Multi-domain Learning for Updating Face Anti-spoofing Models — Supplementary —

Xiao Guo, Yaojie Liu, Anil Jain, and Xiaoming Liu

Michigan State University
{guoxia11,liuyaoj1,liuxm,jain}@cse.msu.edu

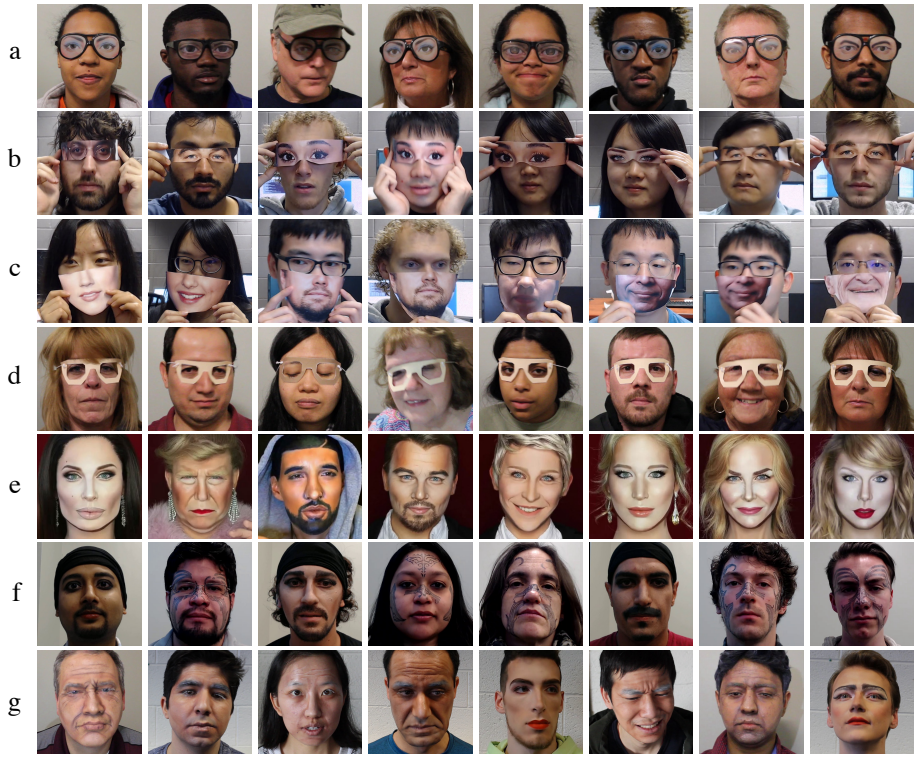

Fig. 1: SiW-Mv2 dataset samples in Covering and Makeup spoof categories. These spoof attacks are: (a) Funny Eyes, (b) Partial Eyes, (c) Partial Mouths, (d) Paperglass, (e) Impersonate Makeup, (d) Obfuscation Makeup, and (f) Cosmetic Makeup. More details are in Tab. 1.

In this supplementary material, Sec. 1 and 2 introduce the SiW-Mv2 dataset, designed protocols, and the baseline model performance. Sec. 3 adds more explanations of the proposed FAS-*wrapper*, and Sec. 4 reports the implementation details of our experiments in the main paper's Sec. 5.2.

Fig. 2: SiW-Mv2 dataset samples in 3D and 2D Attack spoof categories. These spoof attacks are: (a) Full Mask, (b) Transparent Mask, (c) Paper Mask, (d) Silicone Head, (e) Mannequin, (d) Print, and (f) Replay. More details are in Tab. 1.

1 SiW-Mv2 Dataset

In this section, Sec. 1.1 and 1.2 report the SiW-Mv2 dataset and three protocols. In Sec. 2, we first verify the baseline performance on the Oulu-NPU and SiW datasets, and then report the baseline performance on the SiW-Mv2 dataset [1].

1.1 Introduction

Our SiW-Mv2 dataset is the updated version of the original SiW-M dataset [9], which is unavailable due to the privacy issue. For the SiW-Mv2 dataset, we curate new samples and add one more spoof category (*e.g.*, *partial mouths*) to improve the overall spoof attack diversity. As a result, SiW-Mv2 has 785 videos from 493 live subjects, and 915 spoof videos from 600 subjects. Among these spoof videos, we have 14 spoof attack types, spanning from typical 2D spoof attacks (*e.g.*, *print* and *replay*), various masks, different makeups, and physical material

[1] The source code and download instructions can be found on this page.

