WarpGAN: Automatic Caricature Generation (Supplementary Material)

Yichun Shi* Debayan Deb* Anil K. Jain
Michigan State University, East Lansing MI 48824
{shiyichu, debdebay}@msu.edu, jain@cse.msu.edu

In this supplementary material, we include more implementation details and compare with more baselines (previous works on caricature generation). We also compare different transformation methods, show more results on ablation study and texture styles and show caricature generation results on a selfie dataset.

1. Implementation Details

Preprocessing We align all the images with five landmarks (left eye, right eye, nose, mouth left, mouth right) using the ones provided in the WebCaricature dataset [1] protocol. Since the protocol does not provide the locations of eye centers, we estimate them by taking the average of the corresponding eye corners. Then, a similarity transformation is applied for all the images using the five landmarks. The aligned images are resized to 256×256. The whole dataset consists of $6,042$ caricatures and $5,974$ photos from 252 identities. We randomly split the dataset into a training set of 126 identities ($3,016$ photos and $3,112$ caricatures) and a testing set of 126 identities ($2,958$ photos and $2,930$ caricatures). **All the testing images in the main paper and this supplementary material are from the identities in the testing split**.

Experiment Settings We conduct all experiments using Tensorflow r1.9 and one Geforce GTX 1080 Ti GPU. The average speed for generating one caricature image on this GPU is 0.082s.

Architecture Our network architecture is modified based on MUNIT [2]. Let `c7s1-k` be a 7×7 convolutional layer with k filters and stride 1. `dk` denotes a 4×4 convolutional layer with k filters and stride 2. `Rk` denotes a residual block that contains two 3×3 convolutional layers. `uk` denotes a $2\times$ upsampling layer followed by a 5×5 convolutional layer with k filters and stride 1. `fck` denotes a fully connected layer with k filters. `avgpool` denotes a global average pooling layer. We apply Instance Normalization (IN) [3] to the content encoder and Adaptive Instance Normalization (AdaIN) [4] to the decoder. No normalization is used in the style encoder. We use Leaky ReLU with slope 0.2 in the discriminator and ReLU activation everywhere else. The architectures of different modules are as follows:

- Style Encoder:
  `c7s1-64,d128,d256,avgpool,fc8`
- Content Encoder:
  `c7s1-64,d128,d256,R256,R256,R256`
- Decoder:
  `R256,R256,R256,u128,u64,c7s1-3`
- Discriminator:
  `d32,d64,d128,d256,d512,fc512,fc3M`

A separate branch of 1×1 convolutional layer with 3 filters and stride 1 is attached to the last convolutional layer of the discriminator to output D_1, D_2, D_3 for patch adversarial losses. The style decoder (the multi-layer perceptron) has two hidden fully connected layers of 128 filters without normalization and the warp controller has only one hidden fully connected layer of 128 filters with Layer Normalization [5]. The length of the latent style code is set to 8.

2. Additional Baselines

In the main paper, we compared WarpGAN with state-of-the-art style transfer networks as baselines. Here, we compare WarpGAN with other caricature generation works [6, 7, 8, 9, 10, 11]. Since these methods do not release their code and use different testing images, we crop the images from their papers and compare with them one by one. All the baseline results are also taken from their original papers. The results are shown in Figure 1.

3. Transformation Methods

To see the advantage of the proposed control-points estimation for automatic warping, we train three variants of our model by replacing the warping method with (1)

