Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

Lightweight Hybrid CNN-Transformer Architecture for Image-Based Deepfake Detection on Resource-Constrained Systems: Cross-Dataset Generalization and OpenVINO-Based Benchmarking

Vishvas[1]

[1]Department of Computer Science & Engineering, Amritsar Group of Colleges, Amritsar, Punjab, India

Abstract* — Deepfake detection research is typically evaluated on accuracy alone, with little attention to deployment cost on consumer-grade hardware. This paper proposes a lightweight hybrid architecture that pairs a MobileNetV3-Small backbone with a two-layer Transformer encoder for image-level deepfake detection. The model is evaluated using video-level partitioning on FaceForensics++ and Celeb-DF v2, with dataset-specific identity-overlap limitations explicitly considered. On the official FaceForensics++ split, it reaches 80.33% accuracy and 0.891 ROC-AUC; on Celeb-DF v2, 81.59% accuracy and 0.910 ROC-AUC. A depth ablation spanning zero to four Transformer layers, together with EfficientNet-B0 and MobileViT-XXS baselines trained under an identical protocol, shows that same-dataset accuracy does not meaningfully separate the six architectures. Cross-dataset evaluation does: the proposed model achieves the highest generalization AUC (0.670) of all six, including against EfficientNet-B0, which uses 84% more parameters. Converting the model to OpenVINO FP16 costs negligible accuracy and is the recommended deployment precision; INT8 quantization substantially reduces classification performance and did not show a consistent latency advantage over FP16 under the evaluated configuration. End-to-end pipeline measurement, including face detection, shows that detection rather than the classifier accounts for 89–91% of total latency, limiting real-world throughput to 12–19 frames per second under FP16 regardless of how the classifier itself is optimized. Taken together, these results indicate that parameter count and same-dataset accuracy are incomplete proxies for deployment readiness, and that detection-stage latency and hardware-specific quantization behavior both warrant more attention in lightweight deepfake-detection research than they currently receive.

***Keywords* —** *Deepfake detection; hybrid CNN-Transformer; lightweight neural networks; OpenVINO; model quantization; cross-dataset generalization*

I. INTRODUCTION

Generative models based on GANs and diffusion processes can now produce facial manipulations convincing enough to pass casual visual inspection, and the technical barrier to producing them has fallen sharply over the past several years. This has created a practical problem for platforms, financial institutions, and verification systems that need to tell real media from manipulated media, often at the point of upload or capture rather than after the fact.

Detection research has moved through three broad phases. Convolutional networks were the first effective approach, since manipulation artifacts tend to show up as local texture and blending inconsistencies that convolutional filters are well suited to catching [1, 3]. Vision Transformers followed, replacing local convolution with self-attention to model relationships across an entire image rather than within a fixed receptive field [6]. More recently, hybrid CNN-Transformer architectures have combined both mechanisms, using convolution for local feature extraction and attention for global context [7, 8, 10, 11, 12, 13].

Almost all of this work is evaluated for accuracy on GPU hardware, with little attention to what happens when the same model needs to run on a laptop CPU or an edge device with no dedicated accelerator. The gap matters in practice: a detector that cannot run in real time on the hardware actually available to a platform or an end user is of limited use regardless of its benchmark accuracy. Hardware-aware optimization frameworks such as Intel's OpenVINO and NNCF, building on a broader line of work in network pruning and quantization [30, 31], exist specifically to close this gap through graph-level optimization and post-training quantization [5, 14]. Few deepfake-detection papers report results after applying them, and fewer still measure the full inference pipeline rather than the classifier alone. Intel Extension for PyTorch (IPEX), once a complementary optimization layer for this toolchain, reached its planned end of life in March 2026, with Intel directing users to native PyTorch [15, 17]; this study therefore uses native PyTorch together with OpenVINO and NNCF, without IPEX, in line with that guidance.

A second gap concerns how detection performance is actually assessed. Reported accuracy figures in this literature are frequently obtained from random frame- or image-level splits of video datasets, which allow frames from the same source video — and in FaceForensics++ specifically, the same identity — to appear in both training and evaluation [2]. The same risk extends to other widely used benchmarks, including the DeepFake Detection Challenge dataset [27] and DeeperForensics-1.0 [28], and it inflates reported accuracy

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

relative to what a model achieves on genuinely unseen videos and identities.

A third, closely related issue is that detectors reporting strong within-dataset accuracy routinely fail to generalize once training and test data come from different sources or different manipulation techniques. This generalization gap has been documented repeatedly and remains, by several recent accounts, largely unresolved [18–23]; a detector's practical value depends on how it performs on manipulation types it has not seen, not only on the benchmark it was tuned against.

This paper addresses these three gaps directly. I propose a lightweight hybrid CNN-Transformer architecture, built from a MobileNetV3-Small backbone and a shallow Transformer encoder, for single-image deepfake detection. I evaluate it under the official FaceForensics++ and Celeb-DF v2 splits, which keep videos and, to the extent each benchmark's protocol allows, identities separated between training and test data. I compare it against three depth variants of its own Transformer component and against two independent baseline architectures, EfficientNet-B0 [24] and MobileViT-XXS [7], trained under an identical protocol, and I test cross-dataset generalization in both directions between FaceForensics++ and Celeb-DF v2. Finally, I convert the model to OpenVINO's FP16 and INT8 representations and measure not only classifier inference time but the latency of the full detection pipeline, including face localization, on CPU hardware.

The contributions of this paper are as follows:

- A lightweight hybrid CNN-Transformer architecture for image-level deepfake detection, evaluated under a dataset-specific video-level train/validation/test protocol, with identity separation maintained for FaceForensics++ but not for Celeb-DF v2 on two standard benchmarks.
- A depth ablation across the Transformer component (0 to 4 layers) and a comparison against two independent CNN and hybrid baseline architectures trained under an identical data, split, and optimization protocol.
- A bidirectional cross-dataset evaluation between FaceForensics++ and Celeb-DF v2, showing that the proposed architecture achieves the highest observed AUC among the six evaluated models in the FaceForensics++ to Celeb-DF v2 direction, while also documenting substantial generalization limitations in the reverse direction, including a baseline with 84% more parameters.
- A hardware-aware deployment benchmark on Intel CPU hardware using OpenVINO FP32/FP16/INT8 conversion, including an end-to-end pipeline measurement that identifies face detection, rather than the classifier, as the dominant source of inference latency.

II. RELATED WORK

A. Benchmark Datasets

Two video-based benchmarks anchor most reported results in this area, and both are used directly in this paper's experiments. FaceForensics++ [2] pairs real footage with four manipulation techniques — DeepFakes, Face2Face, FaceSwap, and NeuralTextures. Celeb-DF v2 [4] was built specifically to reduce the visible blending artifacts that made earlier face-swap datasets easy to spot. The DeepFake Detection Challenge dataset [27] and DeeperForensics-1.0 [28] extend the same video-based design at a larger scale, though neither is used directly in this paper's experiments. These video datasets can be susceptible to identity leakage when frame-level or inadequately controlled video-level partitioning is used, although the extent of this risk depends on the dataset and split protocol. A naive frame-level split may allow a model to learn identity-specific characteristics rather than manipulation artifacts, inflating performance on evaluation data containing familiar identities, inflating accuracy relative to what the same model would score on a genuinely unseen person. Altuncu et al.'s meta-review [9] documents how inconsistently "deepfake detection" gets defined and measured across this literature, and that inconsistency recurs below: dataset choice and split discipline affect reported numbers at least as much as architecture choice does.