Spoof Category	Spoof Attack	Video #	Subject #	Purpose
Covering	Funny Eyes	179	172	Cover.
	Partial Eyes	57	27	Hide
	Partial Mouths	29	26	Hide
	Paperglasses	76	71	Hide
Makeup	Impersonate	61	61	Imper.
	Obfuscation	22	15	Imper.
	Cosmetic	52	35	Imper.
3D Attack	Full Mask	72	12	Imper.
	Transparent Mask	60	60	Hide
	Paper Mask	17	6	Hide
	Silicone Head	17	4	Hide
	Mannequin	40	29	Hide
2D Attack	Replay	98	21	Imper.
	Print	135	61	Imper.

Table 1: SiW-Mv2 dataset details. Each spoof attack represents a different purpose of spoofing, such as impersonation or hiding the original identity. [**Keys**: Hide: hiding identity, Imper.: impersonation]

coverings. Some samples are shown in Fig. 1 and Fig. 2, and more details are in Tab. 1. Moreover, in the SiW-Mv2 dataset, spoof attacks can either modify the subject appearance to impersonate other people, such as *impersonate makeup* and *silicone head*, or hide the subject identity (*e.g.*, *funny eyes* and *paper mask*). The details of the dataset collection process are in Sec. 4 of the work [9]. Lastly, the recent usage of SiW-Mv2 is also found in the domain of image forensics [3,5], where methods are developed to distinguish real images from images that are manipulated or generated by Artificial Intelligence.

1.2 Protocols and Metrics

In the SiW-Mv2 dataset, we design three different protocols which evaluate the model ability to detect known and unknown spoof attacks, as well as the generalization ability to spoof attacks at different domains, respectively.

- **Protocol I**: *Known Spoof Attack Detection.* We divide live subjects and subjects of each spoof pattern into train and test splits. We train the model on the training split and report the overall performance on the test split.
- **Protocol II**: *Unknown Spoof Attack Detection.* We follow the leave-one-out paradigm — keep 13 spoof attack and 80% live subjects as the train split, and use the remaining one spoof attacks and left 20% live subjects as the test split. We report the test split performance for both individual spoof attacks, as well as the averaged performance with standard deviation.
- **Protocol III**: *Cross-domain Spoof Detection.* We partition the SiW-Mv2 into 5 sub-datasets, as described in Sec. 4 in the main paper. We train the model on the source domain dataset, and evaluate the model on test splits of 5 different domains. Each sub-dataset performance, and averaged performance with standard deviation are reported.

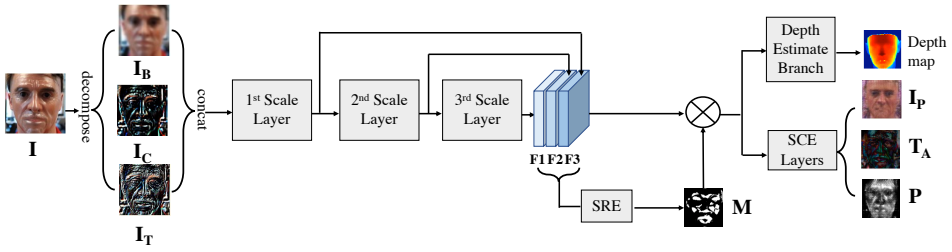

Fig. 3: The overall SRENet architecture. We decompose the input image $\mathbf{I}$ into three elements (*e.g.*, $\mathbf{I_B}$, $\mathbf{I_C}$ and $\mathbf{I_T}$), which represent the image information at different frequency levels. The multi-scale feature extractor takes the concatenation of these three elements to generate multi-scale features (*e.g.*, $\mathbf{F_1}$, $\mathbf{F_2}$, and $\mathbf{F_3}$). Such multi-scale features are fed to the depth estimate branch for estimating the face depth, and SCE layers for estimating inpanting trace ($\mathbf{I_P}$), region trace ($\mathbf{P}$), and additive trace ($\mathbf{T_A}$). More importantly, SRE produces $\mathbf{M}$ to help pinpoint the spoof region. In the work [8], such SCE layers are three different branches, and three traces are used to synthesize the live counterpart of the input image via the adversarial training.

