Towards Robust Blind Face Restoration with Codebook Lookup Transformer

– Supplementary Material –

Shangchen Zhou **Kelvin C.K. Chan** **Chongyi Li** **Chen Change Loy**
S-Lab, Nanyang Technological University
{s200094, chan0899, chongyi.li, ccloy}@ntu.edu.sg
https://shangchenzhou.com/projects/CodeFormer

In this supplementary material, we provide additional discussions and results. In Sec. A, we present the further study on the designs of Codebook $\mathcal{C}$ and Transformer T. In Sec. B, we show more results on the extended tasks, including face color enhancement, face inpainting, and old photo restoration. Sec. C gives more restoration results on the real-world face images. In addition, we also present a video demo that contains some enhanced clips of old films.

A More Discussions on CodeFormer

A.1 Effect of the Number of Codebook

The proposed CodeFormer is affected by the reconstruction capability of learned codebook prior in Stage I. The better reconstruction capability of codebook, the greater the expressiveness of our method. We investigate the effect of the different numbers N of learned codebook for face image reconstruction. Table 1 shows the reconstruction results of LPIPS and PSNR on FFHQ dataset [4] when different numbers of codebooks are learned. The reconstruction performance is better as more codebook items are activated and learned. Thus, we adopt a larger codebook with 1024 items.

Table 1: Reconstruction results in terms of LPIPS and PSNR on FFHQ. The reconstruction is better as more codebook items are activated and learned.

Num. of Codebook N	$N = 384$	$N = 768$	$N = 1024$ **(ours)**
LPIPS↓	0.202	0.184	**0.175**
PSNR↑	22.59	23.04	**23.41**

A.2 Transformer Structure

The main novelty of our work is casting face restoration as a code token prediction task. Therefore, improving the accuracy of code prediction is significant in our method. The proposed CodeFormer employs a Transformer module to model the global interrelations of low-quality faces for better code prediction. Since the Transformer capability could be affected by its structure, we investigated different Transformer structures in pursuit of the best choice. Table 2 shows the accuracy of code prediction and LPIPS scores for the comparison of different structures.

Single-scale and Multi-scale Inputs. We design two different Transformer structures that respectively take the single-scale (16) and multi-scale $(16, 32, 64)$ features of the encoder as input. For multi-scale inputs, we first use a PixelUnshuffle layer to rearrange the features to the resolution of 16×16, and then utilize a Convolutional layer with a kernel of 1×1 size to reduce feature channel

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

to the same dimension. Finally, we add up all the reshaped multi-scale features and feed them to the Transformer T. As shown in Table 2, multi-scale input features do not boost the performance of code prediction, but slightly reduce the accuracy. An explanation is that degradation still exists in the larger-scale features of the shallow layers, which affects the accuracy of code prediction.

Number of Transformer Layer. We conduct an ablation study on Transformers with different numbers of layers. Table 2 shows that more Transformer layers can boost code prediction accuracy and enhance the restoration performance. But the number of Transformer layer beyond nine only leads to slight performance gains of code prediction, and the LPIPS score is not boosted. The ablation study confirms our choice of using nine Transformer layers in CodeFormer, which gives a better trade-off among the computational complexity and performance.

Table 2: Accuracy of code prediction (Accuracy↑) and restoration results (LPIPS↓) with different Transformer structures on the CelebA-Test dataset.

Structures	Multi-scale	Single-scale **(ours)**	4 layers	9 layers **(ours)**	15 layers
Accuracy↑	0.188	**0.192**	0.175	0.192	**0.193**
LPIPS↓	0.305	**0.307**	0.291	**0.307**	**0.307**

B More Results on Extensions

The proposed Codeformer can be easily extended to other tasks, including face color enhancement, face inpainting, and old photo enhancement. In this section, we compare our CodeFormer with state-of-the-art methods on face color enhancement (Sec. B.1) and face inpainting (Sec. B.2). Besides, we present the enhanced results of old historic photos as well (Sec. B.3) .

B.1 Face Color Enhancement

We finetune our model on face color enhancement using the same color augmentations (random color jitter and grayscale conversion) as GFP-GAN [6]. We evaluate and compare our method with GFP-GAN [6] (version 1)[1] on the real-world old photos (from CelebChild-Test dataset [6]) with color loss. The proposed CodeFormer generates high-quality face images with more natural color and faithful details.

Figure 1: Visual comparison with GFP-GAN [6] (version 1) that is jointly trained on face restoration and color enhancement as well.

[1] Version 1 of GFP-GAN is jointly trained on blind face restoration and color enhancement.

B.2 Face Inpainting

For face inpainting, we retrain the Codeformer on the synthetic paired data that generate masked faces by randomly drawing irregular polyline masks [7]. We compare our method with two state-of-the-art face inpainting methods of CTSDG [3] and GPEN [7] and give more results in Fig. 2. The proposed CodeFormer generates more natural face images without strokes and artifacts.

Figure 2: Visual comparison with state-of-the-art face inpainting methods. CodeFormer produces more natural face images and shows superior performance even under very large mask ratios (last row).

B.3 Old Photo Enhancement

We evaluate our method on an old historic photo of *the 5-th Solvay conference* that was taken in 1927. For inference, we first crop out and align all faces, then enhance faces using the proposed CodeFormer, and finally paste back the enhanced faces to the original photo. Fig. 3 shows both overall results and cropped face results.

