Supplementary Material: MineGAN: effective knowledge transfer from GANs to target domains with few images

A. Architecture and training details

MNIST dataset. Our model contains a *miner*, a *generator* and a *discriminator*. For both unconditional and conditional GANs, we use the same framework [2] to design the generator and discriminator. The miner is composed of two fully connected layers with the same dimensionality as the latent space $|z|$. The visual results are computed with $|z| = 16$; we found that the quantitative results improved for larger $|z|$ and choose $|z| = 128$. We randomly initialize the weights of each miner following a Gaussian distribution centered at 0 with 0.01 standard deviation, and optimize the model using Adam [4] with batch size of 64. The learning rate of our model is 0.0004, with exponential decay rates of $(\beta_1, \beta_2) = (0.5, 0.999)$.

In the conditional MNIST case, label c is a one-hot vector. This differs from the conditioning used for BigGAN [1] explained in Section 3.3. Here we extend MineGAN to this type of conditional models by considering each possible conditioning as an independently pretrained generator and using the selector to predict the conditioning label. Given a conditional generator $G(c, z)$, we consider $G(i, z)$ as G_i and apply the presented MineGAN approach for multiple pretrained generators on the family $\{G(i, z)|\, i = 1, ..., N\}$. The resulting selector now chooses among the N classes of the model rather than among N pretrained models, but the rest of the MineGAN training remains the same, including the training of N independent miners.

CelebA Women, FFHQ Children and LSUN (Tower and Bedroom) datasets. We design the generator and discriminator based on Progressive GANs [3]. Both networks use a multi-scale technique to generate high-resolution images. Note we use a *simple* miner for tasks with dense and narrow source domains. The miner comprises out of four fully connected layers (8-64-128-256-512), each of which is followed by a *ReLU* activation and *pixel normalization* [3] except for last layer. We use a Gaussian distribution centered at 0 with 0.01 standard deviation to initialize the miner, and optimize the model using Adam [4] with batch size of 4. The learning rate of our model is 0.0015, with exponential decay rates of $(\beta_1, \beta_2) = (0, 0.99)$.

FFHQ Face and Anime Face datasets. We use the same network as [5], namely SNGAN. The miner consists of three fully connected layers (8-32-64-128). We randomly initialize the weights following a Gaussian distribution centered at 0 with 0.01 standard deviation. For this additional set of experiments, we use Adam [4] with a batch size of 8, following a hyper parameter learning rate of 0.0002 and exponential decay rate of $(\beta_1, \beta_2) = (0, 0.9)$.

Imagenet and Places365 datasets. We use the pretrained BigGAN [1]. We ignore the projection loss in the discriminator, since we do not have access to the label of the target data. We employ a more *powerful* miner in order to allocate more capacity to discover the regions related to target domain. The miner consists of two sub-networks: miner M^z and miner M^c. Both M^z and M^c are composed of four fully connected layers of sizes (128, 128)-(128, 128)-(128, 128)-(128, 128)-(128, 120) and (128, 128)-(128, 128)-(128, 128)-(128, 128)-(128, 128), respectively. We use Adam [4] with a batch size of 256, and learning rates of 0.0001 for the miner and the generator and 0.0004 for the discriminator. The exponential decay rates are $(\beta_1, \beta_2) = (0, 0.999)$. We randomly initialize the weights following a Gaussian distribution centered at 0 with 0.01 standard deviation.

The input of BigGAN [1] is a random latent vector and a class label that is mapped to an embedding space. We therefore have two miner networks, the original one that maps to the input latent space (M^z) and a new one that maps to the latent class embedding (M^c). Note that since we have no class label, the miner (M^c) needs to learn what distribution over the embeddings best represents the target data.

B. Evaluation metric details

Similarly to [6], we compute FID between 10,000 randomly generated images and 10,000 real images, if possible. When the number of target image exceeds 10,000, we randomly select a subset containing only 10,000. On the other hand, if the target set contains fewer images, we cap the amount of randomly generated images to this number to compute the FID. Note we also consider KMMD to evaluate the distance between generated images and real images since FID suffers from instability on small datasets.

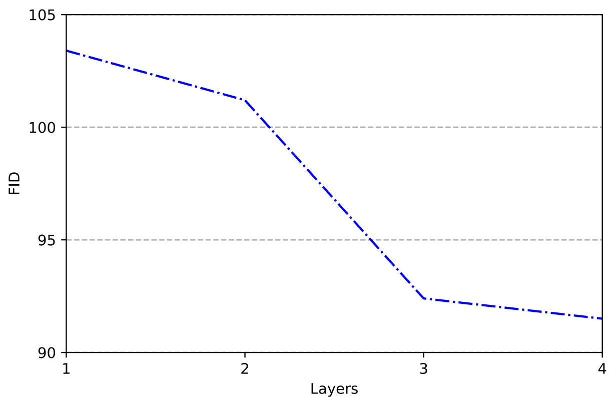

Figure 9: FID values for different number of fully connected layers in the miner. Results are based on the pretrained *BigGAN* with target class *Arch*.