B. CNN-Based Detection

Xception [1] remains a standard reference point in this space, mostly because it was the detector used in the original FaceForensics++ paper [2] and so shows up in nearly every comparison table that followed. Its depthwise separable convolutions give it a smaller parameter count than earlier CNNs of comparable depth, though not dramatically smaller than contemporaries like InceptionV3, a distinction worth noting because later papers sometimes treat Xception as inherently lightweight when its original publication makes no such claim. MobileNetV3 [3] and the broader MobileNet family [26, 29] take efficiency as the explicit design goal, using hardware-aware architecture search and squeeze-and-excitation blocks to cut computation for generic vision tasks. None of these three papers reports a deepfake result. Their relevance here is architectural: whether an efficient general-purpose backbone transfers well to the texture and blending cues that mark a manipulated face is a separate, empirical question.

C. Vision Transformer-Based Detection

The original ViT [6] replaces convolution with patch-level self-attention, letting a model relate distant regions of an image without the layer stacking a CNN needs to grow its receptive field. DeiT [25] shows that the same architecture can be trained without ViT's original data-hungry regime, which matters for any deepfake dataset smaller than ImageNet scale. Within deepfake detection specifically, Luo et al.'s Forgery-Aware Adaptive ViT [10] freezes a pretrained ViT backbone and adds trainable adaptation

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

modules at global, local, and fine-grained scales, reporting solid cross-dataset transfer. That cross-dataset number is a genuinely useful data point. It also says nothing about inference cost: the paper reports no CPU latency, no quantized variant, and no integrated-GPU benchmark, so a transformer that generalizes well on paper may still be impractical to run on constrained hardware.

D. Hybrid CNN-Transformer Architectures

The architectures closest to this paper's own design combine convolution and attention rather than picking one. MobileViT [7] interleaves convolutional and transformer blocks for general mobile vision. It is not a deepfake detector, but its central claim, that local and global feature extraction can share one lightweight model, motivates the architecture used here. Within deepfake detection, Coccomini et al. [8] pair an EfficientNet-B0 extractor with a ViT variant on individual face crops and aggregate predictions across a video by voting. The method does not rely on an LSTM, and the authors make no temporal-modeling claim. Petmezas et al. [11] take the opposite route, building an explicitly temporal pipeline out of 3D morphable models, a CNN, an LSTM, and a transformer for video-level identity verification. They report a speed improvement over a comparison system but note their timing excludes preprocessing, a detail easy to miss on a quick read of the abstract. UAM-Net [12] combines EfficientNet with spatial and channel attention rather than a conventional transformer encoder, and its authors state directly that low-power deployment was outside the paper's scope. The architecture nearest to this work is Rani et al.'s ViT-Linformer hybrid [13], the base paper this study builds on: it reports training time, accuracy, and training FLOPs on a workstation with a discrete GPU, numbers that do not translate into CPU inference latency no matter how favorably they read. These five papers together establish that CNN-transformer hybrids for deepfake detection are not a new idea. What none of them do is apply hardware-aware post-training optimization or report the CPU deployment numbers that determine whether a detector is usable outside a GPU server. Rani et al. [13] report 98.9% accuracy, substantially higher than the figures in Table 1. Their publication does not disclose a video- or identity-level split protocol; given that this study's own experiments show random, leakage-prone splits can inflate reported accuracy by 10 points or more relative to an identity-aware protocol, the two figures are not directly comparable.

E. Hardware-Aware Optimization

Compression and quantization have their own, separate literature. Han et al.'s pruning and Huffman-coding pipeline [30] and Jacob et al.'s integer-arithmetic quantization scheme [31] both predate the deepfake-detection work above and were not written for it. Kozlov et al.'s NNCF [5] packages quantization, sparsity, and pruning into a single compression framework, and Intel's OpenVINO toolkit [14] converts and runs the resulting models on supported Intel hardware. Intel Extension for PyTorch (IPEX) [15] once sat alongside this toolchain as an additional optimization layer, but Intel discontinued active development and pointed users to native PyTorch as the extension reached its planned end of life in March 2026 [17]. That detail matters here only because it rules IPEX out of this paper's own deployment pipeline, which uses OpenVINO and NNCF on native PyTorch instead. None of the five hybrid architectures above, and none of the compression papers themselves, connects the two literatures. To the best of my knowledge, none of the hybrid architectures reviewed above applies hardware-aware post-training optimization and reports both accuracy and CPU latency together.

F. Cross-Dataset Generalization

A separate line of work asks a question that cuts across both architecture and deployment cost: does a detector trained on one dataset or manipulation technique hold up on another? The answer, reported consistently across several recent studies [18–23], is mostly no. Accuracy that looks strong within one dataset routinely collapses under cross-dataset or cross-manipulation evaluation, a pattern that shows up across different model families and dataset pairs rather than being specific to any one architecture. This paper treats that gap as an established property of the field, not something it sets out to solve. Its relevance here is that any cross-dataset number reported later in this paper should be read against that backdrop, not as a surprise specific to the proposed model.

G. Research Gap

Three threads run through the sections above without meeting. The deepfake-detection literature has produced several CNN-transformer hybrids [7, 8, 10, 11, 12, 13], evaluated for accuracy and, occasionally, cross-dataset transfer, but almost never for CPU-deployable latency. The hardware-optimization literature [5, 14, 15, 17, 30, 31] builds the tools that would answer that question but has not been applied to any of the hybrid architectures above. The generalization literature [18–23] shows that same-dataset accuracy is an unreliable signal on its own, which argues for testing any new hybrid cross-dataset before trusting its headline number — something the hybrid papers above do only inconsistently, and the base paper for this work [13] does not do at all. This paper sits at that intersection. It evaluates a CNN-transformer hybrid under a video-disjoint, cross-dataset protocol (identity-disjoint for FaceForensics++, though not for Celeb-DF v2, as Section III-B details) and benchmarks it end-to-end rather than model-only, on OpenVINO-optimized CPU hardware.

III. PROPOSED METHODOLOGY

A. Architecture

The proposed model pairs a MobileNetV3-Small backbone [3] with a shallow Transformer encoder. The backbone produces a 576-channel, 7×7 spatial feature map from a 224×224 face crop; a 1×1 convolution projects this down to the encoder's embedding dimension. The resulting 49 spatial positions are treated as patch tokens, in the same

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

spirit as the original Vision Transformer [6], with a learnable classification token and positional embeddings added before the tokens pass through the Transformer encoder. The encoder's output at the classification token position feeds a small multilayer perceptron head that produces the real/fake decision.

The design intent follows directly from the gap identified in Section II-D: convolution for the local texture and blending cues that mark a manipulated face, attention for context that spans the whole face rather than a single receptive field, and a parameter budget small enough to run on CPU hardware without a dedicated accelerator. MobileViT [7] demonstrates that convolution and attention can share one lightweight model for general vision tasks; this architecture tests whether the same principle holds for deepfake detection specifically, which none of the five hybrid detectors reviewed in Section II-D establish directly, since each was evaluated on different data, hardware, or both.

