from a 3-channel input, the encoder progressively increases feature dimensions from 32 to 64, 128, and 256 channels while reducing spatial resolution to expand the receptive field. The decoder symmetrically restores spatial resolution through lightweight upsampling (e.g., bilinear interpolation) and feature refinement, with skip connections linking corresponding encoder–decoder stages to preserve fine-grained spatial details. This design enables accurate spoof trace localization while maintaining the efficiency required for mobile and real-world deployment.

The FastESR (Fast Early Spoof Regressor) module operates in parallel with the main encoder-decoder pipeline, extracting multi-scale features at three different resolutions. It receives feature maps of 64, 128, and 256 channels from the encoder stages, applies adaptive pooling operations to normalize spatial dimensions, concatenates these multi-scale representations, and processes them through additional convolutional layers to generate early spoof detection scores. This design enables the network to make preliminary spoof predictions using intermediate features, improving both accuracy and computational efficiency.

The multi-scale discriminator employs three discriminators operating at different image resolutions to capture spoof cues at multiple granularities. Each discriminator consists of stacked EfficientBlocks followed by pooling layers, producing both local and global discrimination scores. The EfficientBlocks adopt an inverted residual, depthwise separable

bottleneck (1×1 expansion, 3×3 depthwise convolution, and 1×1 projection) with residual connections, ensuring computational efficiency while preserving discriminative capacity. By jointly modeling fine-grained textures at high resolutions and global structural patterns at lower resolutions, the multi-scale design enhances robustness to resolution variation and improves real-world spoof detection.

3.2. Encoder-Decoder optimization

The encoder-decoder structure is optimized through channel reduction and improved skip connections. The channel configuration is systematically reduced while preserving feature representation capacity.

Table 2: Channel configuration comparison

Layer	STDN	E-STDN-s	E-STDN-m
Encoder Block 1	64	32	48
Encoder Block 2	128	64	96
Encoder Block 3	256	128	192
Bottleneck	512	256	384
Decoder Block 1	256	128	192
Decoder Block 2	128	64	96
Decoder Block 3	64	64	64

Moreover, the skip connections incorporate adaptive resizing to handle dimension mismatches

$$\mathbf{F}_{\text{skip}} = \text{Resize}(\mathbf{F}_{\text{encoder}}, \text{size}(\mathbf{F}_{\text{decoder}})). \tag{18}$$

This ensures proper feature fusion across different resolution levels while maintaining spatial information.

3.3. FastESR module

To reduce computational overhead while preserving spoof sensitivity, the Early Spoof Regressor is redesigned as a lightweight `FastESR` module. FastESR employs adaptive global pooling (`AdaptiveAvgPool2d(8)`) to compact spatial features, replaces standard convolutions with depthwise separable and 1×1 pointwise convolutions, and concatenates multi-scale features from three encoder layers (64, 128, and 256 channels), resulting in a 448-channel representation. These concatenated features are processed through the following operation

$$\mathbf{F}_{\text{concat}} = \text{Concat}(\text{Pool}(\mathbf{F}_1), \text{Pool}(\mathbf{F}_2), \text{Pool}(\mathbf{F}_3)). \tag{19}$$

The final spoof prediction is computed by averaging the output of FastESR with the spoof trace intensity

$$\text{score} = \frac{1}{2K^2}\|\mathbf{M}_{\text{FastESR}}\|_1 + \frac{\alpha_0}{2N^2}\|\mathbf{G}(\mathbf{I})\|_1, \tag{20}$$

where $K = 8$ represents the FastESR output resolution, $N = 256$ is the input image size, and α_0 is a weighting coefficient that balances the spoof intensity component.

3.4. Lightweight multi-scale discriminators

The multi-scale discriminator modules are restructured to be lightweight while preserving their ability to capture both local and global spoof cues. This is achieved by replacing conventional convolutional blocks with EfficientBlocks, which maintain representational power while using fewer parameters. Spatial resolution is quickly reduced using `AdaptiveAvgPool2d(4)`, enabling early dimensionality reduction. The channel configuration is simplified with a progressive setup: $32 \rightarrow 64 \rightarrow 96$, which provides sufficient capacity without redundancy. Additionally, shared architectural components are introduced across different scale-specific discriminators to reduce memory and computational redundancy.

