Appendix

A Implementation

We published our code on https://github.com/AnonymWriter/Iris-Style-Transfer for reproducibility. The repository is anonymized for review.

B Randomness

To guarantee reproducibility, we fixed random seed to 42 at the very beginning of our program. The random seed was fed to *numpy.random.seed*, *torch.manual_seed*, and *random.seed*.

C Datasets

The OpenEDS2019 dataset [Garbin et al. 2019] contains four parts, namely: segmentation data, generative data, sequence data, and corneal topography data. We considered only the segmentation data. The OpenEDS2020 dataset [Palmero et al. 2020, 2021] contains two subsets, respectively for gaze estimation and sparse eye segmentation. We used only the gaze estimation set. The details of both datasets are given in Table 5.

Table 5. Overview of the OpenEDS2019 and OpenEDS2020 datasets.

Property	OpenEDS2019 (segmentation)	OpenEDS2020 (estimation)
samples	12,759	550,400
users	152	80
modality	grayscale image	grayscale image
resolution	400×640	640×400
ground truth	segmentation labels	3D gaze vectors

D Environment

Our environment by date February 1st, 2025 was listed in Table 6. It should be noticed that although our code can be run on a device without dedicated GPU, it is recommended to run the program on a computer with $\geq$ 32GB RAM and a dGPU with $\geq$ 32GB VRAM.

Table 6. Experiment environment.

Hardware	Specification
CPU	AMD EPYC 7763 64-Core
GPU	NVIDIA A100 80GB PCIe$\times$4
Memory	1TB

Software	Version
OS	Ubuntu 22.04.5 LTS
Python	3.12.7 by Anaconda
PyTorch	2.6.0 for CUDA 12.6
OpenCV-Python	4.11.0.86
Scikit-Learn	1.6.1
Scikit-Image	0.25.1
segmentation-models-pytorch	0.4.0
WandB	0.19.5

E Hyperparameters

For training the classifiers, we used the Adam optimizer and conducted grid search in range $\{1e-6, 1e-5, 1e-4, 1e-3, 1e-2\}$ for optimal learning rates. The best performing learning rates were $1e-5$ for both feature sets. For style transfer, we followed the instruction given by [Gatys 2017], namely using an LBFGS optimize with learning 1.0. We used a batch size of 64 for the aforementioned tasks. For training gaze estimators, similar to fitting the classifiers, we applied the Adam optimizer with optimal learning rate $1e-5$ for both estimation models. Batch size of 128 was picked during training. The hyperparameters were listed in Table 7 for a clear view.

Table 7. Details of hyperparameters.

Hyperparameter	**Value**
optimizer (classifiers)	Adam
optimizer (style transfer)	LBFGS
optimizer (gaze estimators)	Adam
learning rate (classifiers)	$1e-5$
learning rate (style transfer)	1.0
learning rate (gaze estimators)	$1e-5$
batch size (classifiers)	64
batch size (style transfer)	64
batch size (gaze estimators)	128
image scaling	224×224

F Model Structures

We implemented two different classification heads. The first one was designed for conventional VGG features, whereas the second was tailor for style features. We designed the structures of both heads so that they resemble the original VGG projection head as much as possible. For the gaze estimators, the projection heads were regression MLPs, which output normalized 3D gaze vectors. For detailed architectures of the models, we refer readers to https://github.com/AnonymWriter/Iris-Style-Transfer/tree/main/models.

G Additional Results

Additional results, such as F1 scores, Matthews correlation coefficients (MCC), and distribution plots, were given here.

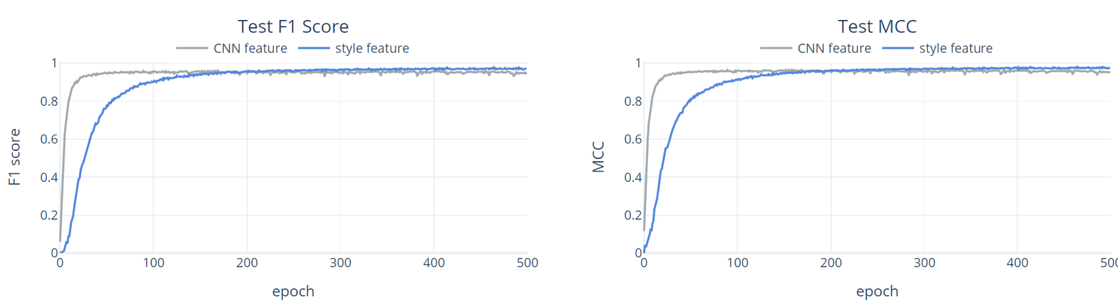

Fig. 8. Test performance of style feature and common CNN feature for iris recognition.