The depth of the Transformer encoder is treated as a free parameter rather than fixed by assumption. Section V reports results for 0 (no Transformer, a convolution-only ablation), 1, 2, and 4 encoder layers, trained and evaluated under an identical protocol, so that the contribution of the attention mechanism itself — rather than of the architecture family in general — can be isolated.

B. Dataset Protocol and Leakage Control

Two datasets are used: FaceForensics++ [2] and Celeb-DF v2 [4], both reviewed in Section II-A. Both are video datasets; faces are extracted from individual frames using MTCNN [16] and treated as independent images for training and evaluation, consistent with this paper's image-level scope.

The leakage risk described in Section II-A — frames or identities from one video appearing on both sides of a split — is addressed at the video-level, not the frame level. For FaceForensics++, the official train, validation, and test video lists are used as published, rather than drawing a fresh random split from the pooled frame set. For Celeb-DF v2, the official test list is used as published; training and validation videos are then drawn at random from the remaining videos, since the benchmark's own release does not define a corresponding train/validation split. This difference matters for how the resulting numbers should be read. The FaceForensics++ protocol is identity-disjoint by construction, since its official splits were built with that property in mind. The Celeb-DF v2 protocol is video-disjoint but not identity-disjoint: because its fake videos reuse the same pool of subjects across many source-target pairs, a large majority of the training videos share an identity with at least one test video, even though no training video is itself a test video. Section V reports this overlap directly rather than treating the two protocols as equivalent, and treats the cross-dataset results in Section III-E as the stronger evidence for generalization precisely because they carry no such overlap.

C. Training and Model-Selection Protocol

Every model — the proposed architecture at each Transformer depth, and both baseline architectures — is trained from ImageNet-pretrained weights rather than warm-started from a checkpoint trained on a different split. This choice costs some convergence speed but removes a specific leakage path: a checkpoint already trained on one random split could have seen videos that a later official split places in validation or test.

Backbone parameters are optimized at a lower learning rate than the newly added Transformer and classification-head parameters, on the reasoning that the backbone arrives already adapted to natural images and needs smaller updates than the randomly initialized layers on top of it. A cosine-annealed schedule and label smoothing are applied uniformly. The best epoch is selected by validation ROC-AUC rather than validation accuracy, since AUC does not depend on a decision threshold and proved less volatile across epochs during early experimentation; training stops early once validation AUC fails to improve for a fixed number of epochs. This recipe is held fixed across every architecture and depth compared in Section V, so that differences in reported results reflect the architectures themselves rather than differences in how each was tuned.

D. Evaluation Protocol

Each test set is evaluated exactly once per trained model. Any hyperparameter choice, threshold choice, or architecture comparison is made using only the validation split; the test split is touched a single time, after those choices are finalized, and the resulting number is reported as final regardless of whether it matches expectation.

Two decision thresholds are considered for every model: the default of 0.5, and a threshold chosen to maximize Youden's J statistic on that model's own validation set. Both are reported, but the default threshold is treated as primary throughout this paper, because the validation-tuned threshold was found, repeatedly, not to transfer reliably to the test set — in several cases performing worse than the untuned default once applied to genuinely unseen data. This instability is reported as a finding in its own right in Section V, not concealed by quietly preferring whichever threshold produces the larger number.

Confidence intervals are computed by bootstrap resampling at the video-level, not the frame level: whole videos, rather than individual frames, are resampled with replacement, since frames from the same video are not independent observations and frame-level resampling would understate the true uncertainty. Results are also broken down by manipulation method where a dataset's labeling supports it. One methodological detail is worth stating plainly: these per-method breakdowns use each model's validation-tuned threshold, for consistency with how the breakdown was computed during development, while the headline accuracy and AUC figures use the default threshold discussed above. The two are not always directly comparable as a result, and

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

Section V notes this explicitly wherever it could otherwise be misread. For datasets with an uneven ratio of real to manipulated test videos, balanced accuracy is reported alongside plain accuracy, since plain accuracy alone can reward a model that defaults toward the majority class.

E. Cross-Dataset Evaluation

Beyond within-dataset testing, each trained model is evaluated on the other dataset's official test set: a model trained on FaceForensics++ is tested on the Celeb-DF v2 test set, and a model trained on Celeb-DF v2 is tested on the FaceForensics++ test set. The decision threshold for this evaluation is the one selected on the training dataset's own validation set; no label from the target dataset is used at any point in selecting it, since a deployed detector would not have access to labeled examples from every dataset it might encounter in practice. This evaluation is treated as the primary test of generalization in this paper, for the reason given in Section III-B: training and test data for this evaluation come from separate benchmark datasets, so no video overlap is possible by construction. Identity overlap between the two datasets' subjects was not systematically verified, and this evaluation should be read as dataset-disjoint rather than confirmed identity-disjoint.

F. Hardware-Aware Optimization Pipeline

After training, each model is converted from its native PyTorch representation to OpenVINO Intermediate Representation [14] at both FP32 and FP16 precision. Post-training INT8 quantization is then applied using NNCF [5], calibrated on a sample of images drawn from the training split only, so that no validation or test data influences the quantization step. All three precisions are benchmarked for classifier-only inference latency on CPU. Separately, the full detection pipeline — frame capture, MTCNN face localization, preprocessing, and classification — is timed end-to-end, since classifier latency alone, as argued in Section II-E, does not establish whether a detector is actually usable at the frame rate a deployment would require.

IV. EXPERIMENTAL SETUP

A. Hardware and Software

All training, evaluation, and CPU benchmarking were performed on an Intel Core i5-1334U (13th Gen), running Windows inside a Python 3.11.9 virtual environment. PyTorch 2.14.0 and torchvision 0.29.0 (both CPU builds) were used for model training; OpenVINO 2026.2.0 and NNCF 3.3.0 for post-training optimization; timm 1.0.30 for the MobileViT-XXS baseline. Face extraction used facenet-pytorch's MTCNN implementation [16]. All CPU benchmarking was performed with the system's power plan set to Best Performance, with no other compute-intensive process running concurrently; latency measurements were repeated after an idle period when an initial run showed signs of thermal throttling, discussed further in Section V-F. AI-assisted tools were used in preparing this manuscript; see Declarations for the full disclosure.

B. Datasets and Splits

FaceForensics++. The official train, validation, and test video lists [2] were used directly, rather than a random split of the pooled frame set. The full official training set (360 of 360 video pairs) was used, together with the official validation (30 pairs) and test (40 pairs) lists. After resolving balance across manipulation methods, this yielded 1,440 training videos, 120 validation videos, and 160 test videos, split evenly between real and manipulated content at each stage.

Celeb-DF v2. The official test list [4] (340 manipulated, 178 real videos) was used as published. No official train/validation split is defined for this benchmark, so training (300 real, 300 manipulated) and validation (60 real, 60 manipulated) videos were drawn at random from the remaining pool, under a fixed seed. As noted in Section III-B, this makes the Celeb-DF v2 training set video-disjoint but not identity-disjoint from its test set, since the dataset's manipulated videos are generated from a comparatively small, reused pool of source identities.