4. EXPERIMENTS

4.1. Dataset

To evaluate the proposed E-STDN against conventional CNN models, a large-scale and well-annotated face anti-spoofing dataset is required, offering subject diversity, varied attack types, and reliable evaluation protocols. Among widely used benchmarks such as CASIA-MFSD, IDIAP-Replay, MSU-MFSD, OULU-NPU, SiW, SiW-M, and CelebA-Spoof, CelebA-Spoof is particularly suitable due to its large scale (over 10,000 subjects and 600,000 images), diverse spoofing attacks (print, cut, replay, and partial), and rich annotations including spoof region localization, attack categories, and environmental conditions. Its standardized evaluation protocols further support objective and reproducible benchmarking, reducing sensitivity to nuisance factors. Consequently, CelebA-Spoof is selected as the primary dataset for training and evaluating the proposed spoof trace disentanglement model to ensure robust generalization.

Table 3: Comparison of face anti-spoofing datasets

Dataset	Live/Spoof	Setup Variations	Attack Types
CASIA-MFSD [17]	150/450	3 image quality settings	Print (flat, wrapped, cut), Replay (tablet)
IDIAP-Replay [2]	200/1000	None	Print (flat), Replay (tablet, phone)
MSU-MFSD [18]	70/210	Indoor, 2 camera types	Print (flat), Replay (tablet, phone)
OULU-NPU [19]	990/3960	Indoor, 6 mobile devices	Print (flat), Replay (phone)
SiW [20]	1320/3300	Indoor, variations in distance, pose, illumination, and expression	Print (flat, wrapped), Replay (phone, tablet, monitor)
SiW-M [21]	660/968	Indoor, variations in distance, pose, illumination, and expression	Print (flat), Replay (phone, tablet, monitor), Mask (hard resin, plastic, silicone, paper, mannequin), Makeup (cosmetics, impersonation, obfuscation), Partial (glasses, cut paper)
CelebA-Spoof [22]	156384/469153	Indoor & outdoor, 4 lighting conditions, rich annotations	Print (flat, wrapped), Replay (phone, tablet, monitor), Mask (paper)

In this work, the CelebA-Spoof dataset is preprocessed using face cropping and landmark detection, resulting in a total of 552,945 images. The original training set is split into 60% for training (292,822 images) and 40% for validation (193,540 images), while the test set remains

unchanged with 66,583 images to ensure fair and consistent comparison with state-of-the-art methods.

4.2. Experimental setup

To provide a fair and consistent comparison between the proposed E-STDN architecture and conventional CNN-based models (VGG11, VGG16, ResNet18, ResNet50, DenseNet121, MobileNetV2), all models were trained under identical conditions using the same dataset splits, preprocessing, and training strategy.

4.2.1. Hardware and training configuration

All models were trained on a system equipped with an NVIDIA RTX 4060 GPU and an Intel Core i5 processor. Each training run spanned 25 epochs and employed mixed-precision (FP16) training with gradient scaling to accelerate computation and reduce memory usage. Standard data augmentation techniques, including random cropping, horizontal flipping, and color jittering, were applied to improve generalization. Model performance was validated after each epoch.

4.3. Evaluation metrics

The performance of classification models is assessed using a comprehensive set of evaluation metrics, including traditional classification metrics and specialized metrics for face anti-spoofing tasks. All metrics are computed based on the confusion matrix components: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN), where TP represents correctly classified positive samples, TN represents correctly classified negative samples, FP represents incorrectly classified positive samples (Type I error), and FN represents incorrectly classified negative samples (Type II error).

4.3.1. Equal Error Rate (EER)

Equal Error Rate (EER) is a widely used evaluation metric in biometric verification systems. It represents the error rate at which the False Acceptance Rate (FAR) and False Rejection Rate (FRR) are equal or nearly equal. The EER is computed at the optimal threshold t that minimizes the absolute difference between FAR and FRR

$$\mathrm{EER} = \frac{\mathrm{FAR}_t + \mathrm{FRR}_t}{2} \quad \text{and} \quad t = \arg\min_t |\mathrm{FAR}_t - \mathrm{FRR}_t|, \tag{21}$$

where

$$\mathrm{FAR}_t = \frac{FP_t}{FP_t + TN_t}, \tag{22}$$

$$\mathrm{FRR}_t = \frac{FN_t}{FN_t + TP_t}. \tag{23}$$

The EER is obtained from the Detection Error Tradeoff (DET) curve and provides a single-valued summary of the system's performance. A lower EER indicates better classification performance, as it implies a better balance between accepting genuine samples and rejecting impostors.

