EyePAD++ : Supplementary material

Prithviraj Dhar[1], Amit Kumar[2], Kirsten Kaplan[2], Khushi Gupta[2], Rakesh Ranjan[2], Rama Chellappa[1]
[1]Johns Hopkins University, [2]Reality Labs, Meta
{pdhar1,rchella4}@jhu.edu, {akumar14,kkaplan,khushigupta,rakeshr}@fb.com

In this supplementary material, we provide the following information:
Section A1: Train and test split for user-to-user verification.
Section A2: Hyperparameters for EyePAD and EyePAD++.
Section A3: Detailed results with HRnet and MobilenetV3 backbones.
Section A4: Ablation experiments for λ_1 (EyePAD).
Section A5: Hyperparameters for baseline methods.
SectionA6: Issues with pre-processing.

A1. Train and test splits for ND-Iris-0405 dataset [1]

In section 4.2 of the main paper, we mention that we randomly split the users into two subsets: U_{train} (for training) and U_{test} (for evaluation). All the images for users in the training split are used for training. Here, we provide the train and test split to enable researchers reproduce our protocol.

Train user IDs: '04200' '04203' '04214' '04233' '04239' '04261' '04265' '04267' '04284' '04286' '04288' '04302' '04309' '04313' '04320' '04327' '04336' '04339' '04349' '04351' '04361' '04370' '04378' '04379' '04382' '04387' '04394' '04395' '04397' '04400' '04407' '04408' '04409' '04418' '04419' '04429' '04430' '04434' '04435' '04436' '04440' '04446' '04447' '04453' '04460' '04471' '04472' '04475' '04476' '04477' '04479' '04481' '04482' '04485' '04495' '04496' '04502' '04504' '04505' '04506' '04511' '04512' '04514' '04530' '04535' '04542' '04553' '04560' '04575' '04577' '04578' '04581' '04587' '04588' '04589' '04593' '04596' '04597' '04598' '04603' '04605' '04609' '04613' '04615' '04622' '04626' '04628' '04629' '04632' '04633' '04634' '04644' '04647' '04653' '04670' '04684' '04687' '04691' '04692' '04695' '04699' '04701' '04702' '04703' '04712' '04715' '04716' '04720' '04721' '04725' '04729' '04734' '04736' '04737' '04738' '04742' '04744' '04745' '04747' '04748' '04751' '04756' '04757' '04758' '04763' '04765' '04768' '04772' '04773' '04774' '04776' '04777' '04778' '04782' '04783' '04785' '04787' '04790' '04792' '04794' '04797' '04801' '04802' '04803' '04813' '04815' '04816' '04818' '04831' '04832' '04839' '04840' '04841' '04843' '04846' '04847' '04850' '04854' '04857' '04858' '04859' '04861' '04863' '04864' '04866' '04867' '04869' '04870' '04871' '04872' '04873' '04876' '04877' '04878' '04879' '04880' '04882' '04883' '04884' '04886' '04888' '04890' '04891' '04892' '04894' '04897' '04898' '04899' '04901' '04905' '04908' '04909' '04910' '04911' '04912' '04914' '04915' '04919' '04920' '04922' '04923' '04928' '04930' '04931' '04932' '04934'

Test user IDs:'02463' '04201' '04202' '04213' '04217' '04221' '04225' '04273' '04285' '04297' '04300' '04301' '04311' '04312' '04314' '04319' '04322' '04324' '04334' '04338' '04341' '04343' '04344' '04347' '04350' '04372' '04385' '04388' '04404' '04423' '04427' '04444' '04449' '04451' '04456' '04459' '04461' '04463' '04470' '04473' '04488' '04493' '04507' '04509' '04513' '04519' '04531' '04537' '04556' '04557' '04569' '04580' '04585' '04595' '04600' '04612' '04621' '04631' '04641' '04662' '04664' '04667' '04673' '04675' '04681' '04682' '04683' '04689' '04693' '04697' '04705' '04708' '04709' '04711' '04714' '04719' '04722' '04724' '04726' '04727' '04728' '04730' '04731' '04732' '04733' '04743' '04746' '04749' '04754' '04760' '04762' '04767' '04770' '04775' '04784' '04786' '04796' '04798' '04806' '04810' '04811' '04812' '04821' '04822' '04823' '04827' '04829' '04830' '04833' '04838' '04842' '04848' '04849' '04851' '04853' '04855' '04856' '04860' '04862' '04865' '04868' '04874' '04875' '04881' '04885' '04887' '04889' '04893' '04895' '04896' '04900' '04902' '04903' '04904' '04906' '04907' '04913' '04916' '04917' '04918' '04921' '04924' '04925' '04926' '04927' '04929' '04933' '04935' '04936'

It is also mentioned in the main paper that the images for the test users are then randomly divided into query and gallery sets. We provide the images in the query and gallery sets in `query_set.txt` and `gallery_set.txt`, respectively. These files are provided in the supplementary material zip file. We also provide a readme file (`README.txt`) for the readers' convenience, wherein we provide information about the left/right labels.