* indicates equal contribution

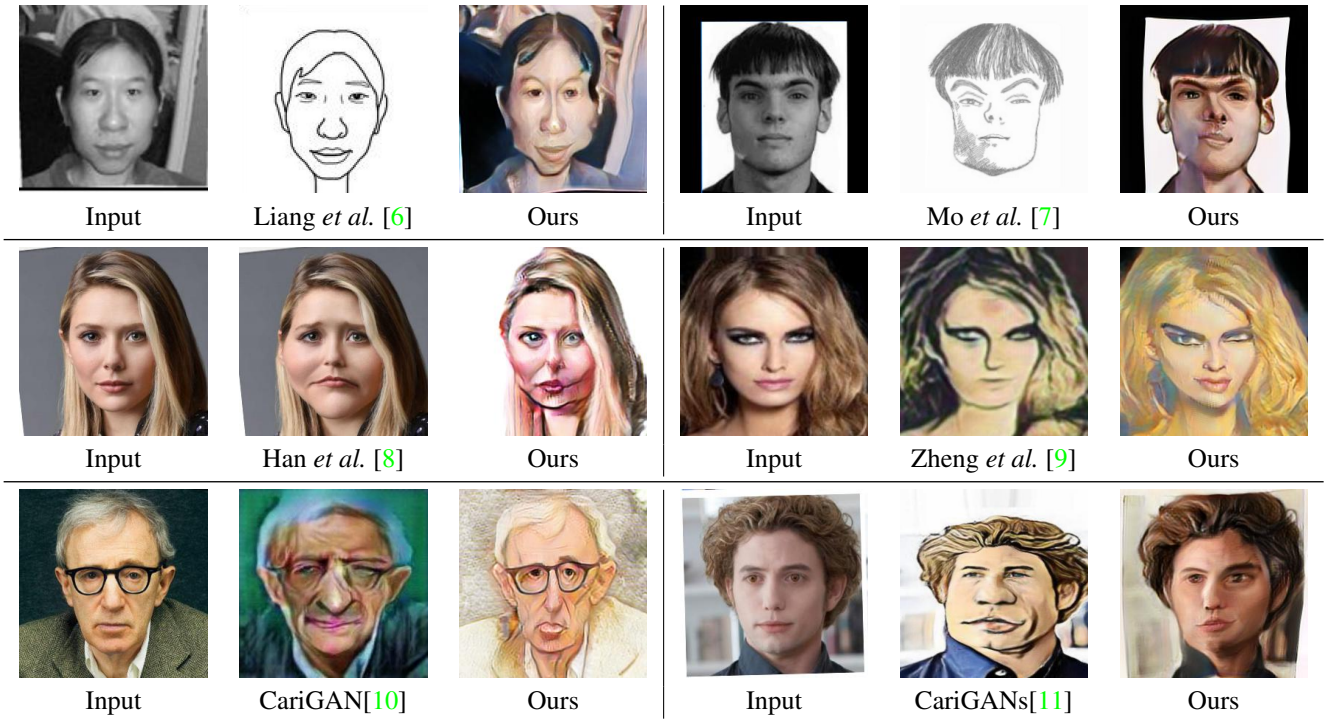

Figure 1: Comparison with previous works on caricature generation. In each cell, the left and middle images are the input and result images taken from the baseline paper, respectively. The right images are the results of WarpGAN.

projective transformation, (2) dense deformation and (3) landmark-based warping. In projective transformation, the warp controller outputs 8 parameters for the transformation matrix. In dense deformation, the warp controller outputs a 16×16 deformation grid, which is further interpolated into 256×256 for grid sampling. In landmark-based warping, we use the landmarks provided by Dlib[1] and the warp controller only outputs the displacements. As shown in Figure 3, the warping is too limited in projective transformation for generating artistic caricatures and too unconstrained in dense deformation that it is difficult to train. Landmark-based warping yields reasonable results, but it is limited by the landmark detector. In comparison, our methods does not require any domain knowledge, has little limitation and leads to visually satisfying warping results.

4. More Results

Ablation Study We show more results of the ablation study in Figure 2. The results are consistent with those in the main paper: (1) the joint learning of texture rendering and warping are crucial for generating realistic caricature images and (2) without patch adversarial loss or identity-preservation adversarial loss, the model cannot learn to generate caricatures with various texture styles and shape exaggeration styles.

Different Texture Styles More results of texture style controlling are shown in Figure 4. Five latent style codes are randomly sampled from the normal distribution $\mathcal{N}(0, \mathbf{I})$. Images in the same column in Figure 4 are generated with the same style code.

Selfie Dataset To test the performance of our model in more application scenarios, we download the public Selfie dataset[2] [12] for cross-dataset evaluation. The dataset includes $46,836$ public selfies crawled from Internet. Unlike our training dataset (WebCaricature), the identities in this dataset are not restricted to celebrities and there is a difference between the visual styles of these images and the ones in our training dataset. The results are shown in Figure 5.

[1] http://dlib.net/face_landmark_detection.py.html

[2] http://crcv.ucf.edu/data/Selfie/

Figure 2: More results on ablation study. Input images are shown in the first column. The subsequent columns show the results of different models trained without a certain module or loss. The texture style codes are randomly sampled from the normal distribution.