Method	MineGAN (mean)	MineGAN (max)
Car	0.51	0.34
Bus	0.49	0.66

Table 4: Estimated probabilities p_i for {Car, Bus} $\rightarrow$ Red vehicles for MineGAN with mean or $\max$ in Eqs. (5) and (6). The actual data distribution is 0.3:0.7 (ratio cars:buses).

C. Ablation study

In this section, we evaluate the effect of each independent contribution to MineGAN and their combinations.

Selection strategies. We ablate the use of the $\max$ operation in Eqs. (5) and (6), and replace it with a mean operation. We call this setting MineGAN (mean). In this case the backpropagation is not only performed for the image with the highest critic score but for all images. We hypothesized that the $\max$ operation is necessary to correctly estimate the probabilities used by the selector. We report the results in Table 4, where we refer to our original model as MineGAN ($\max$). The probability distribution predicted by the selector indicates that MineGAN (mean) equally chooses both pretrained models on Car and Bus, while MineGAN successfully estimates the class distribution of the target data.

Miner Architecture. We performed an ablation study for different variants of the miner based on pretrained BigGAN. The miner always contains only fully connected layers, but we experiment with varying the number of layers. In Fig. 9, we show the results on off-manifold target class *Arch* from Places365 [8]. Using more layers increases the performance of the method. Besides, we find that the results of both 3 and 4 layers are similar, indicating that adding additional layers would only result in slight improvements for our model. Therefore, in this paper, we use miners with 4 fully connected layers.

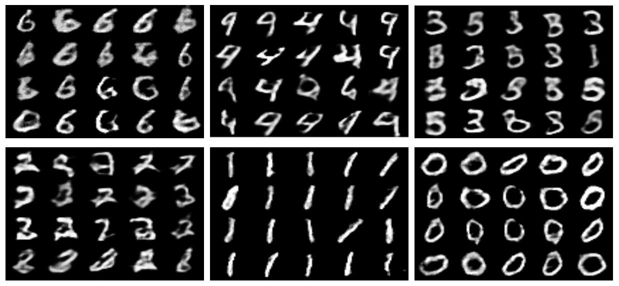

Figure 10: Results for unconditional off-manifold generation of digits ‘6’, ‘4’, ‘3’, ‘2’, ‘1’, ‘0’.

D. MNIST experiment

We expand the MNIST experiments presented in Section 5.1 by providing a quantitative evaluation and including results on conditional GANs. As evaluation measures, we use FID (Section 5) and classifier error [7]. To compute classifier error, we first train a CNN classifier on real training data to distinguish between multiple classes (e.g. digit classifier). Then, we classify the generated images that should belong to a particular class and measure the error as the percentage of misclassified images. This gives us an estimation of how realistic and accurate the generated images are in the context of targeted generation.

Table 5 presents the results for both unconditional and conditional models, using a noise length of $|z| = 128$. The relatively low error values indicate that the miner manages to identify the correct regions for generating the target digits. The conditional model offers better results than the unconditional one by selecting the target class more often. We can also observe that the off-manifold task is more difficult than the on-manifold task, as indicated by the higher evaluation scores. However, the off-manifold scores are still reasonably low, indicating that the miner manages to find suitable regions from other digits by mining local patterns shared with the target. Overall, these results indicate the effectiveness of mining on MNIST for both types of targeted image generation. In addition, in Fig. 10 we have added a visualization for the off-manifold MNIST classes which were not already shown in Fig. 2.

E. Further results on CelebA

We provide additional results for the on-manifold experiment CelebA$\rightarrow$FFHQ women in Fig. 11, and the off-manifold CelebA$\rightarrow$FFHQ children in Fig. 12. In addition, we have also performed an on-manifold experiment with CelebA$\rightarrow$CelebA women, whose results are provided in Fig. 13.

d	On-manifold		Off-manifold	
	Unconditional	Conditional	Unconditional	Conditional
0	13.4 / 2.5	12.6 / 0.7	21.3 / 2.8	15.6 / 1.1
1	13.1 / 1.7	12.6 / 1.9	15.9 / 2.5	14.8 / 2.1
2	14.6 / 6.3	12.8 / 2.7	23.1 / 5.2	18.2 / 3.6
3	14.1 / 10.1	13.3 / 1.6	22.8 / 7.3	14.2 / 1.5
4	14.7 / 6.4	13.4 / 1.2	23.4 / 6.3	15.3 / 4.2
5	13.1 / 9.3	11.7 / 2.1	21.9 / 10.9	17.2 / 5.7
6	13.4 / 2.8	14.3 / 1.8	24 / 3.1	15.8 / 1.6
7	12.9 / 3.2	14.2 / 1.8	24.8 / 4.9	16.3 / 2.6
8	14.2 / 7.5	14.7 / 5.5	25.7 / 9.8	18.7 / 5.6
9	11.3 / 6.8	11.2 / 2.9	12.5 / 7.4	16.3 / 3.5
Average	13.5 / 5.7	13.1 / 2.2	21.5 / 6.0	16.2 / 3.2

Table 5: Quantitative results of mining on MNIST, expressed as FID / classifier error.