To be consistent with the previous work, we use standard FAS metrics to measure the SRENet performance. These metrics are Attack Presentation Classification Error Rate (APCER), Bona Fide Presentation Classification Error Rate (BPCER), and Average Classification Error Rate (ACER) [1], and Receiver Operating Characteristic (ROC) curve, respectively.

2 Baseline and Performance

We present our baseline architecture, dubbed SRENet, in Fig. 3. Specifically, the proposed SRENet is based on PhySTD [8]. However, compared to the original PhySTD, we make two modifications, which simplify the model and even achieves the better spoof detection performance. First, the original PhySTD generates three different traces to reconstruct both spoof and live counterparts of the given input image, whereas in SRENet we only leverage these three traces to construct the live counterpart of the given input image. Secondly, we integrate the Spoof Region Estimator (SRE) in Sec. 3.2 into the architecture, and this SRE serves as an attention module to help pinpoint the spoof area by the binary mask. Note that the difference between SRENet and FAS-*wrapper* is fundamental: SRENet is a face spoof detection model, whereas FAS-*wrapper* targets at the multi-domain FAS updating. Furthermore, from the architectural perspective, the proposed SRE plays key roles in both SRENet and FAS-*wrapper*.

Empirically, we first verify the effectiveness of the SRENet on the Oulu-NPU and SiW datasets. The performance is reported in Tab. 2, which shows that our model performance is comparable with that of state-of-the-art methods, such as PhySTD [8] and PatchNet [12]. After that, we report the SRENet performance on the three designed protocols of the SiW-Mv2 dataset, and results are in Tab. 3.

Protocol	Method	APCER (%)	BPCER (%)	ACER (%)
1	PsySTD. [8]	0.0	0.8	0.4
	PatchNet [12]	**0.0**	**0.0**	**0.0**
	Ours	0.2	0.6	0.4
2	PsySTD. [8]	1.2	1.3	1.3
	PatchNet [12]	**1.1**	1.2	1.2
	Ours	1.4	**0.8**	**1.1**
3	PsySTD. [8]	1.7 ± 1.4	2.2 ± 3.5	1.9 ± 2.3
	PatchNet [12]	1.8 ± 1.5	$\mathbf{0.6 \pm 1.2}$	$\mathbf{1.2 \pm 1.3}$
	Ours	$\mathbf{1.6 \pm 1.6}$	1.2 ± 1.4	1.4 ± 1.5
4	PsySTD. [8]	2.3 ± 3.6	4.2 ± 5.4	3.6 ± 4.2
	PatchNet [12]	2.5 ± 3.8	$\mathbf{3.3 \pm 3.7}$	$\mathbf{2.9 \pm 3.0}$
	Ours	$\mathbf{2.2 \pm 1.9}$	3.8 ± 4.1	3.0 ± 3.0

(a)

Protocol	Method	APCER (%)	BPCER (%)	ACER (%)
1	PsySTD. [8]	0.0	0.0	0.0
	PatchNet [12]	0.0	0.0	0.0
	Ours	0.0	0.0	0.0
2	PsySTD. [8]	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
	PatchNet [12]	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
	Ours	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
3	PsySTD. [8]	13.1 ± 9.4	1.6 ± 0.6	7.4 ± 4.3
	PatchNet [12]	$\mathbf{3.1 \pm 1.1}$	$\mathbf{1.8 \pm 0.8}$	$\mathbf{2.5 \pm 0.5}$
	Ours	6.3 ± 1.3	2.9 ± 0.4	4.6 ± 0.9

(b)

Table 2: The baseline (SRENet) performance on (a) OULU-NPU and (b) SiW datasets.

Metric	Covering				Makeup			3D Attack					2D Attack		Overall
	Fun.	Eye	Mou.	Pap.	Ob.	Im.	Cos.	Imp.	Sil.	Tra.	Pap.	Man.	Rep.	Print	
ACER(%)	1.1	1.1	0.2	1.1	0.0	3.6	2.7	0.0	5.4	0.0	0.6	0.0	1.9	1.5	2.6
TDR@ FDR=1.0(%)	31.2	47.8	100.0	44.8	100.0	80.0	87.5	100.0	34.3	100.0	100.0	100.0	97.4	98.2	89.4

(a) Protocol I: Unknown Spoof Attack Detection.