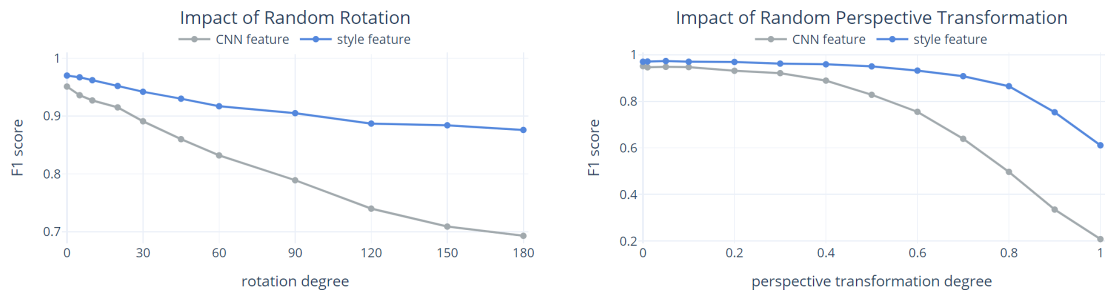

Fig. 9. Style feature vs. common CNN feature regarding robustness against random rotation and perspective transformation (F1 score).

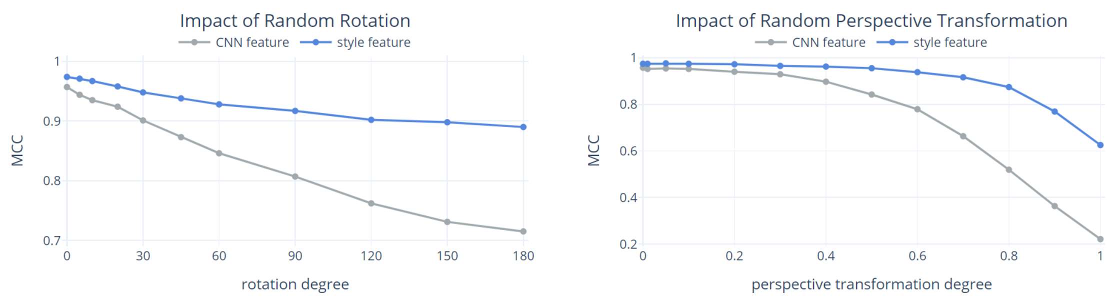

Fig. 10. Style feature vs. common CNN feature regarding robustness against random rotation and perspective transformation (MCC).

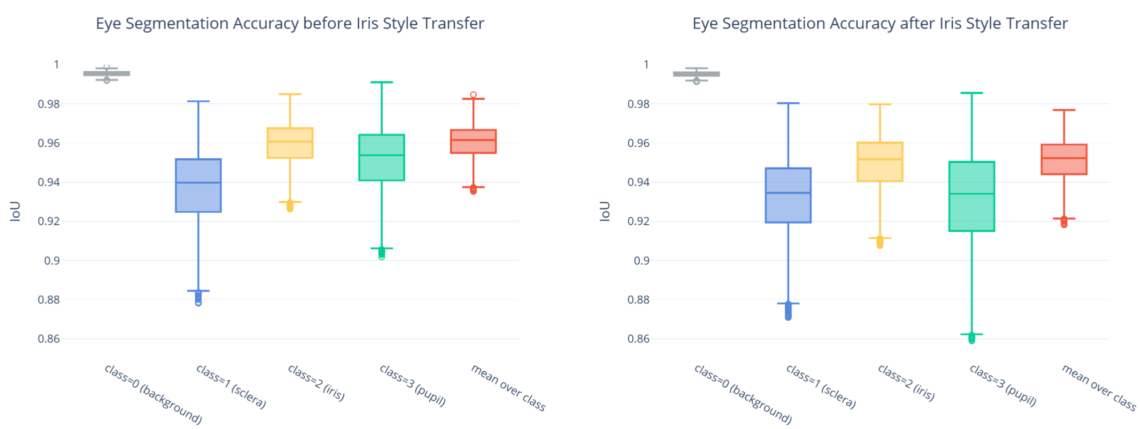

Fig. 11. Influence of iris style transfer on eye segmentation accuracy (distribution over test set).

Received 1 November 2024; revised XXX; accepted XXX

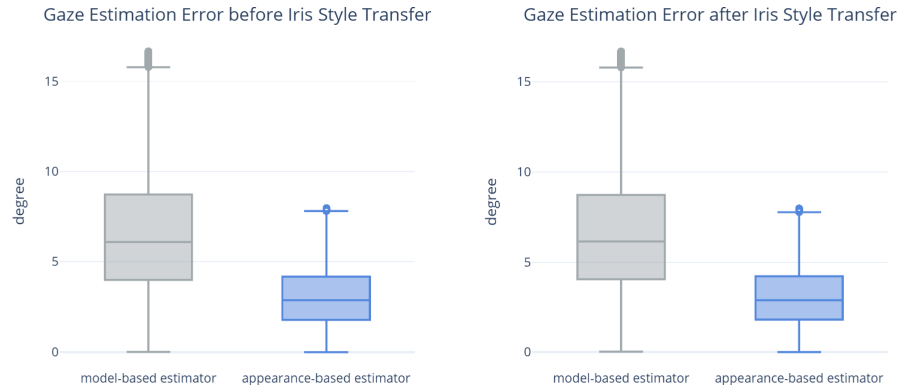

Fig. 12. Influence of iris style transfer on gaze estimation error (distribution over test set).