Face extraction. Faces were extracted from every video using MTCNN [16] (output size 224×224, margin 20 pixels, single largest face per frame), sampling every 20th frame and capping extraction at 15 faces per video for FaceForensics++ and 10 faces per video for Celeb-DF v2, to keep per-video contribution roughly bounded. This produced 21,600 training, 1,800 validation, and 2,400 test images for FaceForensics++, and 5,986 training, 1,199 validation, and 5,156 test images for Celeb-DF v2.

C. Training Configuration

Every model reported in this paper — the proposed architecture at each Transformer depth (0, 1, 2, 4 layers), EfficientNet-B0, and MobileViT-XXS — was trained under an identical recipe: AdamW optimization with a differential learning rate (1×10^{-4} for newly initialized layers, 1×10^{-5} for the pretrained backbone), weight decay of 1×10^{-2}, label smoothing of 0.1, and a cosine-annealed learning-rate schedule. Training ran for up to 6 epochs with early stopping if validation ROC-AUC failed to improve, selecting the checkpoint with the highest validation AUC rather than the final epoch. All models were trained from ImageNet-pretrained weights, for the reason given in Section III-C: no model was warm-started from a checkpoint trained on a different data split. A batch size of 32 and an input resolution of 224×224 were used throughout. For the proposed architecture, the Transformer encoder used an embedding dimension of 256 and 4 attention heads; parameter counts for all six architectures compared are reported in Section V.

D. Baseline Architectures

EfficientNet-B0 [24] was initialized from torchvision's ImageNet-pretrained weights, with its classifier head replaced for binary classification; it was trained under the identical recipe described above with no architectural modification beyond the head. MobileViT-XXS [7] was

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

initialized from timm's ImageNet-pretrained weights (mobilevit_xxs.cvnets_in1k) under the same recipe. A DeiT-tiny [25] baseline was planned but not completed within the available compute and time budget; it is instead discussed in Section II-C and Section V by reference to its published results, with the resulting protocol difference stated explicitly rather than treated as a direct comparison.

E. Evaluation Procedure

For every trained model, the validation split was used to select the best training epoch (by ROC-AUC) and to compute a decision threshold maximizing Youden's J statistic. The corresponding test split — the official FaceForensics++ test set, the official Celeb-DF v2 test set, or the opposite dataset's test set for the cross-dataset evaluation described in Section III-E — was then evaluated exactly once, after these choices were finalized. Accuracy, precision, recall, F1-score, and ROC-AUC are reported at the default 0.5 threshold as the primary figures, with the validation-tuned threshold reported alongside for comparison; Section V discusses cases where the two disagree substantially. Video-level 95% confidence intervals were computed by bootstrap resampling (1,000 resamples) at the video-level. Where a dataset's test distribution is imbalanced between real and manipulated videos — as is the case for the Celeb-DF v2 official test list — balanced accuracy is reported alongside plain accuracy.

F. Hardware-Aware Optimization

Each trained checkpoint was converted from PyTorch to OpenVINO Intermediate Representation at FP32 and FP16 precision, and separately quantized to INT8 using NNCF, with calibration images drawn exclusively from that model's training split [5, 14, 30, 31]. Model-only inference latency was measured on CPU for all three precisions using single-image batches, following a 10-iteration warm-up to exclude lazy-initialization overhead. End-to-end pipeline latency — MTCNN face detection, preprocessing, and classification together — was additionally measured on sampled video frames, for the reason given in Section II-E and Section III-F: a classifier's isolated inference time does not establish the throughput a deployed system would actually achieve.

V. RESULTS AND DISCUSSION

A. FaceForensics++ and Celeb-DF v2 Results

The proposed model, evaluated once on each official test set, reaches 80.33% accuracy and 0.8913 ROC-AUC on FaceForensics++, and 81.59% accuracy (82.57% balanced accuracy) and 0.9096 ROC-AUC on Celeb-DF v2. The effect of validation-based threshold selection differed by dataset: on FaceForensics++, the tuned threshold reduced both accuracy and F1 relative to the default; on Celeb-DF v2, it improved both. The default threshold of 0.5 is retained as the primary operating point throughout, for a consistent basis of comparison across models and datasets.

TABLE 1. Proposed model performance on official test sets

Dataset	Threshold	Accuracy	Bal. Acc.	Precision	Recall	F1	ROC-AUC
FaceForensics++	0.5 (default)	0.8033	0.8033	0.8013	0.8067	0.8040	0.8913
FaceForensics++	0.714 (tuned)	0.7900	0.7900	0.8595	0.6933	0.7675	0.8913
Celeb-DF v2	0.5 (default)	0.8159	0.8257	0.9136	0.7945	0.8499	0.9096
Celeb-DF v2	0.393 (tuned)	0.8351	0.8285	0.8937	0.8498	0.8712	0.9096

Figure 1. Confusion matrices, proposed model, official test sets

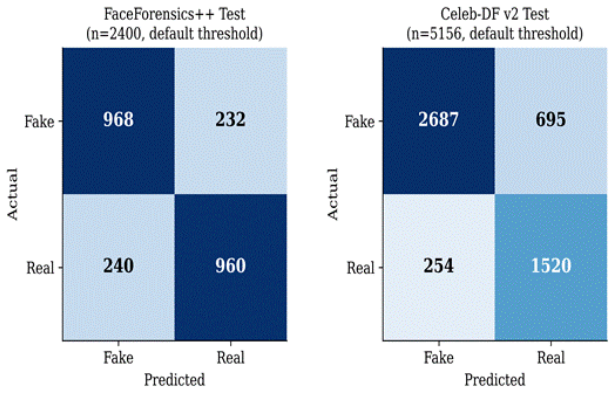


Fig. 1. Confusion matrices, proposed model, official test sets

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

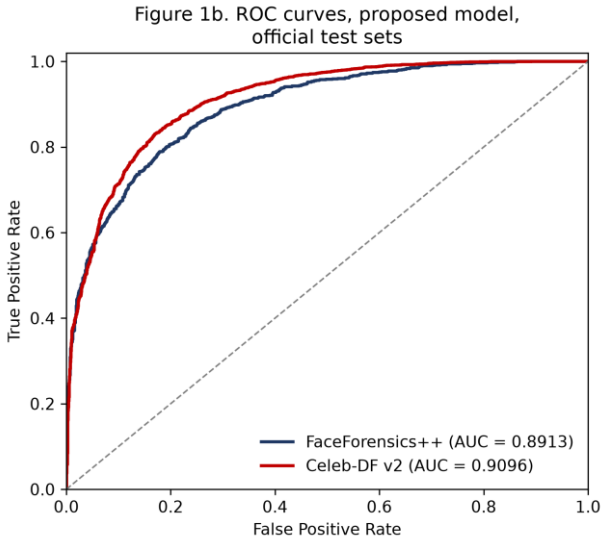


Fig. 1b. ROC curves, proposed model, official test sets

B. Transformer-Depth Ablation

Table 2 reports accuracy and AUC for 0, 1, 2, and 4 Transformer layers under an identical protocol. Same-dataset accuracy does not separate the four depths meaningfully (0.7996–0.8121 accuracy, 0.8869–0.8993 AUC); the 1-layer variant scores marginally highest, but the margin is small relative to the variation observed elsewhere in this study and is unlikely to reflect a meaningful architectural advantage. The two-layer configuration is retained as the architecture examined in this study, rather than being claimed as the optimal depth. Although the one-layer variant performs better on the within-dataset test, the two-layer variant achieves a slightly higher cross-dataset AUC in the FaceForensics++ to Celeb-DF v2 direction; given the small margins involved, this difference should not be interpreted as a reliable or statistically established distinction between the two configurations.