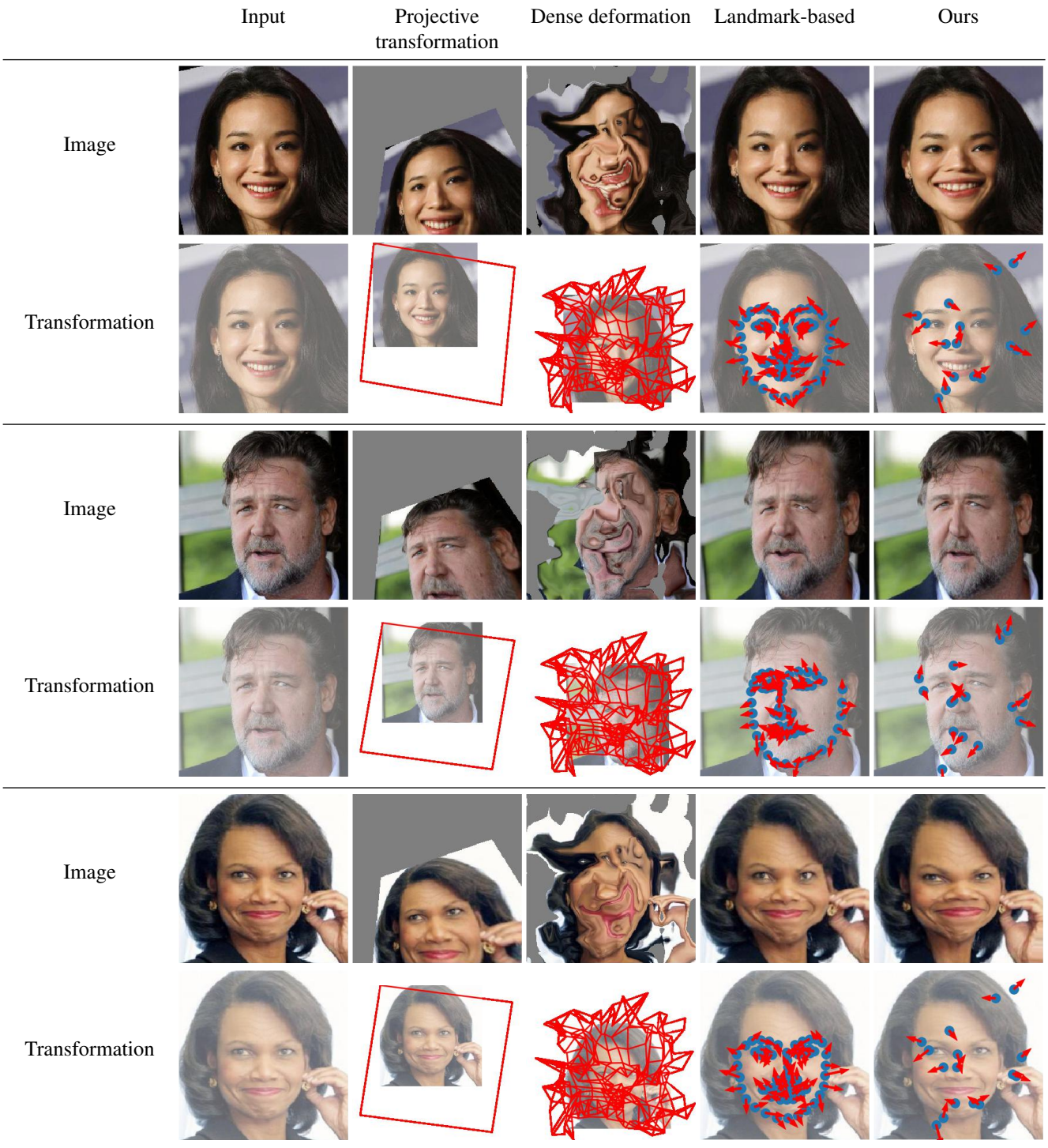

Figure 3: Different transformation methods. Input images are shown in the first column. The next four columns show the results and the transformation visualizations of four different models trained with different transformation methods. The landmark-based model uses 68 landmarks detected by Dlib. Texture rendering is hidden here for clarity.

Figure 4: Results of five different texture styles. Input images are shown in the first column. Subsequent five columns show the results of WarpGAN using five style codes sampled randomly from the normal distribution. All the images in the same column are generated with the same latent style code.