F. Further results for LSUN

We provide additional results for the experiment ($\{$Bus, Car$\}$) $\rightarrow$ Red vehicles in Fig. 16 and for the experiment $\{$Bedroom, Bridge, Church, Kitchen$\}$ $\rightarrow$ Tower/Bedroom in Fig. 17. When applying MineGAN to multiple pretrained GANs, we use one of the domains to initialize the weights of the critic. In Fig. 17 we used *Church* to initialize the critic in case of the target set *Tower*, and *Kitchen* to initialize the critic for the target set *Bedroom*. We found this choice to be of little influence on the final results. When using *Kitchen* to initialize the critic for target set *Tower* results change from 62.4 to 61.7. When using *Church* to initialize the critic for target set *Bedroom* results change from 54.7 to 54.3.

References

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019.

[2] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *NeurIPS*, pages 5767–5777, 2017.

[3] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *ICLR*, 2017.

[4] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2014.

[5] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.

[6] Atsuhiro Noguchi and Tatsuya Harada. Image generation from small datasets via batch statistics adaptation. *ICCV*, 2019.

[7] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. How good is my gan? In *ECCV*, pages 213–229, 2018.

[8] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *NeurIPS*, pages 487–495, 2014.

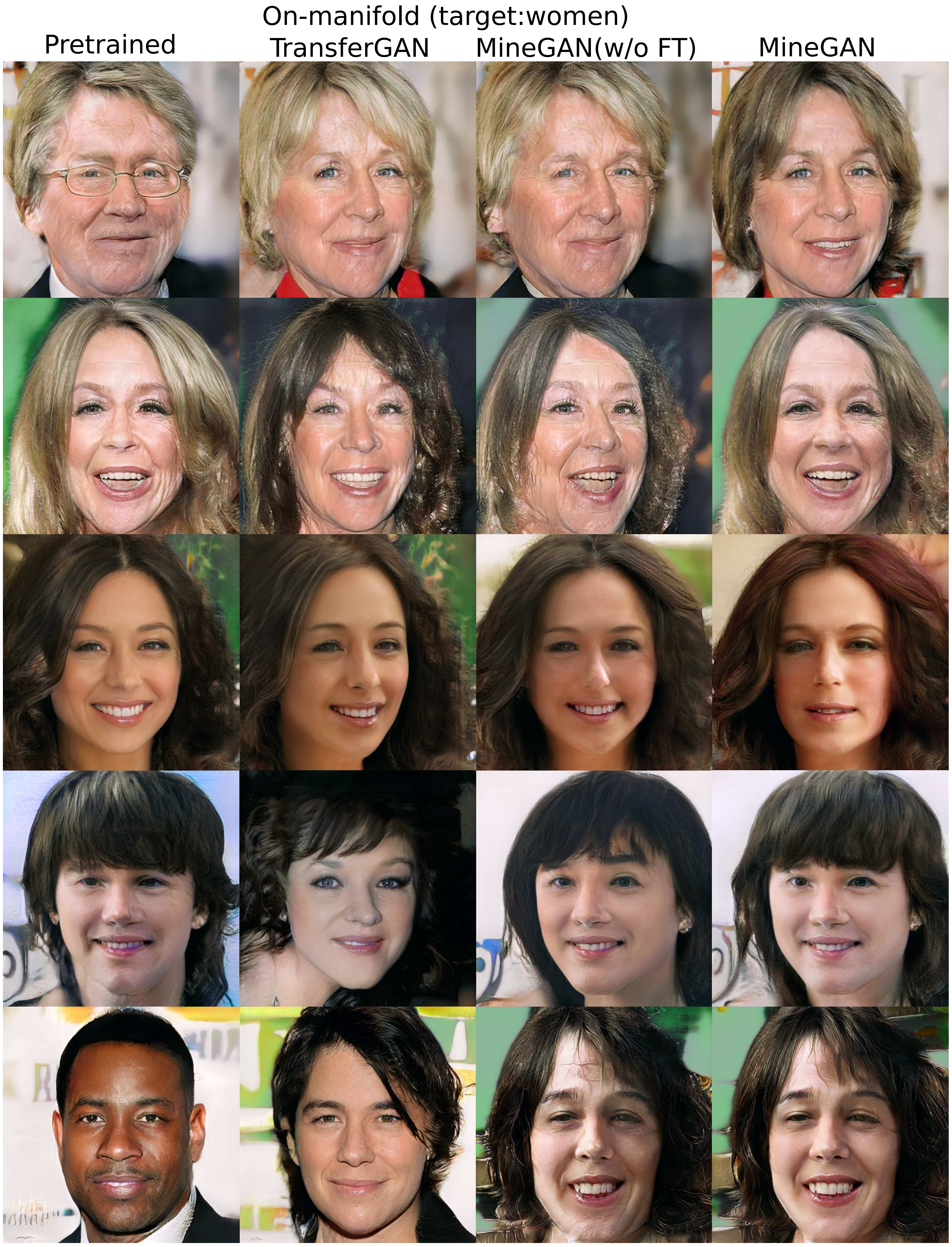

Figure 11: (CelebA→FFHQ women). Based on pretrained *Progressive GAN*.