	User-to-user verification results on ND-Iris-0405 (EA)												PAD results on CU-LivDet			
	1 Query 1 Gallery				1 Query 2 Gallery				1 Query 5 Gallery							
Method	OFRR(↓)	10^{-4}	10^{-3}	10^{-2}	OFRR(↓)	10^{-4}	10^{-3}	10^{-2}	OFRR(↓)	10^{-4}	10^{-3}	10^{-2}	TDR(↑)	APCER	BPCER	HTER(↓)
EA only	-	0.861	0.950	0.990	-	0.875	0.961	0.994	-	0.919	0.983	0.996	-	-	-	-
PAD only	-	-	-	-	-	-	-	-	-	-	-	-	0.962	0.051	0.00	0.026
MTL	0.209	0.682	0.819	0.949	0.156	0.734	0.866	0.962	0.113	0.815	0.930	0.977	0.921	0.053	0.005	0.029
MTMT [3]	0.132	0.777	0.881	0.970	0.091	0.840	0.917	0.971	0.068	0.891	0.945	0.984	0.959	0.033	0.001	0.017
EyePAD	0.094	0.804	0.909	0.985	0.060	0.864	0.942	0.993	0.034	0.898	0.968	0.995	0.934	0.044	0.014	0.029
EyePAD++	**0.062**	0.865	0.942	0.996	**0.048**	0.898	0.959	0.993	**0.031**	0.926	0.976	0.997	0.915	0.041	0.021	0.031
EA only	-	0.829	0.913	0.978	-	0.838	0.944	0.988	-	0.911	0.968	0.992	-	-	-	-
PAD only	-	-	-	-	-	-	-	-	-	-	-	-	0.801	0.089	0.026	0.058
MTL	0.430	0.503	0.650	0.851	0.377	0.509	0.719	0.868	0.293	0.638	0.801	0.924	0.737	0.040	0.331	0.186
MTMT [3]	0.188	0.768	0.898	0.973	0.165	0.846	0.932	0.990	0.129	0.904	0.963	0.994	0.646	0.125	0.046	0.086
EyePAD	0.180	0.805	0.897	0.970	**0.137**	0.877	0.932	0.989	0.132	0.902	0.960	0.993	0.766	0.115	0.008	0.062
EyePAD++	**0.174**	0.830	0.911	0.983	0.158	0.869	0.950	0.988	**0.118**	0.915	0.971	0.989	0.655	0.109	0.025	0.067
EA only	-	0.733	0.893	0.984	-	0.749	0.924	0.990	-	0.837	0.952	0.992	-	-	-	-
PAD only	-	-	-	-	-	-	-	-	-	-	-	-	0.942	0.049	0.008	0.029
MTL	0.338	0.451	0.678	0.856	0.279	0.481	0.737	0.908	0.236	0.587	0.787	0.940	0.899	0.085	0.004	0.045
MTMT [3]	0.184	0.675	0.846	0.944	0.168	0.721	0.862	0.961	0.114	0.824	0.916	0.975	0.894	0.048	0.020	0.034
EyePAD	0.161	0.656	0.848	0.964	0.125	0.718	0.886	0.984	**0.091**	0.800	0.916	0.986	0.914	0.060	0.006	0.033
EyePAD++	**0.157**	0.729	0.869	0.976	**0.114**	0.781	0.916	0.988	0.093	0.840	0.937	0.989	0.887	0.061	0.010	0.036

Table A1. EA and PAD with HRnet64 trained and evaluated on **(top)** original, **(middle)** blurred, **(bottom)** Noisy data: For user-to-user verification, we report TAR@FAR=$10^{-4}, 10^{-3}, 10^{-2}$. For PAD, we report TDR@FDR=0.002 and APCER, BPCER, HTER. OFRR jointly measures EA and PAD performance on ND-Iris-0405. *EyePAD++ obtains the lowest OFRR in most scenarios*. **Bold**: Best, Underlined: Second best.

