Two-Stage Peer-Regularized Feature Recombination for Arbitrary Image Style Transfer Supplementary material

Jan Svoboda[1,2], Asha Anoosheh[1], Christian Osendorfer[1], Jonathan Masci[1]
[1]NNAISENSE, Switzerland [2]Universita della Svizzera italiana, Switzerland
{jan.svoboda,asha.anoosheh,christian.osendorfer,jonathan.masci}@nnaisense.com

1. Network architecture

This section describes our model in detail. We describe the encoder-decoder network and discriminator in separate sections below. We provide our code containing the implementation details [1] to assure full reproducibility of all the presented results.

1.1. Autoencoder

Detailed scheme of the architecture is depicted in Figure 2. Each of the convolutional layers (in yellow) is followed by Instance Normalization (IN) [6] and ReLU nonlinearity [3]. The TPFR module uses a variant of the Peer Regularization Layer [5] with Euclidean distance metric, k-NN with $K = 5$ nearest neighbours and dropout on the attention weights of 0.2.

The generated latent code are 768 feature maps of size $(W/4) \times (H/4)$, where W and H are the input width and height respectively. First 256 feature maps is the content latent representation, while the remaining 512 is for the style. The style latent representation is further split into halves, having first 256 feature maps left unchanged and the second 256 feature maps are passed through the *Global style transform* block producing feature maps of size 1×1 that hold the global part of the style latent representation.

The last convolutional block of the decoder is equipped with TanH nonlinearity and produces the reconstructed RGB image.

The auxiliary decoder copies the architecture of the main decoder, while omitting the *Style transfer* block (see Figure 2).

1.2. Discriminator

The discriminator architecture is shown in Figure 1. It takes two RGB images concatenated over the channel dimension as input and produces a $(W/4) \times (H/4)$ map of predictions. Our implementation uses LS-GAN and therefore there is no Sigmoid activation at the output

To stabilize the discriminator training, we add random Gaussian noise to each input:

$$X = X + N(\mu, \sigma), \tag{1}$$

where N is a Gaussian distribution with mean $\mu = 0$ and standard deviation $\sigma = 0.1$.

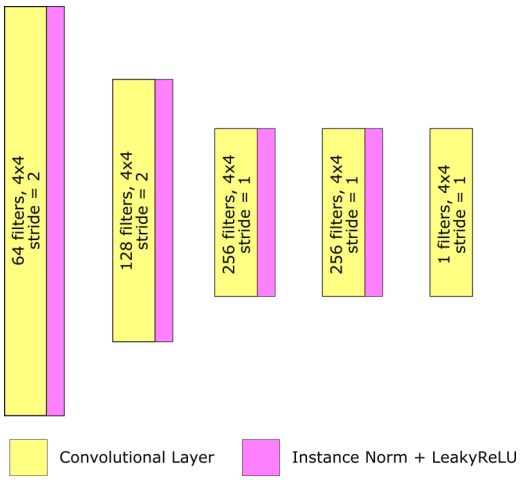

Figure 1. Detailed architecture of discriminator.

2. Style transfer results

This section provides more qualitative results of our style transfer approach that did not fit in the main text. Figure 3 are images generated with resolution 512×512 and shows the generalization of our approach to different styles and ability of our approach to perform zero-shot style transfer. In particular, we have collected some paintings from Salvador Dali, Camille Pissarro, Henri Matisse, Katigawa Utamaro and Rembrandt.

In addition, images in Figure 4 were generated with resolution 256×256 and show results of transfer taking a ran-

[1]Code is available at this link: http://nnaisense.com/conditional-style-transfer

