Supplementary Materials: CodeSwap: Symmetrically Face Swapping Based on Prior Codebook

A Implementation Details

All our experiments are implemented by Pytorch [6]. Our training process is divided into three stages, and for each stage we use Adam optimizer [5] with β_1 of 0.9 and β_2 of 0.999. The learning rate is 10^{-4} for the generator and 10^{-5} for the discriminator. We employ the batch size of 32 on two NVIDIA A100 GPUs.

A.1 Architecture of Stage 1

In the Stage 1, we train the encoder, decoder, and the prior codebook through a large amount of natural faces. We refer to the encoder and decoder design of [2]. The structure is shown in Figure 1, where the encoder and the decoder utilize a symmetrical design. In addition, we adopt the attention mechanism at the scale of 16, which helps to improve the generalization of the model. So in the regional control swap task, even if only part of the code is manipulated, a natural face image can be generated. The codebook is implemented by a set of randomly initialized learnable parameters, with the number of codes set to 1024 and the dimension of the codes to 256.

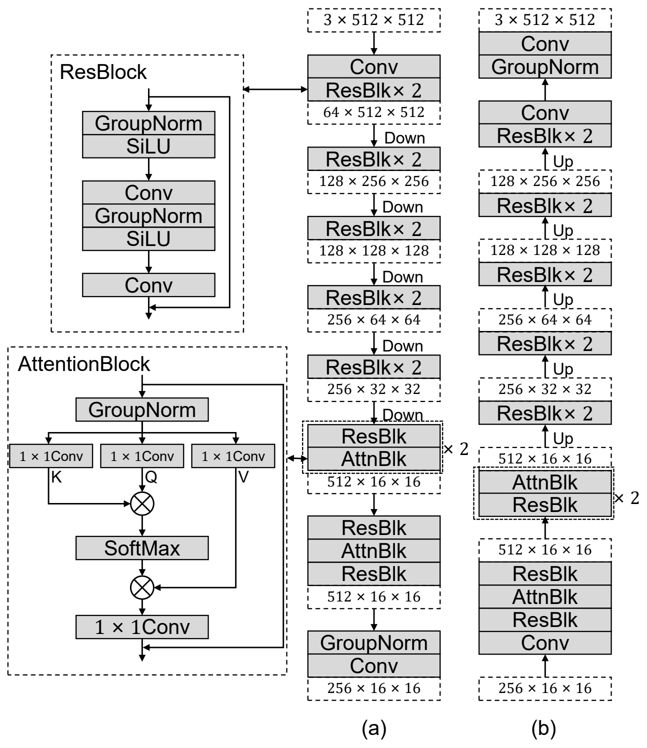

Figure 1: The detailed structure of our encoder and decoder. (a) encoder, (b) decoder.

A.2 Architecture of Stage2

In the Stage 2, we design a Transformer-based Global Fusion Module and a code selector consisting of a classifier, which is shown in Figure 2 (a). The encoder $\mathcal{E}$ and decoder $\mathcal{D}$ are trained in Stage 1 and frozen in this Stage. For the Global Fusion Module, we adopt the learnable positional coding to capture the positional information between different codes [3]. The overall structure of Stage 2 is similar to the Vision Transformer [3], but instead of having additional class token, we pass all codes through the classifier to obtain probability vectors for each code to achieve code manipulation.

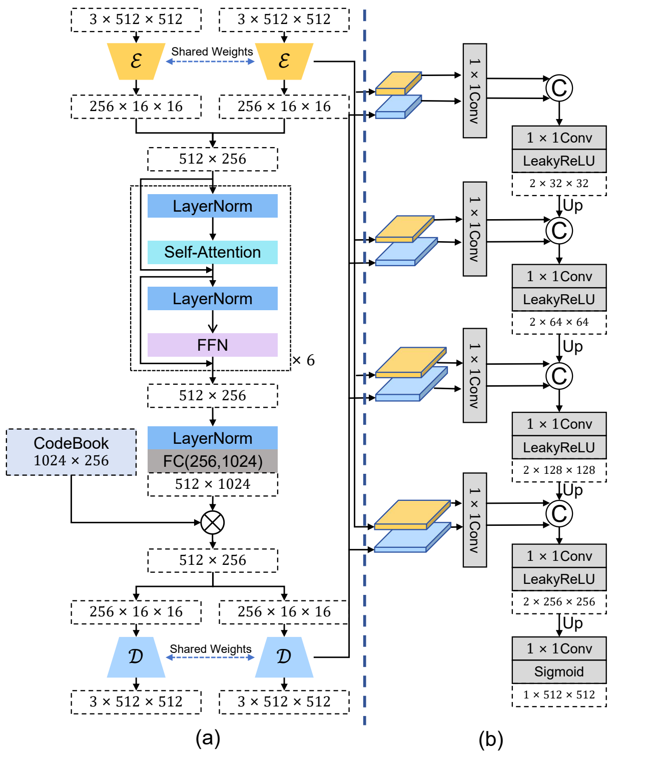

Figure 2: The detailed structure of framework. (a) backbone network for face swapping, (b) mask generator.