Metric	Covering				Makeup			3D Attack					2D Attack		Average
	Fun.	Eye	Mou.	Pap.	Ob.	Im.	Cos.	Imp.	Sil.	Tra.	Pap.	Man.	Rep.	Print	
APCER(%)	26.1	5.4	2.3	6.5	2.7	6.1	8.0	8.8	10.0	0.0	1.1	8.0	19.9	2.7	7.7 ± 7.0
BPCER(%)	33.0	0.0	0.0	17.3	0.0	42.9	13.7	7.1	8.5	0.0	0.0	0.0	16.0	16.5	11.1 ± 12.9
ACER(%)	29.5	2.7	1.1	11.9	1.3	24.5	10.9	8.0	9.2	0.0	0.6	4.0	17.9	9.6	9.4 ± 8.8
TDR@ FDR=1.0(%)	8.9	37.0	88.4	4.0	98.3	23.8	39.2	61.4	47.4	100	100	66.6	39.3	78.9	56.7 ± 32.0

(b) Protocol II: Unknown Spoof Attack Detection.

Metric	Source Domain	Spoof	Race	Age	Illum.	Average
APCER(%)	2.8	12.5	18.9	15.4	7.7	11.5 ± 5.7
BPCER(%)	1.5	17.4	11.1	0.0	0.0	6.0 ± 7.0
ACER(%)	2.2	14.9	15.0	7.7	3.8	8.7 ± 5.4
TDR@ FDR=1.0(%)	86.2	28.3	44.4	55.6	66.7	56.2 ± 19.6

(c) Protocol III: Cross Domain Spoof Detection.

Table 3: The baseline (SRENet) performance on three protocols in SiW-Mv2 dataset.

3 More Method Details

3.1 Face Reconstruction Analysis

The motivation of using face reconstruction in our method is that, although it cannot reconstruct the perfect live faces given its spoof counterpart, but it can largely shed the light on spatial pixel location where spoofness occurs. As introduced in Sec. 3.2, we use such a reconstruction method to generate the preliminary mask $\mathbf{I}_{pre}$ as the pseudo label that supervises the proposed *SRE*. We offer the detailed visualization in Fig. 4.

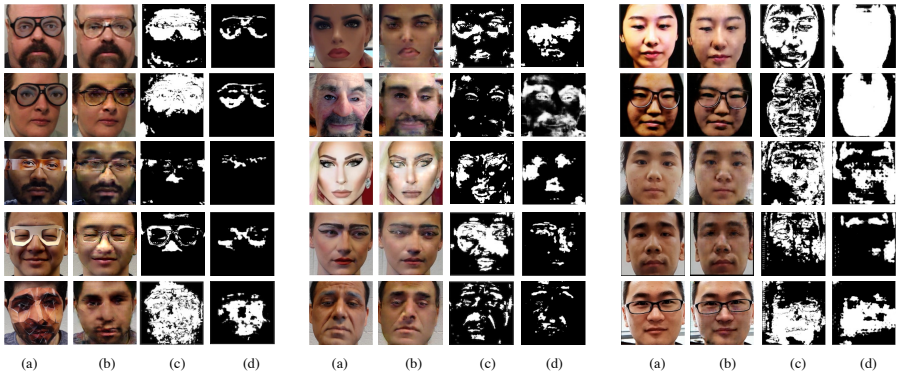

Fig. 4: The visualization of (a) input spoof image, (b) live counterpart reconstruction, (c) generated preliminary mask, and (d) estimated spoof region from our method. Three columns (from left to right) represent three different spoof genera, covered material, makeup stroke and visual artifacts from *replay* and *print* attack.

In general, we have categorized spoof types into three main genera: (a) covered materials; (b) makeup stroke; (c) visual artifacts (*i.e.*, color distortion and moire effect) in the *replay* and *print* attacks. In particular, for covered materials, reconstruction methods largely erase these spoof materials, such as *funny glasses*, and *paper mask*. As shown from Fig. 4, $\mathbf{I}_{pre}$ can roughly locate the pixel-wise spatial location that has been covered by the spoof material, and the estimated spoof region gives the more accurate prediction on pixels that are covered by these spoof materials. For makeup stroke, the reconstruction method changes the color and texture of the facial makeup area, making them similar to the natural skin. $\mathbf{I}_{pre}$ offers the scattered, discrete binary mask and estimated spoof region provides the smoother region indicating the spoofness. For *replay* and *print* attacks, the reconstruction method modifies facial structure (*i.e.,* nose and eyes) of the human face, or largely change the image's appearance, by providing the image with a sense of depth. Similar as makeup stroke genera, $\mathbf{I}_{pre}$ gives very discontinuous predictions on spoofness whereas the estimated spoof region is smoother and semantic.