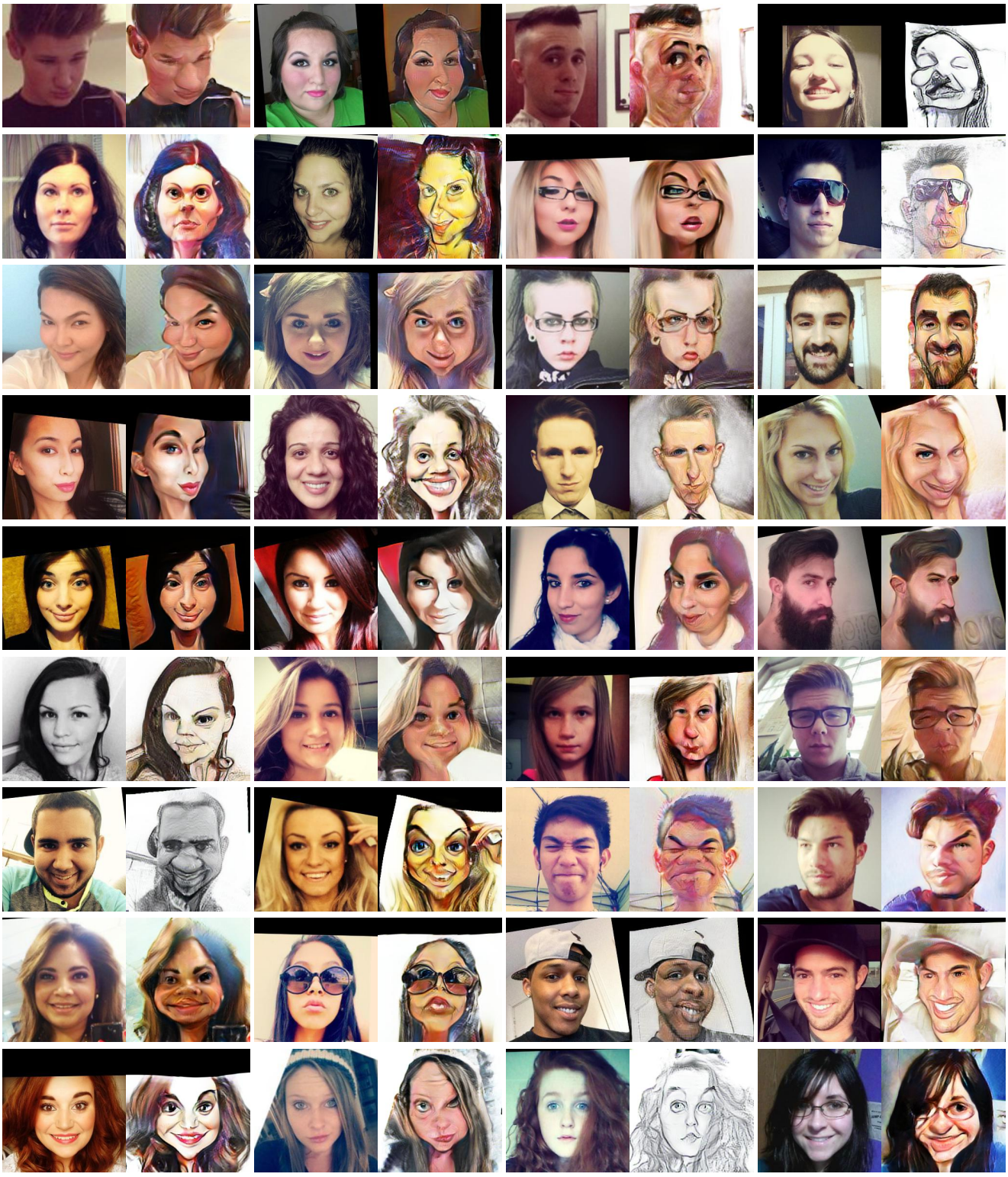

Figure 5: Example results on the Selfie dataset. This is a cross-dataset evaluation and no training is involved. In each pair, the left image is the input and the right image is the output of WarpGAN with a random texture style.

References

[1] Jing Huo, Wenbin Li, Yinghuan Shi, Yang Gao, and Hujun Yin. Webcaricature: a benchmark for caricature face recognition. *arXiv:1703.03230*, 2017. 1

[2] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. *arXiv:1804.04732*, 2018. 1

[3] Dmitry Ulyanov, Andrea Vedaldi, and Victor S Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *CVPR*, 2017. 1

[4] Xun Huang and Serge J Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017. 1

[5] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 1

[6] Lin Liang, Hong Chen, Ying-Qing Xu, and Heung-Yeung Shum. Example-based caricature generation with exaggeration. In *Pacific Conf. on Computer Graphics and Applications*, 2002. 1, 2

[7] Zhenyao Mo, John P Lewis, and Ulrich Neumann. Improved automatic caricature by feature normalization and exaggeration. In *SIGGRAPH*, 2004. 1, 2

[8] Xiaoguang Han, Kangcheng Hou, Dong Du, Yuda Qiu, Yizhou Yu, Kun Zhou, and Shuguang Cui. Caricatureshop: Personalized and photorealistic caricature sketching. *arXiv:1807.09064*, 2018. 1, 2

[9] Ziqiang Zheng, Haiyong Zheng, Zhibin Yu, Zhaorui Gu, and Bing Zheng. Photo-to-caricature translation on faces in the wild. *arXiv:1711.10735*, 2017. 1, 2

[10] Wenbin Li, Wei Xiong, Haofu Liao, Jing Huo, Yang Gao, and Jiebo Luo. CariGAN: Caricature Generation through Weakly Paired Adversarial Learning. *arXiv:1811.00445*, 2018. 1, 2

[11] Kaidi Cao, Jing Liao, and Lu Yuan. CariGANs: Unpaired Photo-to-Caricature Translation. *arXiv:1811.00222*, 2018. 1, 2

[12] Mahdi M Kalayeh, Misrak Seifu, Wesna LaLanne, and Mubarak Shah. How to take a good selfie? In *ACM MM*, 2015. 2