A.3 Architecture of Stage 3

The detail of our mask generator is shown in Figure 2 (b). Inspired with [1, 7] that utilize the features of StyleGAN's [4] middle layer to generate masks, we utilize the capabilities of our pretrained encoder and decoder. Specifically, we extract features from the intermediate layers of the encoder and decoder with scales of 32, 64, 128, and 256. Each layer is processed through a convolutional layer, followed by an activation function and upsampling. The dimensions are progressively concatenated, and finally, a sigmoid

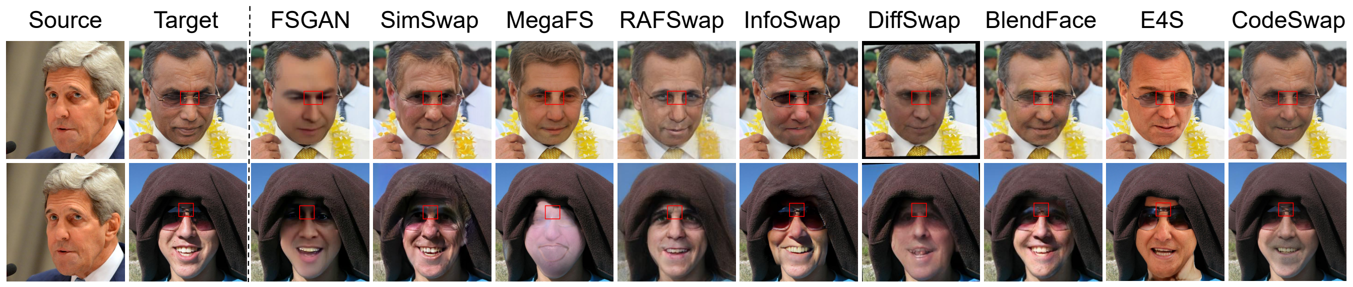

Figure 3: Comparison with existing methods with facial occlusion (Zoom in for details).

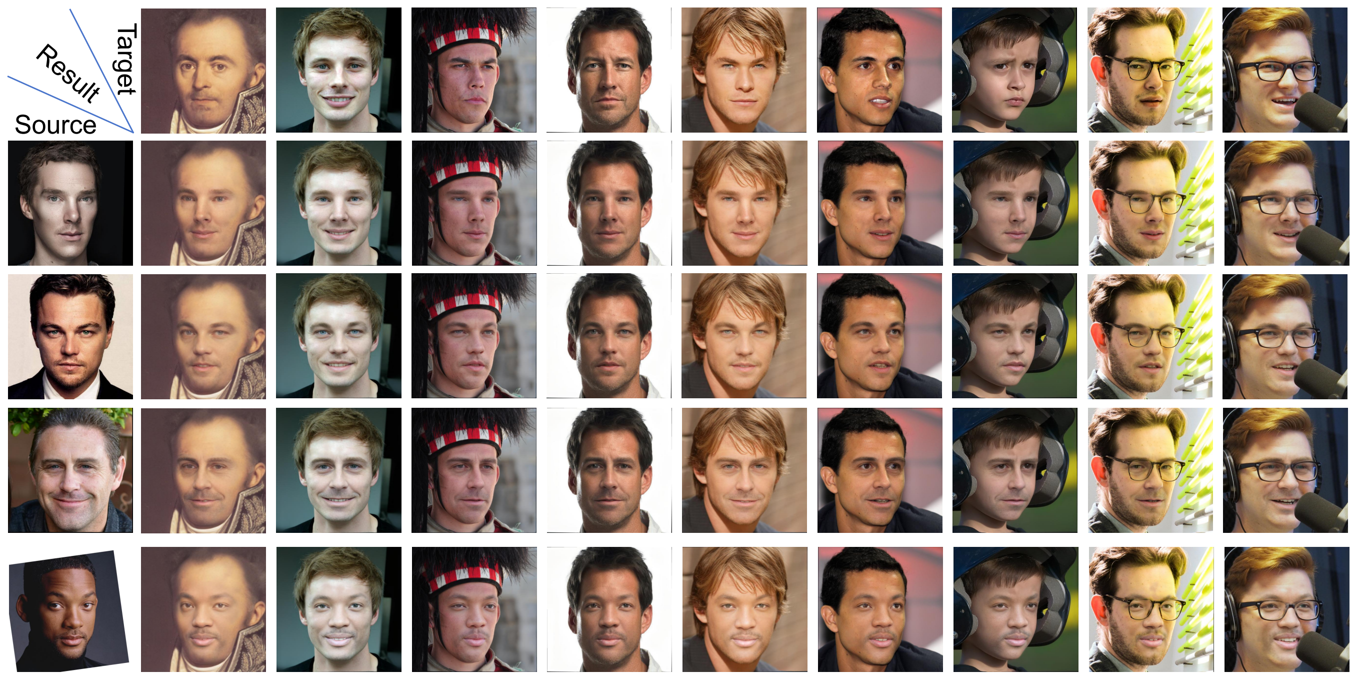

Figure 4: Face swapping between males. These images contain different lighting, poses, ages and occlusions (Zoom in for details).

function mapping the output values to the range between 0 and 1 to generate the mask.