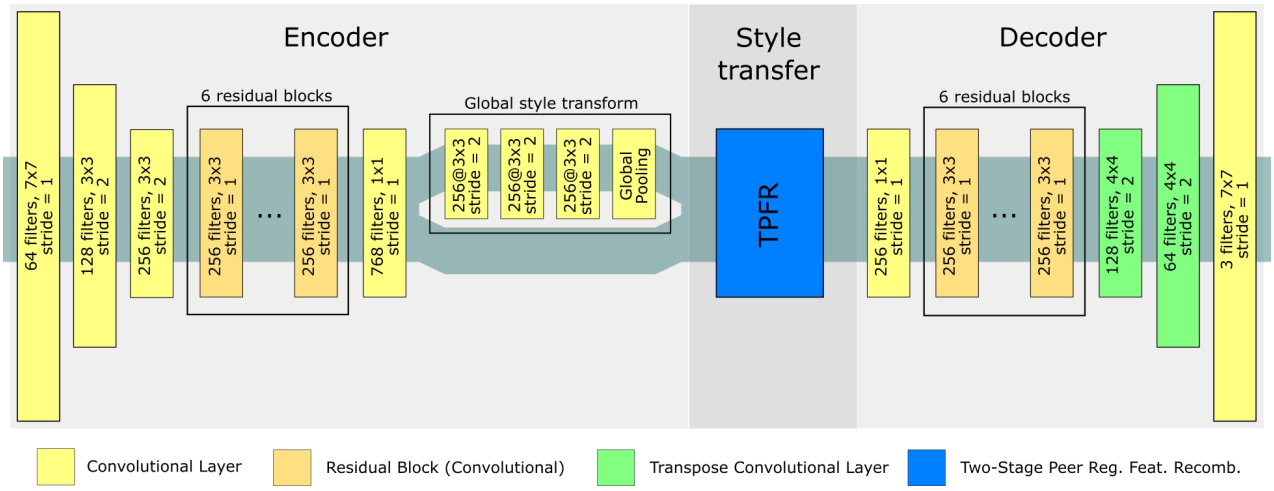

Figure 2. Detailed architecture of autoencoder. The faded blue line in the background shows the flow of information through the model.

dom painting from the dosjoint test set of thirteen painting styles that our model was trained with (Morisot, Munch, El Greco, Kirchner, Pollock, Monet, Roerich, Picasso, Cezzane, Gaugin, Van Gogh, Peploe and Kandinsky).

3. Latent space structure

Our latent representation is split into two parts, $(z)_C$ and $(z)_S$, content and style respectively. Metric learning loss is used on the style part in order to enforce a separation of different modalities in the style latent space.

$$\begin{aligned} L^{pos}_{z_{style}} &= f[(z_{i_1})_S - (z_{i_2})_S] + f[(z_{t_1})_S - (z_{t_2})_S] \\ L^{neg}_{z_{style}} &= f[(z_{i_1})_S - (z_{t_1})_S] + f[(z_{i_2})_S - (z_{t_2})_S] \\ L_{z_{style}} &= L^{pos}_{z_{style}} + max(0.0, \mu - L^{neg}_{z_{style}}). \end{aligned} \quad (2)$$

where $(z_{i1})_S$, $(z_{i2})_S$ are style parts of latent representations of two different input images and $(z_{t1})_S$, $(z_{t2})_S$ are style parts of latent representations of two different targets from the same target class. Parameter $\mu = 1$ and it is the margin we are enforcing on the separation of the positive and negative scores.

3.1. Visualization in image space

Figure 5 visualizes the influence of the $(z)_C$ and $(z)_S$ parts of the latent representation after decoding back into the RGB image space. The TPFR module, which performs the style transfer, is executed first. The resulting latent code is then modified before feeding it to the decoder. Replacing the $(z)_C$ with 0 gives us some rough representation of the style with only approximate shapes. On the other hand, if we replace $(z)_S$ with 0 and we keep $(z)_C$, a rather flat representation of the input with sharper is reconstructed. This demonstrates that $(z)_C$ represents the content, while $(z)_S$ holds most of the style-related information.

The fact that the latent code is passed through the TPFR module first means that the two-stage feature recombination is performed on the data we visualize. As a result, the decoded image $[0, (z)_S]$ slightly resembles the structure of the content image even if the $(z)_C$ is set to 0. Likewise, in case of $[(z)_C, 0]$, the geometry of the objects is already slightly modified based on the resulting style.

4. Computational overhead

Stylization of a single image of resolution 512×512 using our method takes approximately 16ms on a single *Titan-V100* GPU. Execution of the TPFR block takes approximately 3ms, which is 18.75% of the whole runtime. Due to memory requirements, our method can currently process images of size up to 768×768 pixels.

5. Quantitative evaluation

We are aware of the recent efforts bringing in quantities such as deception score [4] or content and style distribution divergence [2]. However we decided not to use these metrics as they are all based on a VGG network trained to classify paintings. We argue that such evaluation may favor models that have used VGG perceptual losses for training. This concern is closely related to the work of [1].

References

[1] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *ICLR*, 2019. 2

[2] Dmytro Kotovenko, Artsiom Sanakoyeu, Sabine Lang, and Bjorn Ommer. Content and style disentanglement for artistic style transfer. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4422–4431, 2019. 2

Figure 3. Qualitative evaluation of our method in zero-shot style transfer setting. The results clearly show that our method generalizes well to previously unseen styles.