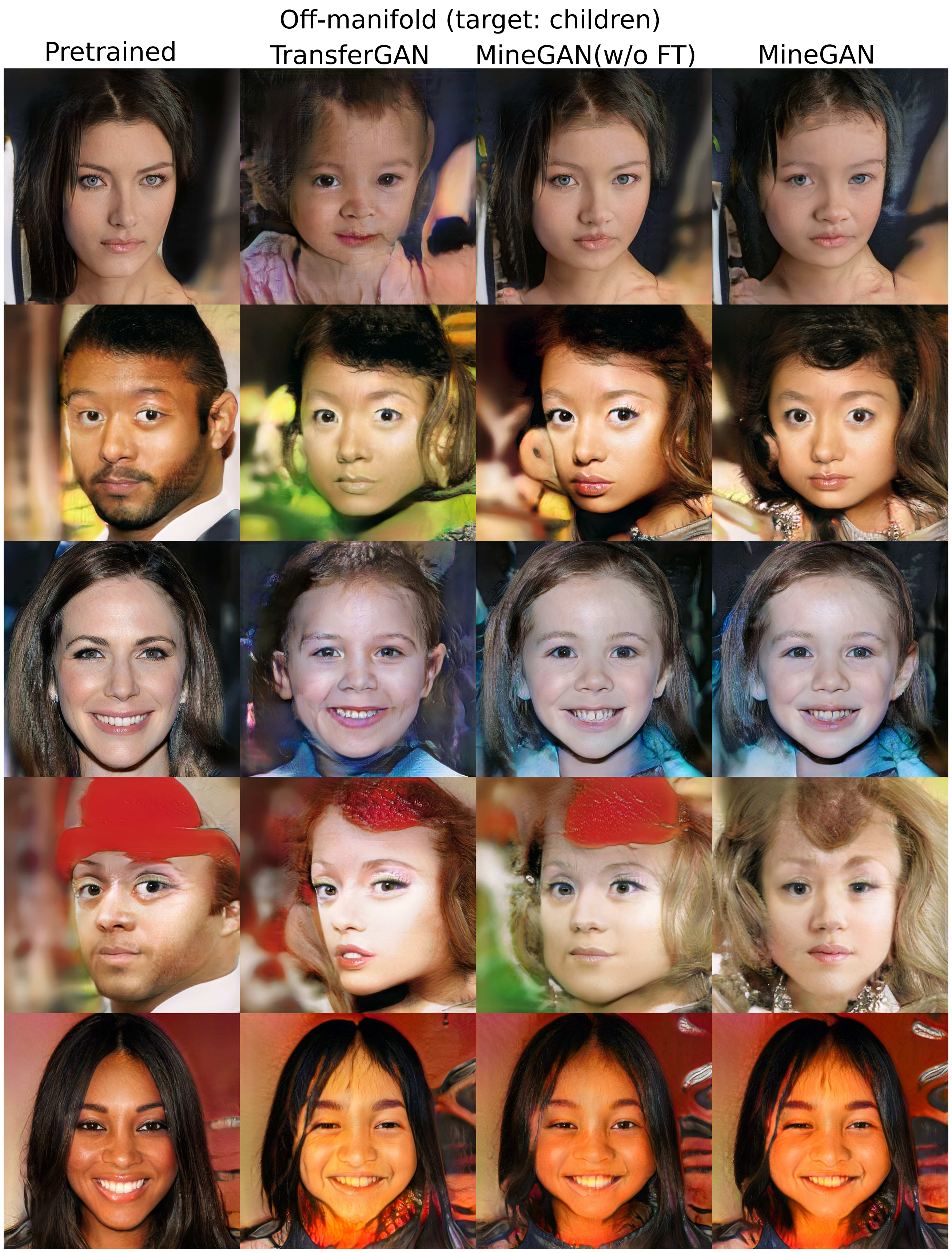

Figure 12: (CelebA→ FFHQ children). Based on pretrained *Progressive GAN*.