Figure 3: Results of the old photo of *the 5-th Solvay conference* taken in 1927. (**Zoom in for best view**)

C More Results on Blind Face Restoration

In this section, we provide more visual comparisons with state-of-the-art methods, including DFDNet [5], PSFRGAN [2], GLEAN [1], GFP-GAN [6], and GPEN [7]. As shown in Fig. 4, the proposed CodeFormer not only produces high-quality faces but also preserves the identifies well, even when the input faces are highly degraded. Moreover, compared with other methods, the proposed CodeFormer is able to recover richer details and produce more natural faces.

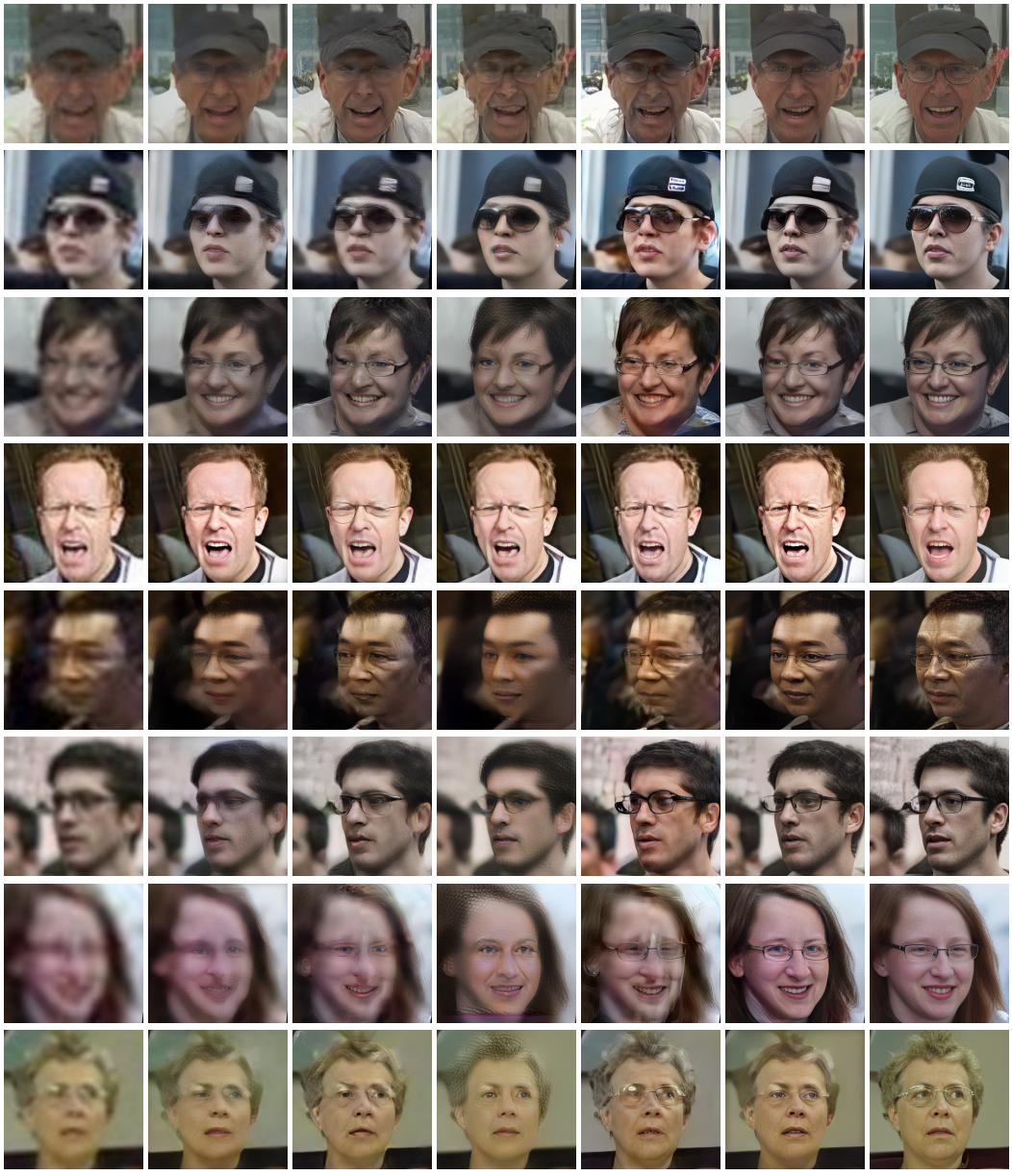

Figure 4: Qualitative comparison on real-world faces from the proposed Wider-Test dataset. Our CodeFormer is able to restore high-quality faces, showing robustness to the heavy degradation. (**Zoom in for best view**)

References

[1] Kelvin CK Chan, Xiangyu Xu, Xintao Wang, Jinwei Gu, and Chen Change Loy. GLEAN: Generative latent bank for image super-resolution and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[2] Chaofeng Chen, Xiaoming Li, Lingbo Yang, Xianhui Lin, Lei Zhang, and Kwan-Yee K Wong. Progressive semantic-aware style transformation for blind face restoration. In *CVPR*, 2021.

[3] Xiefan Guo, Hongyu Yang, and Di Huang. Image inpainting via conditional texture and structure dual generation. In *ICCV*, 2021.

[4] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

[5] Xiaoming Li, Chaofeng Chen, Shangchen Zhou, Xianhui Lin, Wangmeng Zuo, and Lei Zhang. Blind face restoration via deep multi-scale component dictionaries. In *ECCV*, 2020.

[6] Xintao Wang, Yu Li, Honglun Zhang, and Ying Shan. Towards real-world blind face restoration with generative facial prior. In *CVPR*, 2021.

[7] Tao Yang, Peiran Ren, Xuansong Xie, and Lei Zhang. GAN prior embedded network for blind face restoration in the wild. In *CVPR*, 2021.