[3] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 807–814, USA, 2010. Omnipress. 1

[4] Artsiom Sanakoyeu, Dmytro Kotovenko, Sabine Lang, and Björn Ommer. A style-aware content loss for real-time HD style transfer. *CoRR*, abs/1807.10201, 2018. 2

[5] Jan Svoboda, Jonathan Masci, Federico Monti, Michael M. Bronstein, and Leonidas J. Guibas. Peernets: Exploiting peer wisdom against adversarial attacks. *CoRR*, abs/1806.00088, 2018. 1

[6] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. *CoRR*, abs/1701.02096, 2017. 1

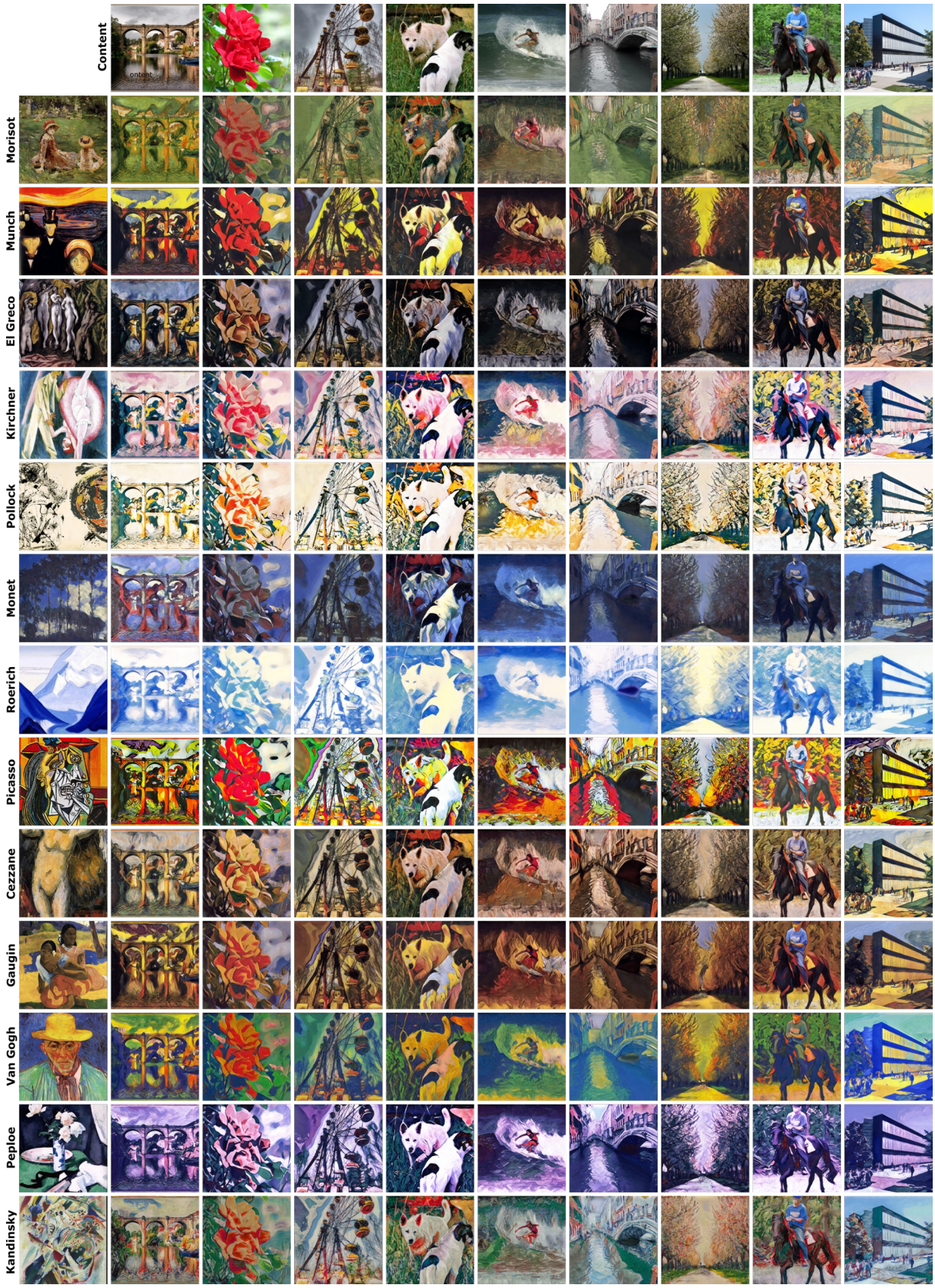

Figure 4. Qualitative evaluation of our method using disjoint set of painting styles that were in the training set.