TABLE 2. Accuracy and AUC by Transformer depth (FaceForensics++ test, default threshold)

Depth	Parameters	Accuracy	AUC	F1
0 (CNN-only)	1,107,874	0.7996	0.8869	0.7915
1	1,648,290	0.8121	0.8993	0.8090
2 (proposed)	2,175,394	0.8033	0.8913	0.8040
4	3,229,602	0.8025	0.8948	0.7873

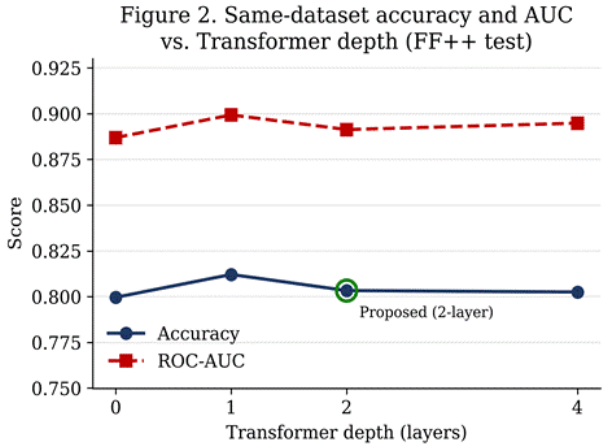


Fig. 2. Same-dataset accuracy and AUC vs. Transformer depth (FaceForensics++ test)

One depth-dependent pattern is worth noting: accuracy on the NeuralTextures manipulation method declines monotonically with Transformer depth, from 0.763 at 0 layers to 0.467 at 4 layers. This is not, on its own, evidence that attention generally harms detection of subtle manipulations — Section V-C reports a baseline architecture that also uses attention and does not show this pattern. The difference may relate to how attention is incorporated into the

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

architectures: the proposed architecture's attention operates late, after the convolutional backbone, through a single learned classification token that attends to the full spatial token sequence, while the baseline's attention is interleaved with convolution throughout the network. MobileViT-XXS achieves the highest NeuralTextures accuracy (0.800) of all six models evaluated, including the proposed architecture (0.527). This pattern may reflect the interaction between the convolutional feature representation and the subsequent attention layers, but the present experiments do not isolate its underlying mechanism. The higher NeuralTextures accuracy of MobileViT-XXS suggests that attention placement and architecture design may influence manipulation-specific performance; however, this remains a hypothesis requiring further analysis.

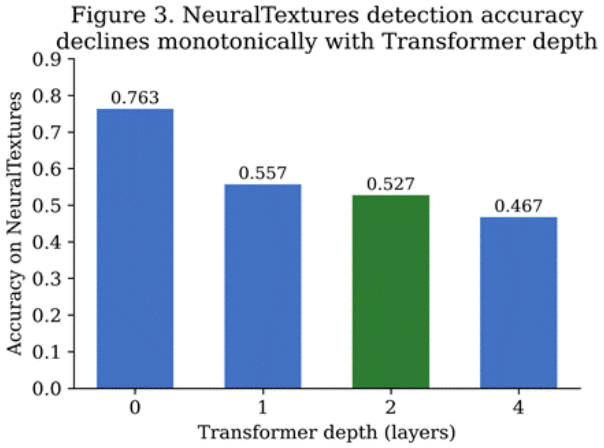


Fig. 3. NeuralTextures detection accuracy declines monotonically with Transformer depth

C. Comparison with Baseline Architectures

EfficientNet-B0 and MobileViT-XXS were trained under the identical protocol used for the proposed architecture. EfficientNet-B0 (4.01M parameters) reaches 82.83% accuracy and 0.9150 AUC on FaceForensics++, exceeding the proposed model's same-dataset figures. MobileViT-XXS (0.95M parameters, the smallest architecture evaluated) reaches 82.42% accuracy and 0.9114 AUC. On same-dataset accuracy alone, the proposed architecture is not the strongest of the three.

TABLE 3. Full six-model comparison (FaceForensics++ test)

Model	Parameters	Accuracy	AUC
0-layer (CNN-only)	1,107,874	0.7996	0.8869
1-layer	1,648,290	0.8121	0.8993
2-layer (proposed)	2,175,394	0.8033	0.8913
4-layer	3,229,602	0.8025	0.8948
EfficientNet-B0	4,010,110	0.8283	0.9150
MobileViT-XXS	951,666	0.8242	0.9114

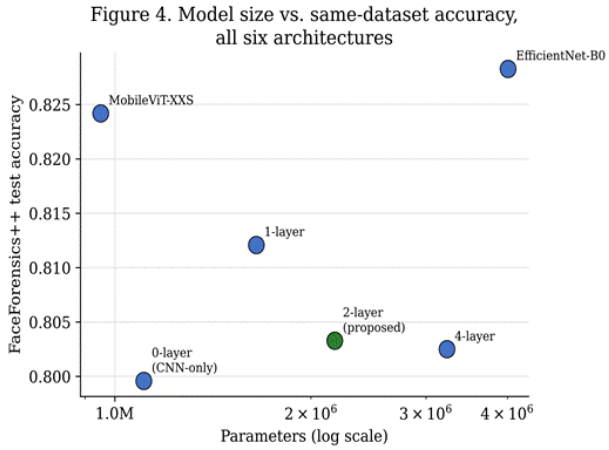


Fig. 4. Model size vs. same-dataset accuracy, all six architectures

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

D. Cross-Dataset Generalization

Evaluated on Celeb-DF v2 after training only on FaceForensics++, the proposed model achieves 0.6700 AUC — the highest of all six architectures tested, ahead of MobileViT-XXS (0.6549), EfficientNet-B0 (0.6525), the 1-layer ablation (0.6615), and the 0- and 4-layer ablations (0.6381, 0.6432). This includes outperforming EfficientNet-B0, a model using 84% more parameters, and narrowly outperforming MobileViT-XXS, the smallest model tested. Confidence intervals overlap across all six architectures, so this is best read as a consistent pattern rather than a statistically decisive result. It is, however, the only evaluation in this paper where the proposed architecture achieves the highest observed AUC, though the overlapping confidence intervals do not establish a statistically significant advantage; same-dataset accuracy, by contrast, favored other architectures throughout, as Fig. 6 makes visible.