	User-to-user verification results on ND-Iris-0405 (EA)												PAD results on CU-LivDet			
	1 Query 1 Gallery				1 Query 2 Gallery				1 Query 5 Gallery							
Method	OFRR(↓)	10^{-4}	10^{-3}	10^{-2}	OFRR(↓)	10^{-4}	10^{-3}	10^{-2}	OFRR(↓)	10^{-4}	10^{-3}	10^{-2}	TDR(↑)	APCER	BPCER	HTER(↓)
EA only	-	0.824	0.923	0.989	-	0.863	0.943	0.987	-	0.898	0.952	0.995	-	-	-	-
PAD only	-	-	-	-	-	-	-	-	-	-	-	-	0.925	0.048	0.010	0.029
MTL	0.180	0.739	0.856	0.958	0.142	0.792	0.895	0.970	0.110	0.872	0.933	0.985	0.884	0.025	0.053	0.039
MTMT [3]	0.193	0.751	0.871	0.973	0.144	0.817	0.917	0.977	0.126	0.859	0.933	0.987	0.793	0.073	0.011	0.042
EyePAD	0.160	0.769	0.893	0.972	0.142	0.838	0.919	0.985	0.114	0.887	0.947	0.991	0.859	0.046	0.034	0.040
EyePAD++	**0.140**	0.819	0.904	0.981	**0.117**	0.863	0.934	0.992	**0.085**	0.901	0.962	0.990	0.883	0.041	0.022	0.032
EA only	-	0.889	0.953	0.993	-	0.853	0.929	0.992	-	0.846	0.921	0.989	-	-	-	-
PAD only	-	-	-	-	-	-	-	-	-	-	-	-	0.581	0.155	0.078	0.117
MTL	0.564	0.585	0.734	0.871	0.533	0.619	0.798	0.927	0.483	0.744	0.855	0.942	0.556	0.133	0.122	0.128
MTMT [3]	0.524	0.642	0.819	0.939	0.477	0.726	0.905	0.965	0.464	0.769	0.913	0.963	0.502	0.227	0.046	0.137
EyePAD	0.562	0.537	0.715	0.915	0.486	0.618	0.810	0.952	0.447	0.711	0.866	0.956	0.589	0.144	0.097	0.121
EyePAD++	**0.385**	0.768	0.874	0.956	**0.372**	0.792	0.913	0.970	**0.332**	0.861	0.938	0.983	0.552	0.092	0.173	0.133
EA only	-	0.756	0.872	0.977	-	0.795	0.912	0.982	-	0.817	0.944	0.985	-	-	-	-
PAD only	-	-	-	-	-	-	-	-	-	-	-	-	0.831	0.079	0.003	0.041
MTL	0.263	0.643	0.799	0.949	0.217	0.682	0.870	0.976	0.173	0.761	0.908	0.974	0.762	0.062	0.065	0.064
MTMT [3]	0.285	0.549	0.766	0.928	0.185	0.705	0.868	0.968	0.162	0.777	0.906	0.977	0.730	0.049	0.069	0.059
EyePAD	0.262	0.722	0.847	0.958	0.227	0.779	0.880	0.974	0.209	0.801	0.927	0.980	0.712	0.083	0.047	0.065
EyePAD++	**0.254**	0.660	0.786	0.947	**0.180**	0.733	0.868	0.975	**0.137**	0.811	0.912	0.990	0.730	0.033	0.126	0.080

Table A2. EA and PAD with MobilenetV3 trained and evaluated on **(top)** original, **(middle)** blurred, **(bottom)** Noisy data: For user-to-user verification, we report TAR@FAR=$10^{-4}, 10^{-3}, 10^{-2}$. For PAD, we report TDR@FDR=0.002 and APCER, BPCER, HTER. OFRR jointly measures EA and PAD performance on ND-Iris-0405. *EyePAD++ obtains the lowest OFRR*. **Bold**: Best, Underlined: Second best.

A2. Training details for EyePAD, EyePAD++

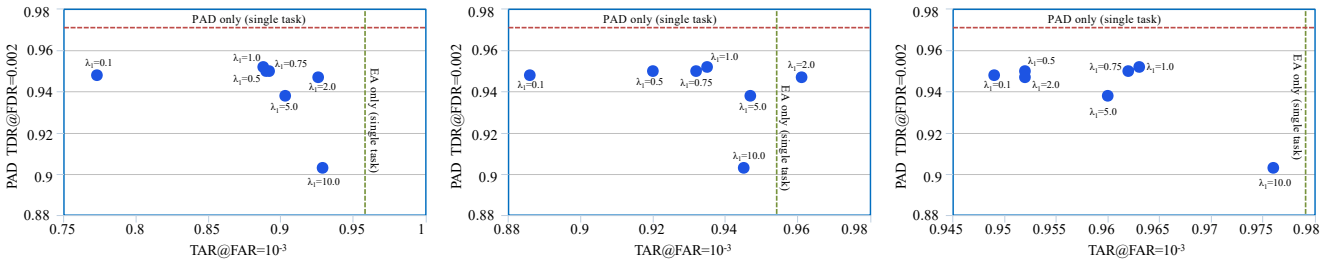
We train all the models in our work with a batch size of 64 in our experiments, for 100 epochs. We use data augmentation such as random horizontal flip, random rotation (30 degrees) and random jitter. The detailed hyperparameter information for training models for user-to-user verification with PAD is provided in Table A3.

