DeepShield: Fortifying Deepfake Video Detection with Local and Global Forgery Analysis (Supplementary Material)

Yinqi Cai[1,2*] Jichang Li[2*] Zhaolun Li[3] Weikai Chen[†] Rushi Lan[3] Xi Xie[1]
Xiaonan Luo[3] Guanbin Li[1,2,4‡]
[1]Sun Yat-sen University [2]Pengcheng Laboratory [3]Guilin University of Electronic Technology
[4]Guangdong Key Laboratory of Big Data Analysis and Processing
caiyq27@mail2.sysu.edu.cn, li.jichang@pcl.ac.cn, liguanbin@mail.sysu.edu.cn

This supplementary material provides additional details and insights into our proposed model namely *DeepShield*. Specifically, we include the following components: detailed model architecture, additional implementation details, and additional experimental analysis. The pseudo-code for the training procedures of DeepShield is illustrated in Algorithm 1.

1. Detailed Model Architecture

In this work, we utilize CLIP's pre-trained image encoder, referred to as CLIP-ViT, as the video encoder for generalizable deepfake video detection. To capture spatial manipulations and temporal inconsistencies efficiently, we fine-tune CLIP-ViT with the parameter-efficient ST-Adapter [2], as depicted in Figure 1. The ST-Adapter is inserted before the Multi-Head Self-Attention and Feed-Forward Network in each Transformer block of CLIP-ViT.

Let $\mathbf{X} \in \mathbb{R}^{T\times P\times d}$ denote the input patch embeddings, where T is the number of frames per video clip, P is the number of patch tokens, and d is the embedding dimension. The ST-Adapter can be defined as follows,

$$\text{ST-Adapter}(\mathbf{X}) = \mathbf{X} + \text{g}(\text{DWConv3D}(\mathbf{X}\mathbf{W}^{\text{down}}))\mathbf{W}^{\text{up}}, \tag{1}$$

where $\mathbf{W}^{\text{down}}$ and $\mathbf{W}^{\text{up}}$ are learnable parameter weights of the down- and up-projection linear layers, respectively, and $\text{g}(\cdot)$ is an activation function. As well, $\text{DWConv3D}(\cdot)$ is a depth-wise 3D convolution layer with kernel size $T \times H \times W = 3 \times 1 \times 1$, which effectively captures temporal information.

Algorithm 1 Pseudo-code for Training Process

Input: Training set of real and fake videos, Training epochs *Total_Epoch*, Iterations per epoch *Total_Iter*, Initialized model ψ with backbone E and classifier ϕ
Output: Optimized model parameters ψ

for $epoch$ = 1 **to** *Total_Epoch* **do**
  for $iter$ = 1 **to** *Total_Iter* **do**
    Randomly select a mini-batch $\mathcal{B}$ from the training set
    Initialize $\mathcal{B}^{\text{real}} \leftarrow \emptyset$, $\mathcal{B}^{\text{fake}} \leftarrow \emptyset$, $\mathcal{B}^{\text{blend}} \leftarrow \emptyset$
    for each video $\mathbf{V}$ in mini-batch $\mathcal{B}$ **do**
      if $\mathbf{V}$ is real **then**
        Apply SAM to blend video: $\mathbf{V}^{\text{blend}} \leftarrow \text{SAM}(\mathbf{V})$
        $\mathcal{B}^{\text{real}} \leftarrow \mathcal{B}^{\text{real}} \cup \{\mathbf{V}\}$
        $\mathcal{B}^{\text{blend}} \leftarrow \mathcal{B}^{\text{blend}} \cup \{\mathbf{V}^{\text{blend}}\}$
        $\mathcal{B}^{\text{fake}} \leftarrow \mathcal{B}^{\text{fake}} \cup \{\mathbf{V}^{\text{blend}}\}$
      else
        Randomly select a real video $\mathbf{V}^{\text{real}}$ from the training set
        $\mathcal{B}^{\text{fake}} \leftarrow \mathcal{B}^{\text{fake}} \cup \{\mathbf{V}\}$
        $\mathcal{B}^{\text{real}} \leftarrow \mathcal{B}^{\text{real}} \cup \{\mathbf{V}^{\text{real}}\}$
      end if
    end for
    Apply DFA on fake videos: $\mathcal{B}^{\text{DFA}} \leftarrow \text{DFA}(\mathcal{B}^{\text{fake}})$
    $\mathcal{B}^{\text{fake}} \leftarrow \mathcal{B}^{\text{fake}} \cup \mathcal{B}^{\text{DFA}}$
    $\mathcal{B} \leftarrow \mathcal{B} \cup \mathcal{B}^{\text{DFA}}$
    $\mathcal{L}^{\text{overall}} \leftarrow \mathcal{L}_{\text{LPG}}(\mathcal{B}^{\text{real}} \cup \mathcal{B}^{\text{blend}}) + \mathcal{L}_{\text{GFD}}(\mathcal{B})$
    Optimize ψ using $\mathcal{L}^{\text{overall}}$
  end for
end for
return Optimized model ψ

*Equal Contribution.
†This paper solely reflects the author's personal research and is not associated with the author's affiliated institution.
‡Corresponding Author.