TABLE 4. Cross-dataset AUC, FaceForensics++ → Celeb-DF v2 (all six models)

Model	Cross-dataset AUC	95% CI	Bal. Acc.
0-layer (CNN-only)	0.6381	0.6034–0.6745	0.5686
1-layer	0.6615	0.6262–0.6955	0.5907
2-layer (proposed)	0.6700	0.6346–0.7064	0.6163
4-layer	0.6432	0.6054–0.6819	0.5560
EfficientNet-B0	0.6525	0.6187–0.6849	0.6040
MobileViT-XXS	0.6549	0.6282–0.6839	0.6175

Figure 5. Cross-dataset generalization AUC, all six models (error bars: 95% bootstrap CI)

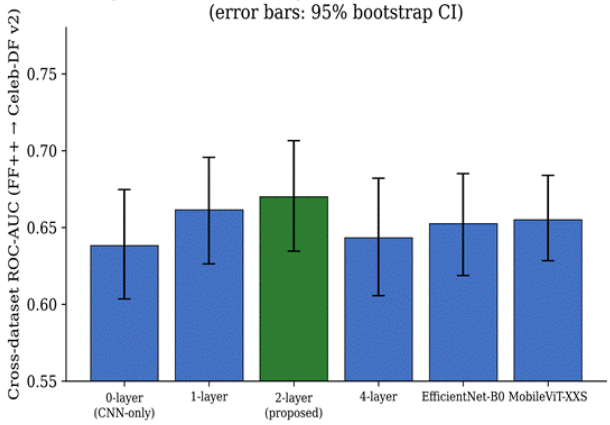


Fig. 5. Cross-dataset generalization AUC, all six models (error bars: 95% bootstrap CI)

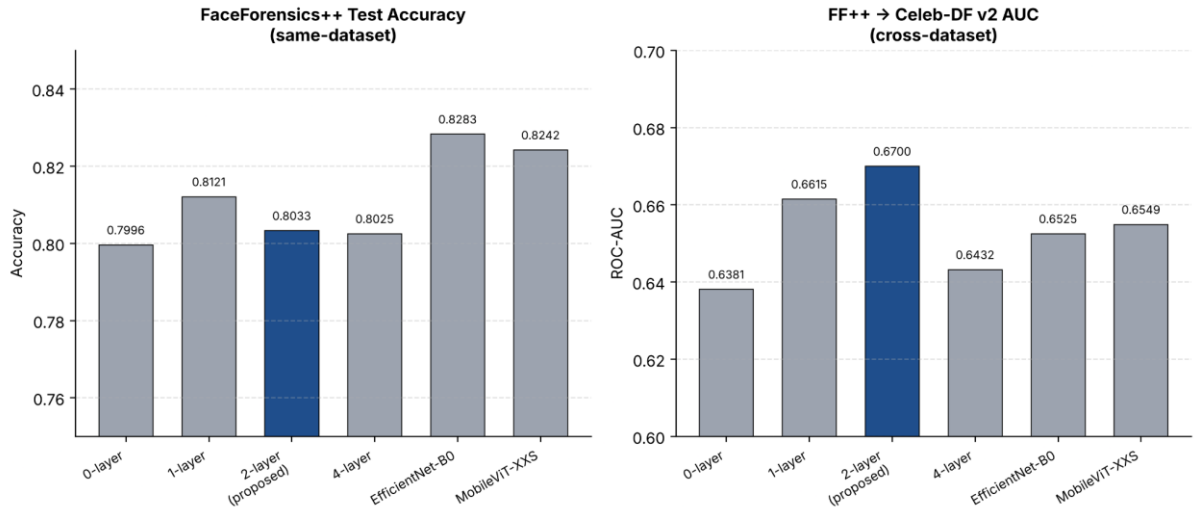


Fig. 6. FaceForensics++ test accuracy (left) and FF++ → Celeb-DF v2 cross-dataset AUC (right) for all six architectures; model ranking reverses between the two metrics

The reverse direction — a model trained on Celeb-DF v2, tested on FaceForensics++ — reaches only 0.6196 AUC, with a stark asymmetry across manipulation methods: 88.33% accuracy on real images against 34–37% on three of

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

the four manipulation types, and 4.33% on FaceSwap specifically. This is consistent with the model having learned features specific to Celeb-DF v2's synthesis method rather than general manipulation cues, and is reported here as a limitation, not something this paper claims to solve.

TABLE 5. Celeb-DF v2 → FaceForensics++ (reverse direction), per-method accuracy

Method	Accuracy
Real	0.8833
Face2Face	0.3700
Deepfakes	0.3500
NeuralTextures	0.3433
FaceSwap	0.0433

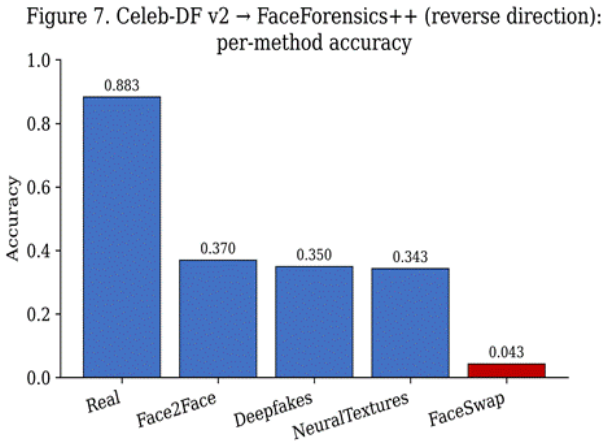


Fig. 7. Celeb-DF v2 → FaceForensics++ (reverse direction): per-method accuracy

E. Hardware-Aware Optimization

Table 6 reports OpenVINO FP32, FP16, and INT8 results for the proposed model on an Intel Core i5-1334U CPU. FP16 conversion costs negligible accuracy (80.29% vs. 80.33% accuracy, 0.8918 vs. 0.8913 AUC on the FaceForensics++ test set) and is recommended as the default deployment precision. INT8 quantization, calibrated on 300 training images via NNCF, costs substantially more: accuracy falls to 70.37% and AUC to 0.8098, a drop of roughly 10 and 8 percentage points respectively (Fig. 8).

TABLE 6. OpenVINO precision comparison (proposed model, FaceForensics++ test)

Inference Precision	Accuracy	F1	ROC-AUC
FP32	0.8033	0.8040	0.8913
FP16	0.8029	0.8043	0.8918
INT8	0.7037	0.6200	0.8098