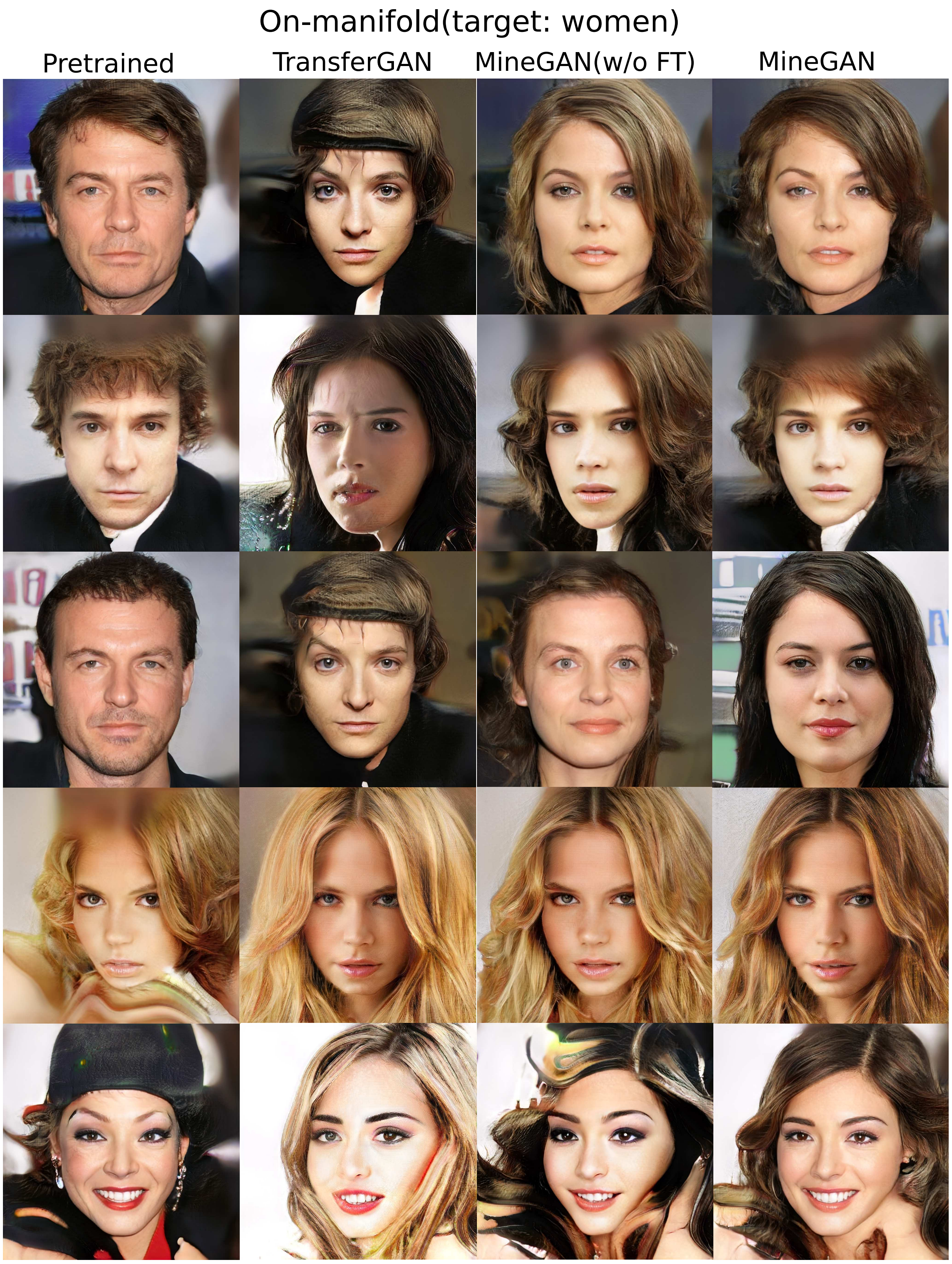

Figure 13: (CelebA→CelebA women). Based on pretrained *Progressive GAN*.