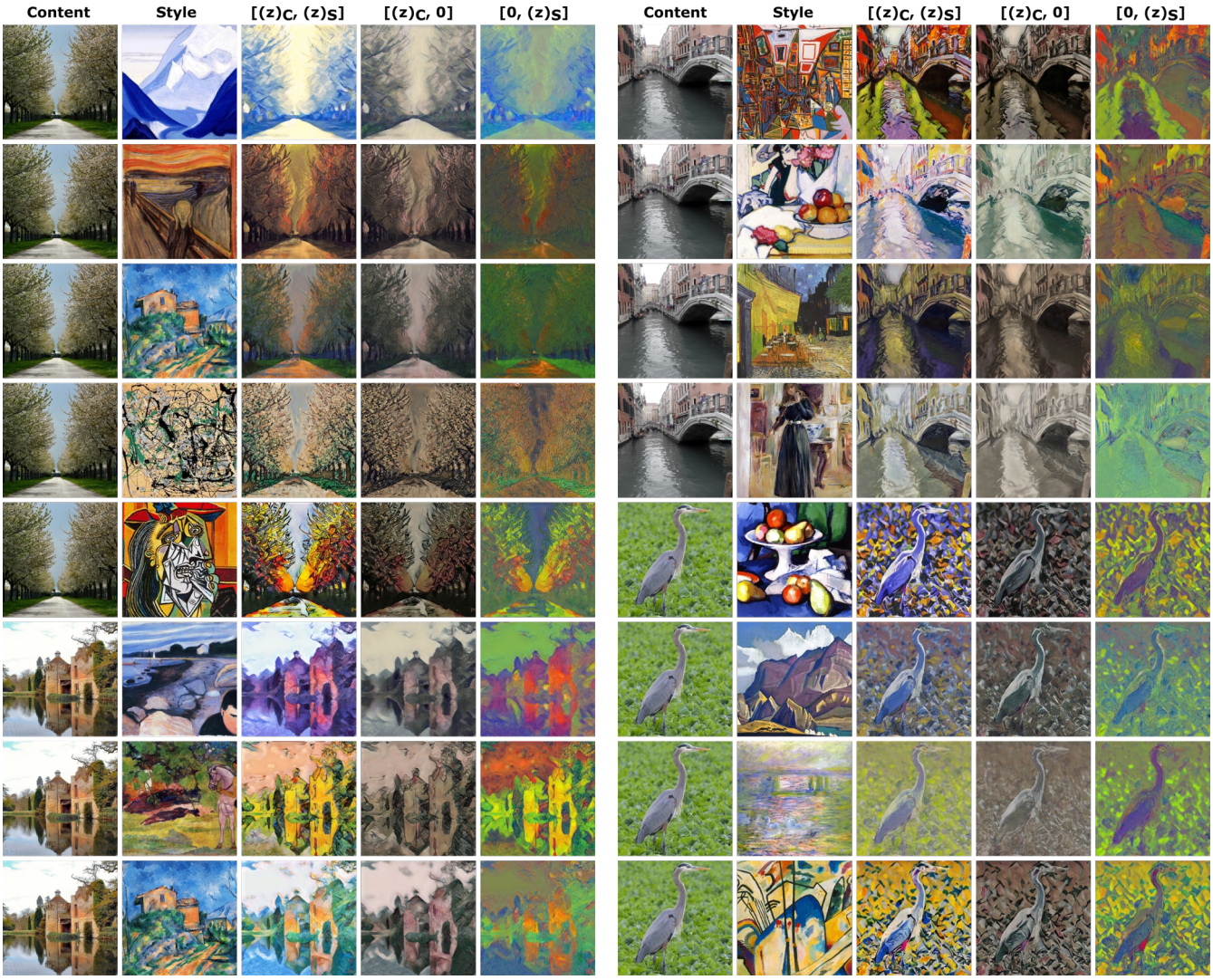

Figure 5. Visualization of information contained in content and style parts of the latent representation. Even if $(z)_C$ is set to 0, there is still some rough resemblance of the structure of the *Content* image, because the TPFR module transforms a partially local style features based on the content features.