4.3.2. Average classification error rate (ACER)

ACER provides a balanced error metric by computing the arithmetic mean of False Acceptance Rate and False Rejection Rate

$$\text{ACER} = \frac{\text{FAR} + \text{FRR}}{2}, \tag{24}$$

ACER offers a comprehensive performance measure that equally weights both types of classification errors, making it particularly suitable for evaluating biometric systems where both false acceptances and false rejections are equally critical.

4.3.3. TDR@FDR=0.5(%)

This metric measures the True Detection Rate (TDR) when the False Discovery Rate (FDR) is constrained to 0.5

$$\text{TDR@FDR=0.5(\%)} = \text{TDR}|_{\text{FDR=0.5}}, \tag{25}$$

where

$$\text{TDR} = \frac{TP}{TP + FN} = \text{Recall}, \tag{26}$$

$$\text{FDR} = \frac{FP}{TP + FP} = 1 - \text{Precision}. \tag{27}$$

This metric evaluates the model's detection capability while maintaining a very low false discovery rate, which is crucial for applications requiring high precision with acceptable recall performance.

4.4. Experimental results

Due to computational constraints and hardware limitations, larger models were trained with reduced batch sizes and slightly elevated learning rates to accelerate convergence and enable comprehensive performance evaluation. The detailed training configurations and hyperparameters for each model are summarized in Table 4.

Table 4: Comparison of training configurations across different model architectures

Model	Parameters	Epochs	Batch Size	Training Time
VGG11	128M	25	1024	2h
VGG16	134M	25	1024	2h
ResNet18	11M	25	1024	2h
ResNet50	24M	25	1024	2h
DenseNet121	7M	25	1024	2h
MobileNetV2	2M	25	1024	2h
STDN	4M	15	8	16d
E-STDN-s	0.7M	15	8	8d
E-STDN-m	1.5M	15	8	16d

Consistent with our goal of architectural optimization, our experimental comparison focuses on: (1) the original STDN (to demonstrate the specific efficiency gains of our proposed

modifications), and (2) standard generic CNN backbones ranging from heavy (VGG, ResNet) to lightweight (MobileNet) architectures. This protocol is designed to explicitly highlight the trade-off between parameter efficiency and spoof trace disentanglement capability, justifying the practical value of E-STDN for resource-constrained applications.

Traditional CNN architectures such as VGG11 and VGG16 demonstrate substantial parameter overhead (128M and 134M respectively), inherently increasing overfitting susceptibility and computational requirements. ResNet and DenseNet architectures incorporate advanced techniques including skip connections and dense connectivity patterns to achieve parameter efficiency while preserving representational capacity. MobileNetV2 employs depthwise separable convolutions optimized for resource-constrained environments, achieving competitive performance with merely 2M parameters.

Task-specific architectures such as STDN and its E-STDN variants illustrate key computational trade-offs in face anti-spoofing. While STDN achieves strong performance with only 4M parameters through a multi-branch spatio-temporal design, its architectural complexity leads to long training time (16 days for 15 epochs). The proposed E-STDN variants mitigate this cost via architectural simplification: E-STDN-s reduces the model to 0.7M parameters and halves training time to 8 days, whereas E-STDN-m (1.5M parameters) preserves training cost while improving performance, highlighting the effectiveness of MobileNet-based components in balancing efficiency and task-specific representation.

To comprehensively evaluate the impact of these architectural modifications, we analyze the classification performance across all models using standard metrics. The fundamental classification performance across all evaluated models is presented in Table 5, revealing distinct performance characteristics for each architectural approach.