Method	Architecture	Input Type	Testing Set AUC (%)				
			FF++	CDF	DFDCP	DFDC	DFD
CLIP-ViT (vanilla)	ViT-B/16	Frame	55.2	60.0	59.0	55.2	57.4
CLIP-ViT (adapted)	ViT-B/16+ST-Adapter	Video	98.2	85.4	88.9	78.4	95.3
DeepShield (Ours)	ViT-B/16+ST-Adapter	Video	**99.2**	**92.2**	**93.2**	**82.8**	**96.1**

Table 1. Comparison results (video-level AUC, %) between DeepShield and its two variants on **cross-dataset evaluation**. The comparison methods include both frame-based and video-based approaches, employing various model architectures.

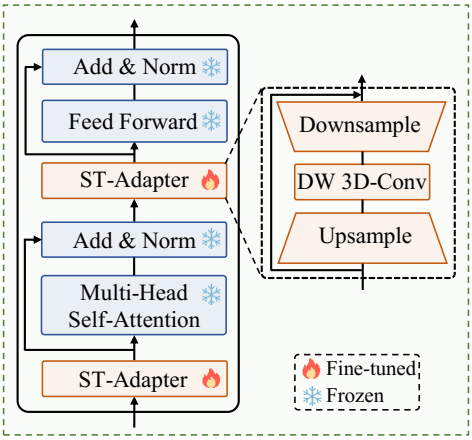

Figure 1. Detailed architecture of the transformer block in the CLIP image encoder, with two ST-Adapters integrated. Each ST-Adapter incorporates a depth-wise 3D convolution layer to effectively capture temporal information.

Method	FaceDancer (WACV23)	MCNet (ICCV23)	Avg.
TALL [4]	86.0	53.3	69.7
DeepShield (Ours)	**98.1**	**95.3**	**96.7**

Table 2. Comparison results of our proposed DeepShield and TALL [4] on the DF40 subset [5] using video-level AUC (%) as metric.

2. Additional Implementation Details

Each training batch consists of 16 samples: four real videos, four fake videos (either sampled from the training set or generated using *Spatiotemporal Artifact Modeling*), and eight additional video features derived from these fake videos through *Domain Feature Augmentation* (including Domain-Bridging Feature Generation and Boundary-Expanding Feature Generation). During training, four clips of 12 consecutive frames are randomly sampled from each video. For inference, each video is divided into four segments. From each segment, the first 12 frames are extracted to form four clips, and the final prediction is computed by averaging the prediction probabilities of these clips.

3. Additional Experimental Analysis

Exploration towards Model Variants In this work, we build two variants of our proposed DeepShield to evaluate its effect on deepfake video detection: **CLIP-ViT (vanilla)**, which directly employs the pre-trained CLIP image encoder without fine-tuning for the deepfake video detection task, and **CLIP-ViT (adapted)**, where the original CLIP-ViT model is equipped with the ST-Adapter and fine-tuned via binary real-fake supervision for parameter-efficient adaptation to deepfake video detection. As illustrated by Table 1 and Table 1 (of manuscript), our baselines do not demonstrate a significant advantage over existing state-of-the-art (SOTA) methods, ensuring a fair comparison between our full model and others. In the meanwhile, our full model DeepShield consistently outperforms the baseline variants across all cases in cross-dataset evaluations and even surpasses all current SOTA algorithms. This demonstrates the effectiveness of our proposed strategy in significantly improving the model's overall performance in detecting forged samples.

More Results on Up-to-date Deepdake Evaluation Dataset To better evaluate DeepShield's generalization capability across different types of forgeries, we conducted tests on a subset of the DF40 dataset [5]. This subset includes FaceDancer [3] (with face-swapping deepfakes generated by neural networks) and MCNet [1] (talking-head generation). As shown in Table 2, DeepShield outperforms TALL [4] on these more complex forgeries, demonstrating its strong adaptability even beyond blending-based manipulations.