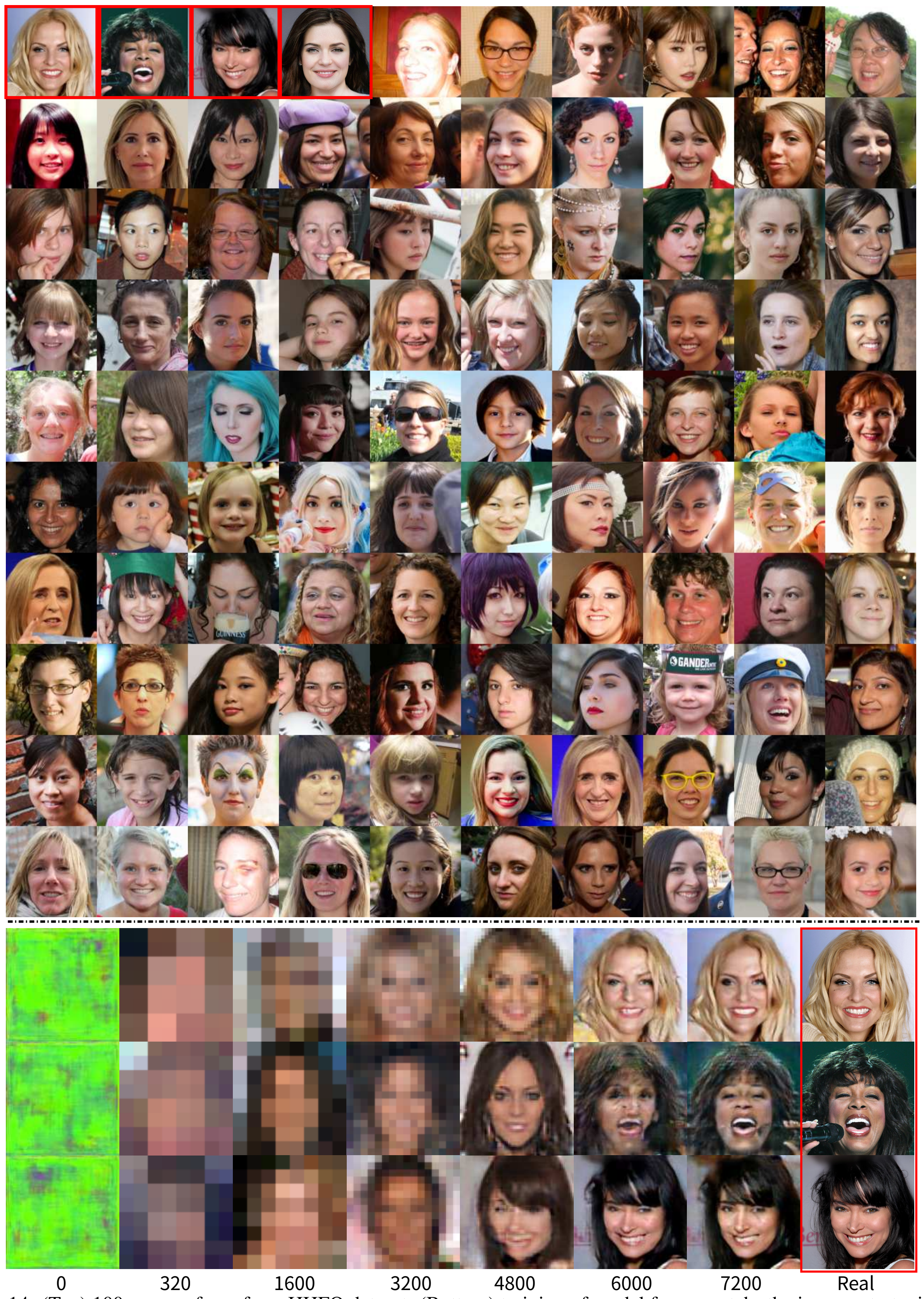

Figure 14: (Top) 100 women faces from HHFQ dataset. (Bottom) training of model from scratch: the images start with low quality and iteratively overfit to a particular training image. Red boxes identify images which are remembered by the model trained from scratch or from TransferGAN (see Fig. 4). Based on pretrained *Progressive GAN*.