Table 5: Classification performance metrics across different model architectures

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
VGG11	85.6	97.7	81.4	88.8
VGG16	85.4	99.1	80.0	88.5
ResNet18	88.9	97.9	73.8	84.1
ResNet50	83.3	98.2	77.7	86.8
DenseNet121	84.2	97.4	79.6	87.6
MobileNetV2	84.1	98.0	79.0	87.5
STDN	95.6	97.8	95.9	96.8
E-STDN-s	90.3	96.5	89.4	92.8
E-STDN-m	95.0	98.1	94.8	96.4

The results show that STDN delivers the best overall performance (95.6% accuracy, 96.8% F1-Score) with a well-balanced precision–recall trade-off, while E-STDN-m closely matches this performance (95.0% accuracy, 96.4% F1-Score) using only 37.5% of STDN's parameters. The lightweight E-STDN-s further demonstrates strong parameter efficiency, achieving competitive results (90.3% accuracy, 92.8% F1-Score) with just 17.5% of the original model size. In contrast, conventional CNNs exhibit high precision but markedly lower recall, leading to inferior F1-Scores and limited effectiveness in security-critical face anti-spoofing scenarios.

To comprehensively assess the models' suitability for face anti-spoofing deployment, we

evaluate performance using specialized metrics including ACER, EER, and TDR@FDR=0.5(%), as presented in Table 6.

Table 6: Task-specific performance metrics for face anti-spoofing evaluation

Model	ACER (%)	EER (%)	TDR@FDR=0.5 (%)	Inference Time
VGG11	11.6	65.9	45.0	0.89ms
VGG16	10.9	71.5	43.0	0.91ms
ResNet18	15.0	68.8	21.1	0.89ms
ResNet50	12.8	68.3	8.9	0.89ms
DenseNet121	12.7	61.9	25.5	0.96ms
MobileNetV2	12.4	65.4	37.4	0.89ms
STDN	4.6	4.3	98.2	62ms
E-STDN-s	9.1	9.9	93.5	15ms
E-STDN-m	4.8	4.9	97.8	28ms

Task-specific evaluation highlights a clear gap between general-purpose and specialized architectures under strict security constraints. Conventional CNNs perform poorly at TDR@FDR=0.5(%), with detection rates below 50%, whereas STDN and its variants remain robust. STDN achieves 98.2% TDR, E-STDN-m closely follows at 97.8% with reduced complexity, and the compact E-STDN-s still attains 93.5% TDR using only 0.7M parameters. Overall, task-specific models more than double the detection capability of conventional CNNs under security-critical operating conditions.

The computational efficiency analysis highlights clear advantages of the proposed E-STDN variants over the original STDN. E-STDN-s achieves 15 ms inference time, corresponding to a $\sim$4$\times$ speedup over STDN (62 ms), while maintaining competitive spoof detection performance (93.5% TDR). E-STDN-m offers a more balanced trade-off, achieving near-identical security performance to STDN (97.8% vs. 98.2% TDR) with a 2.2$\times$ inference speedup (28 ms vs. 62 ms), making it suitable for applications requiring both high security and low latency. Notably, the observed latency reduction from 62 ms to 15 ms represents a $\approx$75.8% improvement, closely matching the theoretical 75% parameter reduction predicted by the channel scaling strategy ($1 - 0.5^2$) in Eq. (17). This strong empirical-theoretical agreement confirms that the task-specific E-STDN design, leveraging depthwise separable convolutions, effectively translates architectural compression into real-world performance gains. Overall, E-STDN-s achieves an 82.5% parameter reduction (4M$\rightarrow$0.7M), 4.13$\times$ inference acceleration, and 50% training time reduction, while E-STDN-m preserves near-optimal performance with a 62.5% parameter reduction, positioning the two variants favorably on the Pareto frontier of efficiency versus security.

These results demonstrate that task-specific architectural optimization can achieve near-optimal security performance while meeting stringent computational constraints of practical deployment scenarios, establishing E-STDN as a foundation for next-generation security systems across mobile devices, edge computing systems, and embedded security applications.