Necessity of Temporal Artifact Generating (TAG) In our study, we propose Temporal Artifact Generating (TAG) in the SAM component, which applies Spatial Artifact Generating (SAG) frame-by-frame to simulate temporal inconsistencies in video frames. This strategy maintains the augmentation consistency on source or target frames and uniformity in blending mask adjustments across T frames. As shown in Table 3, removing TAG reduces the average deepfake detection performance by 2.5%, which suggests that using spatial artifact generation alone is insufficient for optimizing DeepShield's deepfake video detection capabilities. The strong detection performance on both cross-dataset and

Variant	CDF	DFDCP	DFDC	DFD	Avg.
DeepShield (Ours)	**92.2**	**93.2**	**82.8**	**96.1**	**91.1**
DeepShield w/o TAG	89.7	90.0	78.5	96.0	88.6

Table 3. Ablation study results (video-level AUC, %) of DeepShield on TAG, showing the impact of removing itself. These experiments conduct model training on FF++ (HQ) and perform cross-dataset evaluations on CFD, DFDC, DFDCP, and DFD.

cross-manipulation cases also confirms that our SAM approach does not cause overfitting, thus maintaining detection efficacy in deepfake video analysis.

Hyper-Parameter Sensitivity to the Threshold θ We conduct ablation studies to analyze the impact of the threshold θ in the *Patch Scoring Function* on cross-dataset performance. As shown in Table 4, the threshold plays a critical role in balancing sensitivity and robustness in scoring patch regions. Lower thresholds (e.g., $\theta = 10$) result in more consistent performance across datasets, achieving the best average score of 92.7. However, when the threshold becomes too high (e.g., $\theta = 150$), the performance slightly declines as critical low-scoring patches may be overlooked. These results demonstrate the importance of carefully tuning θ to optimize the balance between precision and coverage in patch scoring.

Threshold (θ)	CDF	DFDCP	Avg.
10	92.2	**93.2**	**92.7**
20	92.3	92.8	92.5
50	**92.6**	92.6	92.6
100	91.9	93.0	92.4
150	92.2	92.3	92.2

Table 4. Impact of the threshold θ in *Patch Scoring Function* on cross-dataset performance, using video-level AUC(%) as the metric. The best results are highlighted in bold.

Ablation for Weights in the Losses We conduct ablation studies on the balancing weights ω and υ in the loss function. As shown in Table 5, the weights significantly influence performance. When $\omega = 0.5$ and $\upsilon = 0.5$, the model achieves the best average performance of 89.4, indicating a balanced optimization. Increasing ω or υ can improve specific dataset scores, such as DFDC (83.1 at $\upsilon = 1$), but may reduce generalization. These results highlight the importance of tuning weights for optimal trade-offs between dataset-specific and overall performance.

ω	υ	CDF	DFDC	DFDCP	Avg.
0.5	0.5	92.2	82.8	93.2	**89.4**
1	0.5	**92.8**	79.4	**94.1**	88.8
5	0.5	89.8	82.0	93.1	88.3
0.5	1	91.6	**83.1**	92.9	89.2
0.5	5	91.5	82.1	92.9	88.8

Table 5. Impact of the balancing weights ω and υ in the loss function on cross-dataset performance, using video-level AUC(%) as the metric. The best results for each metric are highlighted in bold.

References

[1] Fa-Ting Hong and Dan Xu. Implicit identity representation conditioned memory compensation network for talking head video generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23062–23072, 2023. 2

[2] Junting Pan, Ziyi Lin, Xiatian Zhu, Jing Shao, and Hongsheng Li. St-adapter: Parameter-efficient image-to-video transfer learning. *Advances in Neural Information Processing Systems*, 35:26462–26477, 2022. 1

[3] Felix Rosberg, Eren Erdal Aksoy, Fernando Alonso-Fernandez, and Cristofer Englund. Facedancer: Pose-and occlusion-aware high fidelity face swapping. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3454–3463, 2023. 2

[4] Yuting Xu, Jian Liang, Gengyun Jia, Ziming Yang, Yanhao Zhang, and Ran He. Tall: Thumbnail layout for deepfake video detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22658–22668, 2023. 2

[5] Zhiyuan Yan, Taiping Yao, Shen Chen, Yandan Zhao, Xinghe Fu, Junwei Zhu, Donghao Luo, Chengjie Wang, Shouhong Ding, Yunsheng Wu, et al. Df40: Toward next-generation deepfake detection. *arXiv preprint arXiv:2406.13495*, 2024. 2