B More Quantitative Results

In this section, we provide some qualitative comparison with existing methods with facial occlusion. What's more, we also provide more results of our method in the form of the image matrix to show our robustness.

B.1 Comparison with Facial Occlusion

We compare CodeSwap with existing methods to test the ability of our method to handle facial occlusion. As shown in Figure 3, our face swapping results still retain the occlusion in the red box better, which demonstrates the excellence of our method in dealing with facial occlusion.

B.2 More Swapping Results

We provide more face swapping results of our method in the form of image matrices, as shown in Figure 4 and Figure 5. These images contain different poses, lighting, ages, and occlusions. The results show that our method performs well in these difficult cases.

References

[1] Rameen Abdal, Peihao Zhu, Niloy J Mitra, and Peter Wonka. 2021. Labels4free: Unsupervised segmentation using stylegan. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 13970–13979.

[2] Sam Bond-Taylor, Peter Hessey, Hiroshi Sasaki, Toby P. Breckon, and Chris G. Willcocks. 2022. Unleashing Transformers: Parallel Token Prediction with Discrete Absorbing Diffusion for Fast High-Resolution Image Generation from Vector-Quantized Codes. In *European Conference on Computer Vision (ECCV)*.

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[4] Tero Karras, Samuli Laine, and Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference*

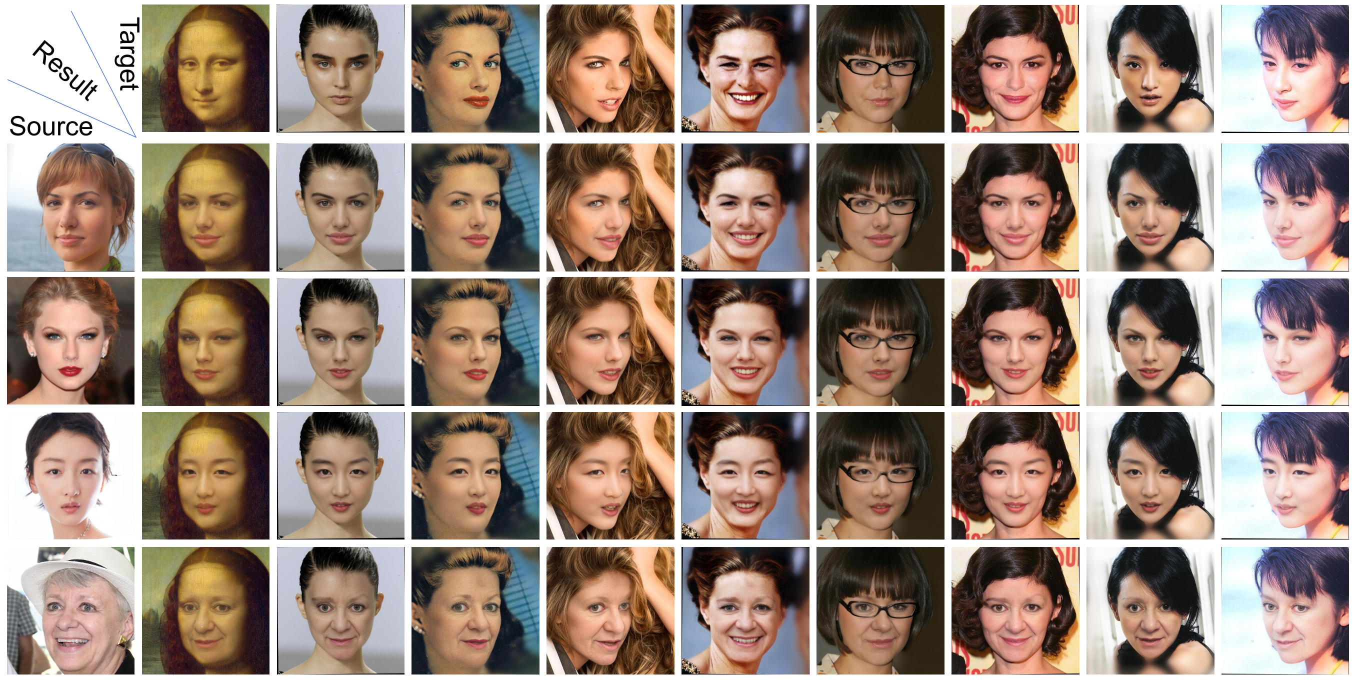

Figure 5: Face swapping between females. These images contain different lighting, poses, ages and occlusions (Zoom in for details).

on computer vision and pattern recognition. 4401–4410.

[5] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[6] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).

[7] Chao Xu, Jiangning Zhang, Miao Hua, Qian He, Zili Yi, and Yong Liu. 2022. Region-aware face swapping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 7632–7641.