3.2 Model Response

When a target domain image $\mathbf{I}_{target}$ is fed to the pre-trained model, the pre-trained model will be activated, as if the pre-trained model takes as input source domain images $\mathbf{I}_{source}$ which has resemblance with $\mathbf{I}_{target}$. In other words, the pre-trained model recognizes it as source domain images $\mathbf{I}_{source}$ which has common characteristic and pattern with $\mathbf{I}_{target}$. Therefore, source data can manifest themselves on the response of the model, or in other words, keeping model response allows us to have memory or characteristics of the source data.

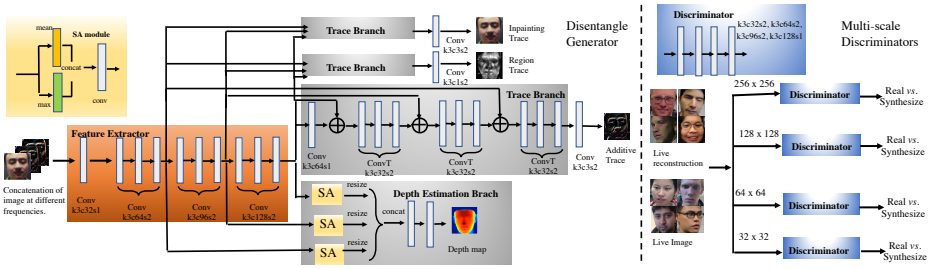

Fig. 5: The detailed architecture of PhySTD. The overall architecture contains Disentangle Generator and Multi-scale Discriminators. Notably, in the architecture of PhySTD, each convolutional layer is followed by Batch Normalization layer, RELU activation function and Dropout. This level of details is not included here.

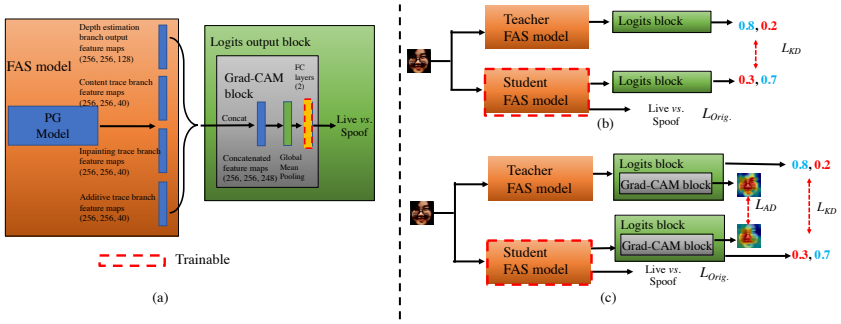

Fig. 6: In (a), we modify the pre-trained FAS model into a binary classifier. In (b) and (c), we modified architectures for LwF and LwM methods.

4 Experimental Implementation

4.1 PhySTD method details

In the experimental section, we apply FAS-*wrapper* on PhySTD for the analysis in Sec. 5.2. We depict the details of PhySTD in the Fig. 5, and more can be found in the original work [8].

4.2 The implementation details of prior methods

We are the first work that studies MD-FAS, in which no source data being available during the model updating process. To the best of our knowledge, there does not exist FAS works in such a source-free scenario. Therefore, in order to have a fair comparison, we need to implement methods from other topics (*e.g.*, *anti-forgetting learning* and *multi-domain learning*) on FAS dataset. In this section, we explain our implementation details on prior methods.