While training Densenet121 network for eye-to-eye verification with PAD, we use $\lambda_1 = 2.0$ (for EyePAD) and $\lambda_2 = 2.0$ (for EyePAD++). All the other parameters used in this experiment are same as those mentioned in the first row of Table A3.

A3. Detailed results

A3.1. EA and PAD with HRnet64

In Table A1, we provide the full version of Table 4 from the main paper, where we present results with HRnet64 [4]. Here, we report the user-to-user verification performance



(a) 1 query 1 gallery (b) 1 query 2 gallery (c) 1 query 5 gallery

Figure A1. PAD performance (TDR@FDR=0.002) v/s User-to-user verification performance (TAR@FAR=10^{-3}) obtained by the EyePAD student network, for different values of λ_1.

Backbone	Dataset	λ_1	λ_2	Optimizer	LR	γ	Decay after
Densenet121	Original	2.0	0.75	Adam	10^{-4}	0.5	12
Densenet121	Blurred	1.0	2.0	Adam	10^{-4}	0.5	12
Densenet121	Noisy	1.0	2.0	Adam	10^{-4}	0.5	12
HRnet64	Original	2.0	2.0	Adam	10^{-4}	0.5	12
HRnet64	Blurred	5.0	2.0	Adam	10^{-4}	0.5	12
HRnet64	Noisy	2.0	5.0	Adam	10^{-4}	0.5	12
MobilenetV3	Original	1.0	0.75	SGD	10^{-1}	0.1	15
MobilenetV3	Blurred	1.0	0.75	SGD	10^{-1}	0.1	15
MobilenetV3	Noisy	5.0	2.0	SGD	10^{-1}	0.1	15

Table A3. Hyperparameter information for EyePAD and EyePAD++.

Backbone	Dataset	λ_{auth}	λ_{pad}	Optimizer	LR	γ	Decay after
Densenet121	Original	1.0	1.0	Adam	10^{-4}	0.5	12
Densenet121	Blurred	0.75	0.75	Adam	10^{-4}	0.5	12
Densenet121	Noisy	0.5	0.75	Adam	10^{-4}	0.5	12
HRnet64	Original	1.0	1.0	Adam	10^{-4}	0.5	12
HRnet64	Blurred	0.75	0.75	Adam	10^{-4}	0.5	12
HRnet64	Noisy	2.0	2.0	Adam	10^{-4}	0.5	12
MobilenetV3	Original	1.0	2.0	SGD	10^{-1}	0.1	15
MobilenetV3	Blurred	1.0	1.0	SGD	10^{-1}	0.1	15
MobilenetV3	Noisy	0.1	0.1	SGD	10^{-1}	0.1	15

Table A4. Hyperparameter information for MTMT [3].

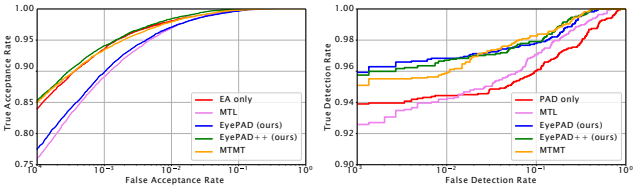
with $K = 1, 2, 5$ gallery pairs. In most of the scenarios, *EyePAD++ outperforms the existing baselines in terms of the OFRR score*.

A3.2. EA and PAD with MobilenetV3

In Table A2, we provide the full version of Table 5 from the main paper, where we present results with MobilenetV3 [2]. Here, we report the user-to-user verification performance with $K = 1, 2, 5$ gallery pairs. *EyePAD++ outperforms the existing baselines in terms of the OFRR score*, in all the problem settings.

A3.3. Eye-to-eye verification with PAD

In Table 6 of the main paper, we provide the eye-to-eye verification and PAD performance for EyePAD, EyePAD++ and our baselines (i.e. EA-only model, PAD-only model, MTL, MTMT[3]). Here, in Fig. A2, we provide the ROC plots for eye-to-eye verification and PAD, generated using these methods.