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

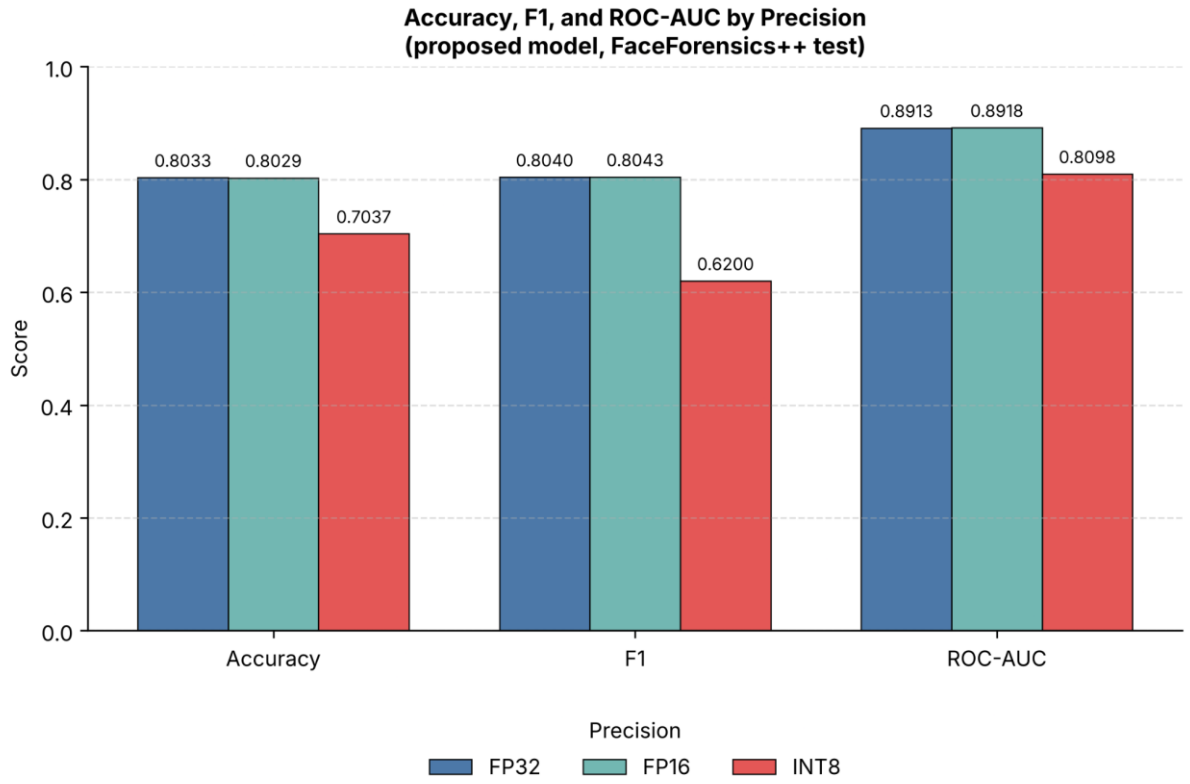


Fig. 8. Accuracy, F1, and ROC-AUC by precision

Two latency-measurement methodologies gave different orderings for INT8 versus FP16 model-only inference. A synthetic benchmark using a single, repeatedly reused input tensor measured INT8 as marginally faster than FP16 (5.87 ms vs. 7.93 ms); the same comparison embedded in the full real-world pipeline, using varying video frames processed through image-to-tensor conversion on each call, measured INT8 as slower (7.03 ms vs. 3.83 ms, Section V-F). This paper treats the latter as the more representative figure, since it reflects the data-handling overhead an actual deployment would incur rather than the more favorable conditions of a cached, single-tensor micro-benchmark. A third measurement, taken under thermally throttled CPU conditions, also showed INT8 running slower, though its absolute values are not treated as reliable for this reason. On balance, this paper does not find a dependable speed advantage for INT8 on this hardware, while its accuracy cost was large and consistent across every measurement; INT8 is therefore not recommended for the evaluated configuration because it produced a substantial accuracy reduction without demonstrating a consistent latency advantage over FP16. The underlying performance difference may depend on execution configuration, operator support, implementation overhead, and hardware characteristics, which were not isolated in this study.

F. End-to-End Pipeline Latency

Classifier-only latency understates the cost of a deployed system. Measured end-to-end — MTCNN face detection, preprocessing, and classification together, on real video frames — the FP16 pipeline runs at 52.59 ms per frame (19.01 FPS), of which face detection accounts for 90.7% (47.71 ms) against 3.83 ms for the classifier itself. In the recorded measurements, the INT8 configuration exhibited higher total latency than FP16. However, the face-detection stage also differed substantially between runs, and the experiment was not designed to isolate the effect of classifier precision on end-to-end latency. Consequently, the total pipeline timings should be interpreted as observed measurements rather than a controlled comparison of FP16 and INT8 pipeline performance; the INT8 configuration recorded 80.06 ms per frame (12.49 FPS), with face detection accounting for 89.3% of total latency. Repeated paired measurements under identical input and runtime conditions would be required to establish a reliable end-to-end latency difference. In both configurations, optimizing the classifier leaves most of the end-to-end latency untouched, because the classifier was never the bottleneck. A meaningful gain in real-world throughput on this class of hardware would need to target face detection, not the classification model evaluated here.

TABLE 7. End-to-end pipeline latency breakdown

Stage	FP16 (ms)	INT8 (ms)
MTCNN detection	47.71	71.48

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

Stage	FP16 (ms)	INT8 (ms)
Preprocessing	1.05	1.55
Model inference	3.83	7.03
Total	52.59	80.06
Throughput (FPS)	19.01	12.49

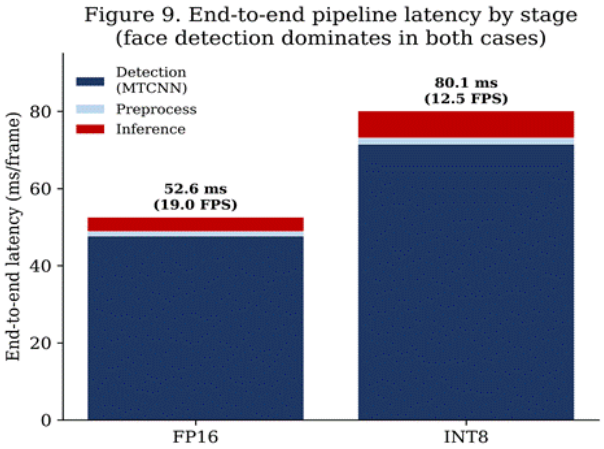


Fig. 9. End-to-end pipeline latency by stage (face detection dominates in both cases)

VI. CONCLUSION AND FUTURE WORK

This paper set out to evaluate a lightweight hybrid CNN-Transformer architecture for deepfake detection under conditions closer to how such a system would actually be deployed: dataset-specific video-level evaluation protocols rather than random frame-level splits, with identity separation maintained for FaceForensics++ but not for Celeb-DF v2, a comparison against independent baseline architectures rather than only the proposed model's own variants, and a hardware benchmark that measures the full detection pipeline rather than the classifier in isolation.

The results do not support a simple "the proposed model wins" narrative, and this paper has tried not to write one. On same-dataset accuracy, EfficientNet-B0 and MobileViT-XXS both score higher than the proposed architecture on FaceForensics++, and a depth ablation shows that one Transformer layer performs marginally better than two. The one area where it leads consistently, against every alternative tested, is cross-dataset generalization: trained on FaceForensics++ and tested on Celeb-DF v2, it reaches the highest AUC of all six architectures compared, including a baseline with 84% more parameters, though the overlapping confidence intervals do not establish a statistically significant advantage. Given that same-dataset accuracy proved unable to separate these architectures in any meaningful way, this cross-dataset result is the more informative evaluation, and the more defensible basis for preferring this architecture over the alternatives tested here.

The hardware benchmark produced a finding this paper did not set out to look for. FP16 conversion costs almost nothing and is the precision recommended for deployment. INT8 quantization, by contrast, costs around ten points of accuracy on this architecture and showed no reliable speed advantage over FP16 on the Intel Core i5-1334U used for testing. The underlying reason for this was not isolated in this study, so the result should be interpreted as specific to the evaluated hardware and execution configuration, not as a general property of quantization. Whether that holds on hardware with VNNI or AVX-512 INT8 support is left open; the finding here is specific to the processor tested, and should be read as a caution against assuming INT8 quantization pays off without checking the target hardware, rather than as a general claim about quantization.

The end-to-end pipeline measurement produced the paper's clearest practical conclusion: face detection, not the classifier, is the bottleneck. At FP16, detection accounts for roughly 91% of total latency, and the fastest classifier evaluated here barely changes the frame rate a deployed system would actually achieve. A thesis or paper proposing a lightweight classifier while leaving detection unexamined is answering only part of the deployment question, and this limitation applies as much to this paper's own scope as to the prior work it reviews.