4.5. Ablation study: The critical role of FastESR

To rigorously evaluate the contribution of the FastESR module to the overall architectural stability, we performed an ablation study using the E-STDN-m variant. We selected the

Medium variant as a representative benchmark because its deeper structure imposes higher optimization challenges, making it an ideal candidate to test the efficacy of our design choices. Theoretically, the final prediction score is a fusion of the coarse-grained FastESR output ($\mathbf{M}$) and the fine-grained spoof trace ($\mathbf{G}(\mathbf{I})$)

$$\text{score} = \underbrace{\frac{1}{2K^2}\|\mathbf{M}\|_1}_{\text{FastESR term}} + \underbrace{\frac{\alpha_0}{2N^2}\|\mathbf{G}(\mathbf{I})\|_1}_{\text{Trace term}}. \tag{28}$$

In the "w/o FastESR" configuration, the first term is eliminated, forcing the network to rely solely on the trace reconstruction for supervision.

Table 7: Ablation study of FastESR module of E-STDN on CelebA-Spoof

Model	Config	ACER (%)	EER (%)	TDR@0.5 (%)
E-STDN-m	w/o FastESR	80.60	93.03	4.89
	w/ FastESR	**4.80**	**4.90**	**97.80**

The results in Table 7 highlight the importance of the FastESR module within the E-STDN-m architecture. Removing FastESR leads to a severe degradation in performance, with ACER increasing to 80.60% and TDR@0.5 dropping to 4.89%, indicating a failure to learn discriminative representations. This suggests that, for a medium-scale backbone such as E-STDN-m, the trace reconstruction objective alone is insufficient to provide effective optimization signals during training. The FastESR module serves as an auxiliary supervision branch that facilitates gradient propagation to the encoder, thereby stabilizing the training process. As a result, FastESR plays a crucial role in enabling the successful optimization of the E-STDN-m model.

5. CONCLUSION

This paper presents E-STDN, an efficient and practical approach to face anti-spoofing that balances security performance with computational efficiency. By integrating lightweight architectural designs such as depthwise separable convolutions and optimized normalization, E-STDN overcomes the limitations of conventional CNN models in security-critical scenarios.

The proposed architecture maintains strong spoof detection capabilities while considerably reducing computational complexity, enabling flexible deployment across a range of environments, from mobile devices to edge computing systems. E-STDN establishes a new paradigm in anti-spoofing system design, demonstrating that task-specific model optimization can achieve robust real-world performance without compromising efficiency.

ACKNOWLEDGMENT

This work has been supported by Academy of Cryptography Techniques.

REFERENCES

[1] LivDet-Face Organizers, "Face liveness detection competition (livdet-face) 2021," https://livdet.org/face2021-livdet-org/face_2021.pdf, 2021, accessed: June 16, 2025.

[2] I. Chingovska, A. Anjos, and S. Marcel, "On the effectiveness of local binary patterns in face anti-spoofing," in *2012 BIOSIG – Proceedings of the International Conference of the Biometrics Special Interest Group (BIOSIG)*. Darmstadt, Germany: IEEE, Sep. 2012, pp. 1–7, idiap Replay-Attack Database. [Online]. Available: https://ieeexplore.ieee.org/document/6313548

[3] I. Manjani, S. Tariyal, M. Vatsa, R. Singh, and A. Majumdar, "Detecting silicone mask-based presentation attack via deep dictionary learning," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 7, pp. 1713–1723, 2017. [Online]. Available: https://doi.org/10.1109/TIFS.2017.2676720

[4] A. Agarwal, R. Singh, and M. Vatsa, "Face anti-spoofing using haralick features," in *2016 IEEE 8th International Conference on Biometrics Theory, Applications and Systems (BTAS)*. IEEE, 2016, pp. 1–6. [Online]. Available: https://doi.org/10.1109/BTAS.2016.7791171

[5] Antispoofing Wiki, "Presentation attacks: Types, instruments and detection," https://antispoofing.org/presentation-attacks-types-instruments-and-detection/, 2021, accessed: June 16, 2025.

[6] Y. Liu, J. Stehouwer, and X. Liu, "On disentangling spoof trace for generic face anti-spoofing," in *Computer Vision – ECCV 2020*, ser. Lecture Notes in Computer Science, vol. 12363. Springer, 2020. [Online]. Available: https://doi.org/10.1007/978-3-030-58523-5_24

[7] Z. Yu, Y. Qin, X. Li, C. Zhao, Z. Lei, and G. Zhao, "Deep learning for face anti-spoofing: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 5, pp. 5609–5631, 2023. [Online]. Available: https://doi.org/10.1109/TPAMI.2022.3215850