(a) Eye-to-eye verification (b) PAD

Figure A2. ROC curves for (a)EA performance on ND-Iris-0405 (b) PAD performance on CU-LivDet

A4. Ablation study: Effect of λ_1 in EyePAD

The hyperparameter λ_1 is used to weight the feature-level distillation loss L_{dis} (Eq 2 of the main paper). L_{dis} is used to preserve the EA information, while student M_s (initialized with M_t) is trained for PAD. Using Densenet121 and the original (clean) EA and PAD datasets, we analyze the effect of λ_1 on user-to-user verification performance and PAD performance. We perform experiments with $\lambda_1 = [0.1, 0.5, 0.75, 1.0, 2.0, 5.0, 10.0]$ and present the corresponding results in Fig. A1. We find that in general, when λ_1 increases, the user to user verification performance improves and the PAD performance gets degraded. This is expected because a higher value for λ_1 enforces M_s to preserve authentication and restrics it from learning PAD-specific features. However, we do not find any such trend with respect to parameter λ_2.

A5. Training details for baselines

For training the MTL baseline for user-to-user or eye-to-eye verification with PAD, we use the same parameter values mentioned in Table A3, except for λ_1, λ_2. The hyperparameters used for training MTMT [3] are provided in Table A4. While training MTMT [3] for eye-to-eye verification with PAD (using Densenet121 and the original dataset), we use $\lambda_{auth} = 1.0$ and $\lambda_{pad} = 1.0$. The rest of the hyper-

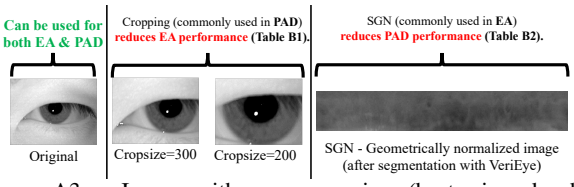


Figure A3. Issues with pre-processing (best viewed when zoomed).

Cropsize/FAR	EER(↓)	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}
200×200	2.96	0.458	0.620	0.804	0.936	0.991
300×300	1.59	0.571	0.741	0.890	0.975	0.998
Original (No preprocess)	**1.36**	**0.729**	**0.839**	**0.936**	**0.980**	**0.998**

Table A5. Eye-to-eye verif^{n} with original and cropped (pre-processed) images, using Densenet121-based EA-only network.

Pre-processing	TDR(↑)	APCER(↓)	BPCER(↓)	HTER(↓)
Geometrically normalized	87.26	10.11	0.60	5.36
Original(no preprocessing)	**94.02**	**5.96**	**0.02**	**2.99**

Table A6. PAD performance with Densenet121-based PAD-only network (using official split of CU-LivDet-2017).

parameters are same as those mentioned in the first row of Table A4.

A6. Issues with pre-processing

In Section 1 of the main paper, we mentioned that pre-processing steps potentially make the EA and PAD pipelines computationally expensive. Here, we provide more information to explain why pre-processing cannot be used to jointly perform EA and PAD:

(1) Most EA-only frameworks perform 'Segmentation + Geometric Normalization (SGN)' for preprocessing using third party softwares like VeriEye (Table 1 in main paper). However, our goal is to jointly perform both EA and PAD. So, we perform SGN using VeriEye; and train/test a PAD-only network with normalized images, but find that *SGN significantly reduces the PAD performance* (Table A6). Hence, we cannot use SGN for our EA+PAD frameworks.

(2) The main application of our work is EA and PAD in wearable headsets, where pre-processing tools such as VeriEye cannot be easily integrated due to sensory limitations. Furthermore, we find that VeriEye takes 114.8 ms to normalize 1 image, on top of the time taken by the network to process the input image.

(3) Several PAD-only frameworks detect the iris region for pre-processing. We train and test an EA-only network using the detected iris region (with crop size of 200 and 300, Fig. A3). But, in Table A5 we find that *this step (i.e. detection) reduces the EA performance quite significantly*, and hence cannot be used to jointly perform EA and PAD.

References

[1] Kevin W Bowyer and Patrick J Flynn. The ND-IRIS-0405 iris image dataset. *arXiv preprint arXiv:1606.04853*, 2016. 1

[2] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019. 3

[3] Wei-Hong Li and Hakan Bilen. Knowledge distillation for multi-task learning. In *European Conference on Computer Vision*, pages 163–176. Springer, 2020. 2, 3

[4] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 2