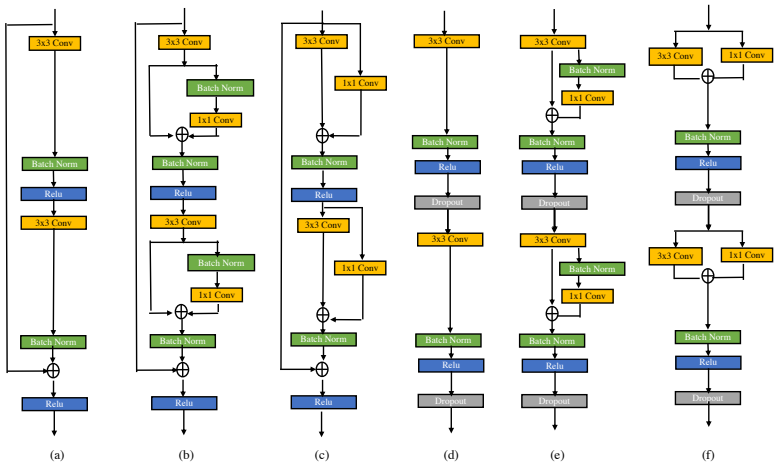

Fig. 7: Based on the RseNet building block (a), [10,11] have proposed RseNet modified building blocks in (b) and (c) for learning multiple domain knowledge. Likewise, given two consecutive building blocks in PhySTD, we construct modified building blocks, based on [10,11], in (e) and (f).

The implementation details of prior anti-forgetting methods We compare our methods to prior works that have anti-forgetting mechanism: LwF [7], LwM [4] and MAS [2]. Firstly, we pre-train the FAS model that is based on PhySTD on the source domain dataset. After the pre-training, we concatenate output feature maps generated from the last convolution layer in different branches as a new concatenated feature maps. Then we feed such feature maps through Global Average Pooling Layer and a fully-connected (FC) layer, such that we can obtain a binary classifier. The details are depicted in Fig. 6(a). We fix the pre-trained FAS model weights and train the last FC layer. As a result, we can use concatenated feature maps for a binary classification result indicating spoofness. We denote newly-added layers as the Logits block, part of which generates the class activate map is denoted as Grad-CAM block. We use these two blocks with the original FAS model for implementing LwF and LwM, as illustrated in Fig. 6(b)(c). In terms of MAS [2], we apply the publicly available source code [2] on the binary classifier we construct, without significantly changing the architecture. **The implementation details of multi-domain learning methods** Seri. Res-Adapter [10], and Para. Res-Adapter [11] are proposed for learning knowledge in multiple visual domains. Specifically, they use domain-specific adapter to enhance model ability in learning a universal image representation for multiple domains. They design such an idea on ResNet [6], which can be seen in Fig. 7. Based on the same idea, we modify the building block in PhySTD for learning the new domain knowledge. Notably, we have examine different adapter architectures, such as convolution filter with kernel size 1×1, 3×3, 5×5 and 7×7 , and find that 1×1 convolution offers the best

[2] https://github.com/rahafaljundi/MAS-Memory-Aware-Synapses

FAS performance. We also consider the publicly available source code [3] as the reference for the implementation.

References

1. international organization for standardization. Iso/iec jtc 1/sc 37 biometrics: Information technology biometric presentation attack detection part 1: Framework. In: https://www.iso.org/obp/ui/iso., accessed: 2022-03-3
2. Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., Tuytelaars, T.: Memory aware synapses: Learning what (not) to forget. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 139–154 (2018)
3. Asnani, V., Yin, X., Hassner, T., Liu, X.: Malp: Manipulation localization using a proactive scheme. In: CVPR (2023)
4. Dhar, P., Singh, R.V., Peng, K.C., Wu, Z., Chellappa, R.: Learning without memorizing. In: CVPR (2019)
5. Guo, X., Liu, X., Ren, Z., Grosz, S., Masi, I., Liu, X.: Hierarchical fine-grained image forgery detection and localization. In: CVPR (2023)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
7. Li, Z., Hoiem, D.: Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence **40**(12), 2935–2947 (2017)
8. Liu, Y., Liu, X.: Physics-guided spoof trace disentanglement for generic face anti-spoofing. arXiv preprint arXiv:2012.05185 (2020)
9. Liu, Y., Stehouwer, J., Jourabloo, A., Liu, X.: Deep tree learning for zero-shot face anti-spoofing. In: CVPR (2019)
10. Rebuffi, S.A., Bilen, H., Vedaldi, A.: Learning multiple visual domains with residual adapters. arXiv preprint arXiv:1705.08045 (2017)
11. Rebuffi, S.A., Bilen, H., Vedaldi, A.: Efficient parametrization of multi-domain deep neural networks. In: CVPR (2018)
12. Wang, C.Y., Lu, Y.D., Yang, S.T., Lai, S.H.: Patchnet: A simple face anti-spoofing framework via fine-grained patch recognition. In: CVPR (2022)

[3] https://github.com/srebuffi/residual_adapters