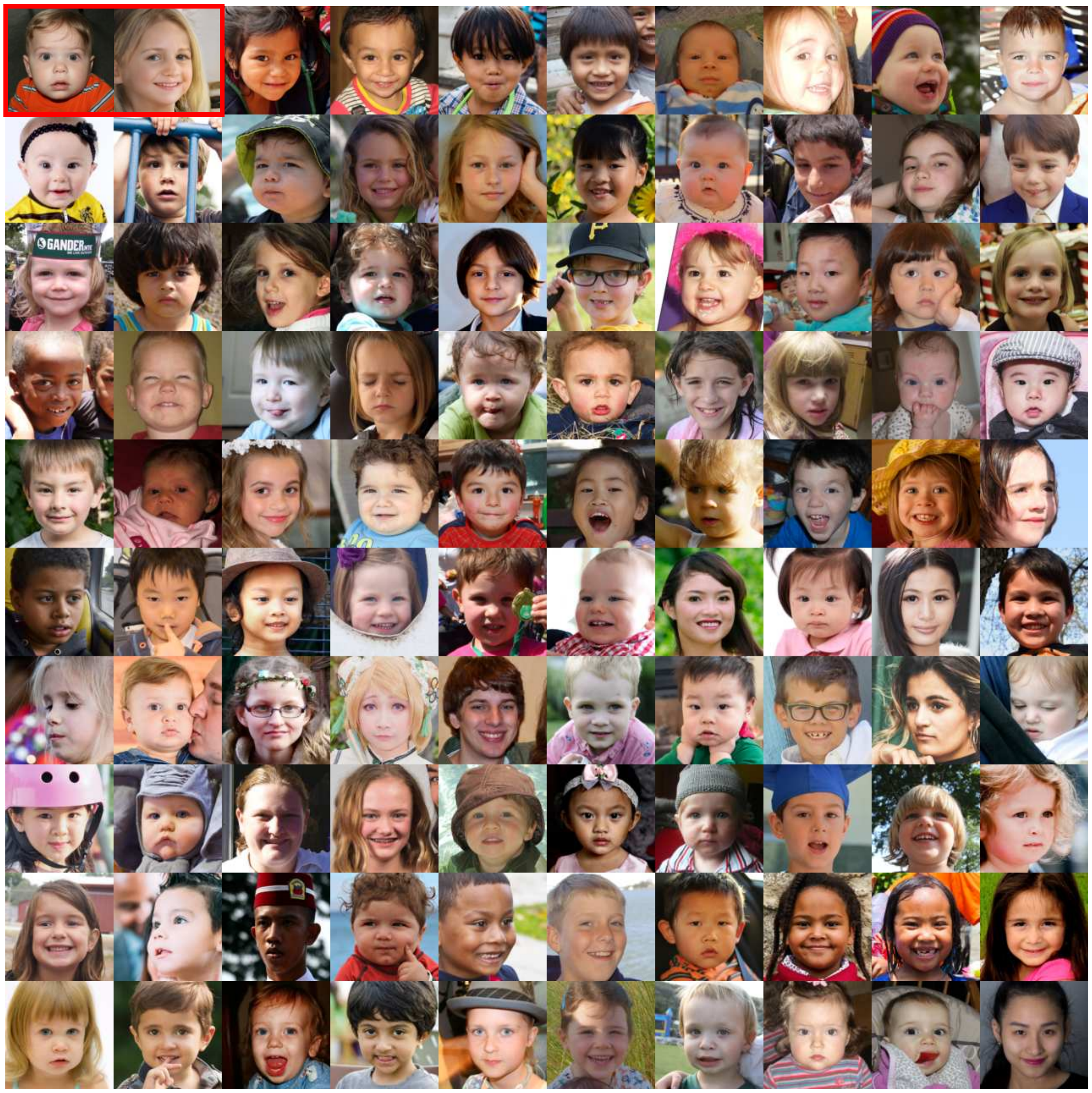

Figure 15: 100 children faces from HHFQ dataset. Red boxes identify images which are remembered by the model trained from scratch (see Fig. 4). Based on pretrained *Progressive GAN*.