Several limitations should be stated plainly rather than left for a reader to discover. The Celeb-DF v2 training and test sets are video-disjoint but not identity-disjoint, since the dataset's manipulated videos draw on a comparatively small, reused pool of source identities; the within-dataset Celeb-DF v2 result should be read with that in mind, and the cross-dataset results are the stronger evidence for this reason. A DeiT-tiny baseline was planned but not completed within the available time and compute budget, so the comparison in Section V-C is limited to two baseline architectures rather than three. The reverse cross-dataset direction revealed a sharp, method-specific failure on FaceSwap that this paper has not fully explained, only documented. And the per-

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

method breakdowns reported throughout use each model's validation-tuned decision threshold, while headline accuracy and AUC figures use the default threshold; the two are not always directly comparable, and this is noted here so it is not mistaken for an oversight.

Several directions follow naturally from the limitations above. Completing the DeiT-tiny baseline under this protocol would round out the architecture comparison. The end-to-end latency result in Section V-F suggests that face detection, rather than the classifier, is the more promising target for further optimization — a lighter or hardware-accelerated detector is a natural candidate, though this paper does not test one. Training on more than one manipulation technique simultaneously may help address the FaceSwap-specific failure observed in the reverse cross-dataset direction, though this remains speculative without direct experimentation. Finally, whether the INT8 quantization result reported here is specific to the CPU tested or holds more generally is an open question that would benefit from testing on hardware with dedicated INT8 acceleration.

ACKNOWLEDGMENT

The author would like to thank Er. Bhuvnesh Kumar, Assistant Professor (CSE), Department of Computer Science & Engineering, Amritsar Group of Colleges, for his valuable guidance and suggestions throughout the course of this work.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

USE OF GENERATIVE AI

AI-assisted tools (Claude, ChatGPT) were used during preparation of this manuscript for drafting and language editing, as disclosed in this statement. All AI-generated content was reviewed and verified by the author, who takes full responsibility for the accuracy and integrity of the work.

REFERENCES

[1] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2017.

[2] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, "FaceForensics++: Learning to detect manipulated facial images," in Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV), 2019.

[3] A. Howard et al., "Searching for MobileNetV3," in Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV), 2019, pp. 1314-1324.

[4] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A large-scale challenging dataset for deepfake forensics," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2020, pp. 3204-3213.

[5] A. Kozlov, I. Lazarevich, V. Shamporov, N. Lyalyushkin, and Y. Gorbachev, "Neural network compression framework for fast model inference," arXiv:2002.08679, 2020.

[6] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in Proc. Int. Conf. Learning Representations (ICLR), 2021.

[7] S. Mehta and M. Rastegari, "MobileViT: A lightweight, general-purpose, and mobile-friendly vision transformer," in Proc. Int. Conf. Learning Representations (ICLR), 2022.

[8] D. A. Coccomini, N. Messina, C. Gennaro, and F. Falchi, "Combining EfficientNet and vision transformers for video deepfake detection," in Proc. Int. Conf. Image Analysis and Processing (ICIAP), 2022, pp. 219-229.

[9] E. Altuncu, V. N. L. Franqueira, and S. Li, "Deepfake: Definitions, performance metrics and standards, datasets, and a meta-review," Frontiers in Big Data, vol. 7, art. 1400024, 2024, doi: 10.3389/fdata.2024.1400024.

[10] A. Luo, R. Cai, C. Kong, X. Kang, J. Huang, and A. C. Kot, "Forgery-aware adaptive learning with vision transformer for generalized face forgery detection," IEEE Trans. Circuits Syst. Video Technol., vol. 35, pp. 4116-4129, 2025.

[11] G. Petmezas, V. Vanian, K. Konstantoudakis, E. E. I. Almaloglou, and D. Zarpalas, "Video deepfake detection using a hybrid CNN-LSTM-transformer model for identity verification," Multimedia Tools Appl., vol. 84, pp. 40617-40636, 2025.

[12] K. Sudarshana and Y. Vamsidhar, "UAM-Net: Robust deepfake detection through hybrid attention into scalable convolutional network," Expert Systems, vol. 42, e70009, 2025.

[13] G. Rani, A. Kothekar, S. G. Philip, V. S. Dhaka, E. Zumpano, and E. Vocaturo, "Lightweight and hybrid transformer-based solution for quick and reliable deepfake detection," Frontiers in Big Data, vol. 8, 1521653, 2025.

[14] W. Foster and Intel OpenVINO Team, "OpenVINO 2026.1: New models, more performance," Intel OpenVINO Toolkit Blog, 2026.

[15] F. Zhao, J. Gong, and E. Wang, "Accelerating PyTorch with Intel Extension for PyTorch," Intel Developer technical article, n.d.

[16] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," IEEE Signal Process. Lett., vol. 23, no. 10, pp. 1499-1503, 2016.

[17] Intel Corporation, "Intel Extension for PyTorch Installation Guide (EOL)," official project documentation, n.d. Available: https://pytorch-extension.intel.com/. Accessed: 9 Oct. 2026.

[18] M. Brodarič, P. Peer, and V. Štruc, "Cross-Dataset Deepfake Detection: Evaluating the Generalization Capabilities of Modern DeepFake Detectors," Proc. 27th Computer Vision Winter Workshop (CVWW), 2024, pp. 1-10.

[19] B. Li et al., "How Generalizable are Deepfake Image Detectors? An Empirical Study," arXiv:2308.04177, 2023.

[20] H. Cheng, Y. Guo, T. Wang, L. Nie, and M. Kankanhalli, "Towards Generalizable Deepfake Detection by Primary Region Regularization," arXiv:2307.12534, 2023.

[21] R. Yang, K. You, C. Pang, X. Luo, and R. Lan, "CSTAN: A Deepfake Detection Network with CST Attention for Superior Generalization," Sensors, vol. 24, no. 22, art. 7101, 2024.

[22] B. Batagelj, A. Kronovšek, V. Štruc, and P. Peer, "Robust Cross-Dataset Deepfake Detection with Multitask Self-Supervised Learning," ICT Express, vol. 11, pp. 1-6, 2025, doi: 10.1016/j.icte.2025.02.011.

[23] S. Yang, H. Guo, S. Hu, B. Zhu, Y. Fu, S. Lyu, X. Wu, and X. Wang, "CrossDF: Improving Cross-Domain Deepfake Detection with Deep Information Decomposition," Frontiers in Big Data, vol. 8, art. 1669488, 2025, doi: 10.3389/fdata.2025.1669488.

[24] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," Proc. ICML, 2019, pp. 6105-6114.

[25] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training Data-Efficient Image Transformers & Distillation Through Attention," Proc. ICML, PMLR vol. 139, 2021, pp. 10347-10357.

[26] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," Proc. CVPR, 2018, pp. 4510-4520.

[27] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. Canton Ferrer, "The DeepFake Detection Challenge (DFDC) Dataset," arXiv:2006.07397, 2020.

[28] 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," Proc. CVPR, 2020, pp. 2889-2898.

[29] A. G. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv:1704.04861, 2017.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

[30] S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," Proc. ICLR, 2016.

[31] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," Proc. CVPR, 2018, pp. 2704-2713.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)