[8] T. H. M. Siddique, S. Khan, Z. Wang, and K. Huang, "AdvSpoofGuard: Optimal transport driven robust face presentation attack detection system," *Knowledge-Based Systems*, 2025. [Online]. Available: https://doi.org/10.1016/j.knosys.2025.113759

[9] H. Xing, S. Y. Tan, F. Qamar, and Y. Jiao, "Face anti-spoofing based on deep learning: A comprehensive survey," *Applied Sciences*, vol. 15, no. 12, p. 6891, 2025. [Online]. Available: https://doi.org/10.3390/app15126891

[10] P.-K. Huang, C.-H. Chiang, T.-H. Chen, J.-X. Chong, T.-L. Liu, and C.-T. Hsu, "One-class face anti-spoofing via spoof cue map-guided feature learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 277–286. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2024/html/Huang_One-Class_Face_Anti-spoofing_via_Spoof_Cue_Map-Guided_Feature_Learning_CVPR_2024_paper.html

[11] X. Lin, S. Wang, R. Cai, Y. Liu, Y. Fu, W. Tang, Z. Yu, and A. Kot, "Suppress and rebalance: Towards generalized multi-modal face anti-spoofing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 211–221. [Online]. Available: https://doi.org/10.1109/CVPR52733.2024.00028

[12] X. He, Y. Li, D. He, R. Yuan, and L. Huang, "Lightweight face anti-spoofing method based on cross-fused multi-modal features," *Journal of Electronic Imaging*, vol. 33, no. 2, p. 023033, 2024. [Online]. Available: https://doi.org/10.1117/1.JEI.33.2.023033

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR), Poster Track*, San Diego, CA, USA, 2015. [Online]. Available: https://dblp.org/rec/journals/corr/KingmaB14.html

[14] Y. Liu, A. Jourabloo, W. Ren, and X. Liu, "Dense face alignment," in *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW)*, Venice, Italy, 2017, pp. 1619–1628. [Online]. Available: https://doi.org/10.1109/ICCVW.2017.190

[15] Y. Wu and K. He, "Group normalization," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 3–19. [Online]. Available: https://doi.org/10.1007/978-3-030-01261-8_1

[16] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520. [Online]. Available: https://doi.org/10.1109/CVPR.2018.00474

[17] Z. Zhang, J. Yan, S. Liu, Z. Lei, D. Yi, and S. Z. Li, "A face antispoofing database with diverse attacks," in *2012 5th IAPR International Conference on Biometrics (ICB)*. IEEE, 2012, pp. 26–31, cASIA-MFSD (a.k.a. CASIA-FASD) Database. [Online]. Available: https://doi.org/10.1109/ICB.2012.6199754

[18] D. Wen, H. Han, and A. K. Jain, "Face spoof detection with image distortion analysis," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 4, pp. 746–761, 2015, mSU-MFSD Database. [Online]. Available: https://doi.org/10.1109/TIFS.2015.2400395

[19] Z. Boulkenafet, J. Komulainen, L. Li, X. Feng, and A. Hadid, "OULU-NPU: A mobile face presentation attack database with real-world variations," in *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*. IEEE, 2017, pp. 612–618, oULU-NPU Database. [Online]. Available: https://doi.org/10.1109/FG.2017.77

[20] Y. Liu, A. Jourabloo, and X. Liu, "Learning deep models for face anti-spoofing: Binary or auxiliary supervision," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 389–398, siW Database. [Online]. Available: https://doi.org/10.1109/CVPR.2018.00048

[21] Y. Liu, J. Stehouwer, A. Jourabloo, and X. Liu, "Deep tree learning for zero-shot face anti-spoofing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4680–4689, siW-M Database. [Online]. Available: https://doi.org/10.1109/CVPR.2019.00481

[22] Y. Zhang, Z. Yin, Y. Li, G. Yin, J. Yan, J. Shao, and Z. Liu, "CelebA-Spoof: Large-scale face anti-spoofing dataset with rich annotations," in *Computer Vision – ECCV 2020*, ser. Lecture Notes in Computer Science, vol. 12357. Springer, 2020, pp. 70–85. [Online]. Available: https://doi.org/10.1007/978-3-030-58610-2_5

Received on October 07, 2025
Accepted on January 19, 2026