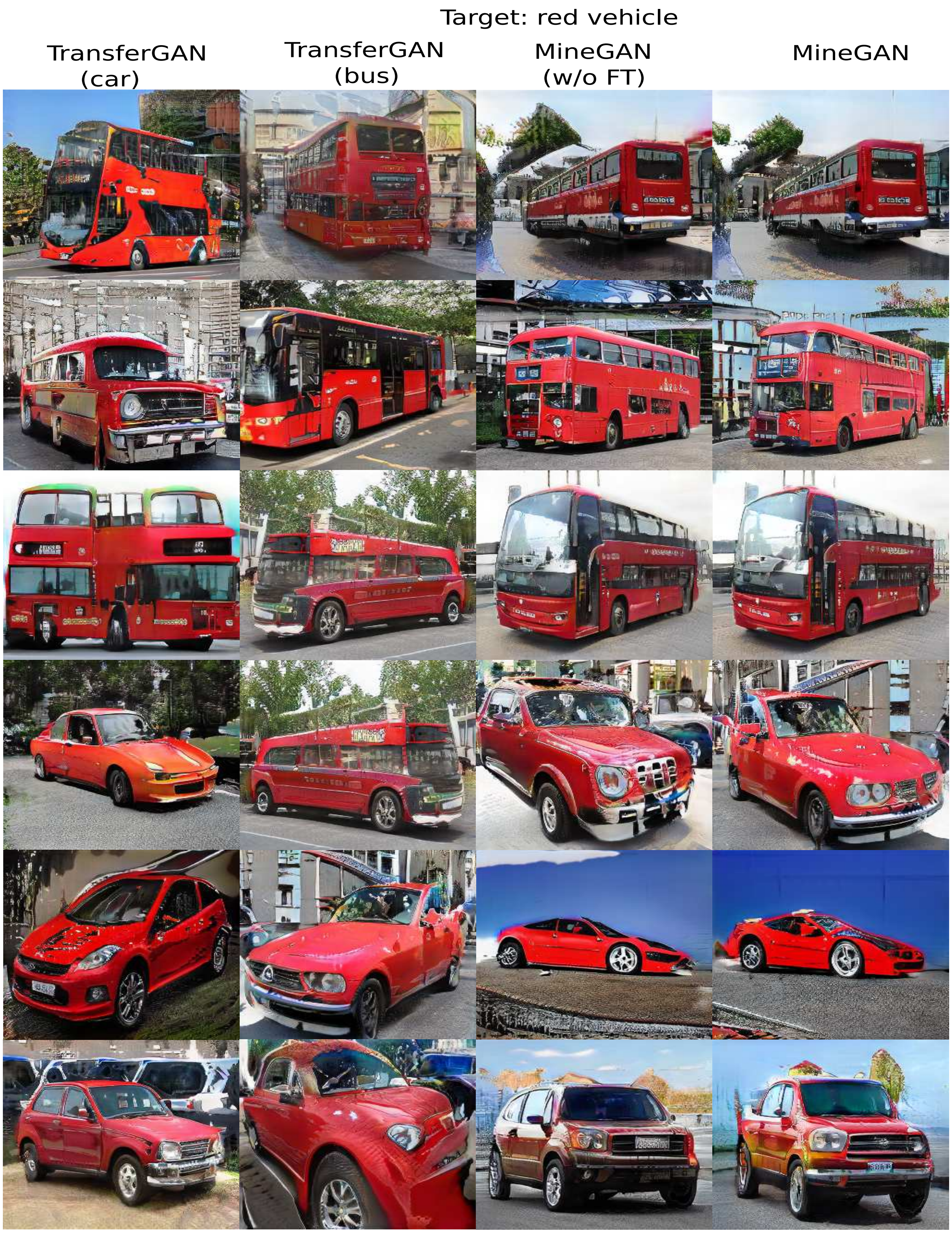

Figure 16: ({bus, car}) →red vehicles. Based on pretrained *Progressive GAN*.

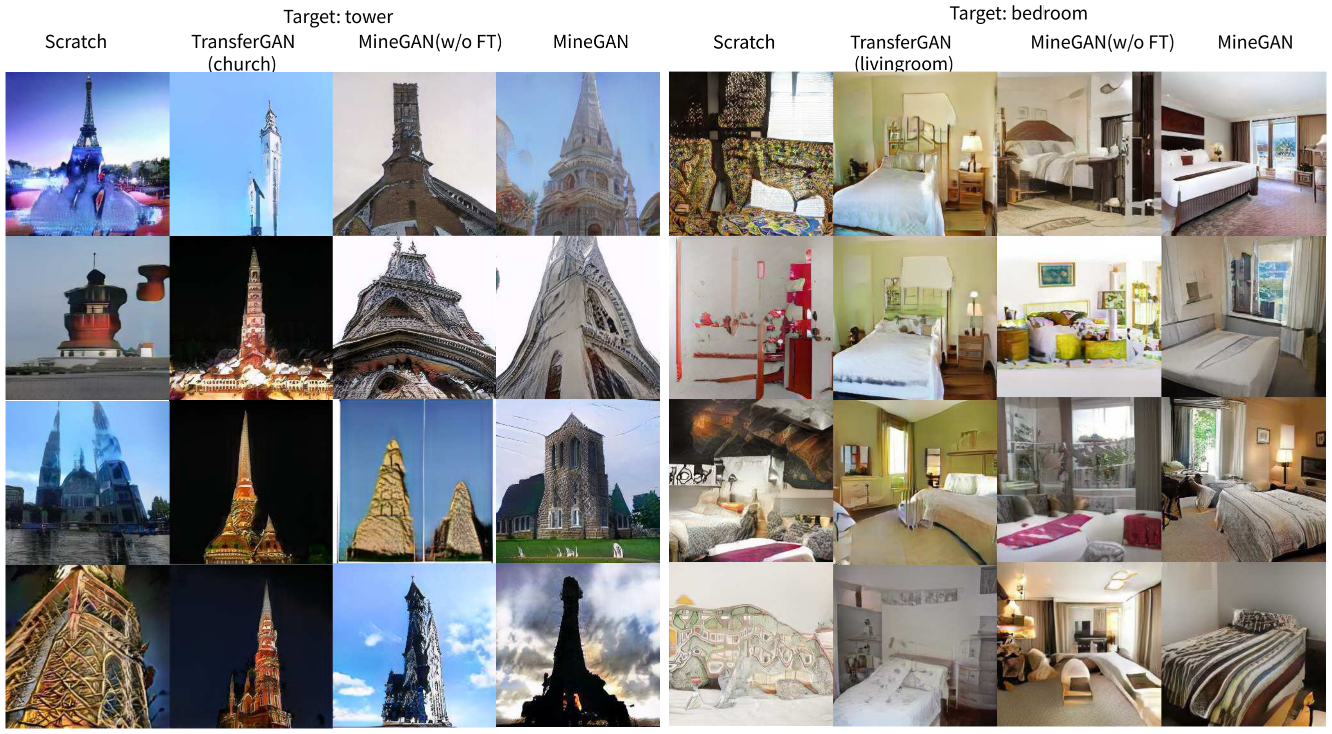

Figure 17: Results for unconditional GAN. (Top) (Livingroom, kitchen, bridge, church)→Tower. (Bottom) (Livingroom, kitchen, bridge, church)→Bedroom. Based on pretrained *Progressive GAN*.