Deeply Supervised Artist Style Transfer

Adrian Schneebeli, Alain Ryser, Nicolas Baumann, Tim Fischer

***Abstract*—Style transfer is a technique by which the content of an image is preserved while its style is modified. Existing methods either transfer the style from an input image and apply it to a content image, or learn a specific style which they are then able to apply to an arbitrary image. While the first type provides great amount of flexibility, it lacks a wider understanding of what a certain style amounts to. Conversely, the second type is severely limited in flexibility, since the network has to be retrained for each style. We propose an Auto Encoder (AE) architecture with deep supervision to learn the styles of artists and transfer images from one style to another. The style transfer is performed using normalisation and swapping of the encoded latent space statistics between source image and target artist embedding. Our method not only allows us the transfer of the style of a single image but rather of the complete works of a certain artist.**

I. Introduction

Image style transfer is the act of transforming an input image to a new target style. The input image has to be separated into content information and style information. The style information can then be swapped with a different style before the image is recomposed. Previous work deals with this problem by providing a content image and a style image. The resulting image should then combine the content and style of said images, i.e transferring the style onto the content, hence *style transfer*.

Due to the previous two-input image approach, the style transfer can only take place from one image to the other. Hence the transferred style is bound to a single image. A broader generalisation of style, would be to define it over the entire painting collection of an artist.

Therefore instead of transferring the style from one image onto another, this work focuses on learning the general style of various artists. Following this, their works can be visualized in any of the available artists style. The network follows an AE structure with an additional regularization branch. During training this branch learns to predict the artist of the input image from the latent space. This forces said latent space to contain meaningful information about an artist style.

The work presented here implements a similar network architecture as in [1] with instance normalization in the latent space. The architecture is extended to use a Variational Auto Encoder (VAE). Furthermore, style transfer results of various normalisation statistics are presented and compared. Finally, the classical AE architecture is compared to a β Variational Auto Encoder (β-VAE) approach. The effects of the disentangling properties of the latter, are analyzed with respect to the reconstruction and style transfer results for various β values in table I.

II. Related Work

The preliminaries on Convolutional Neural Network (CNN)s for style transfer was laid by Gatys et al. in [2]. Their method relies on an iterative optimization process to minimize the content and style loss. This combines the style and content of two images, effectively transferring the style from the later onto the content of the first. This approach is very flexible and can match style and content of any two arbitrarily chosen images. The downside is the long time it takes for the resulting image to converge. A possible solution to this are feed forward networks trained to reduce a similar loss function. Johnson et al. proposed a framework for calculating perceptual losses i.e. content and style loss from a pretrained network [3]. This method manages to solve the optimization of [2] with comparable results but up to 3 orders of magnitude faster. The gain in speed comes with a loss of flexibility as a separate network has to be trained for each new style. To mitigate this problem, Chen et al. introduce the concept of a StyleBank [4] containing explicit representations of an arbitrary number of pretrained styles. They use an AE structure to decouple the image style and content to perform the style transfer.

Huang and Belongie also used an AE network as they proposed the Adaptive Instance Normalisation (AdaIn) layer [1] to align the style of a target image to the input image. This allows for a fast style transfer with arbitrary input images. Burgess et al. proposed a training method for additional disentanglement of β-VAE [5].

III. Models and Methods

A. Architecture

The nature of our task forces us to train a model that is able to give a disentangled representation of an artist's painting style based on their images. As described in [5], so called β-VAE provide a way to achieve just that. We hence base our model on this architecture, but extend it with some additional tweaks. Data encoding is performed with a 34-layer ResNet [6], pretrained on ImageNet [7], which can be replaced with ResNets or VGGs of different depths. The output of the encoder is then passed through a *Reparametrization Layer*, which learns Gaussian parameters μ_{cij} and σ_{cij} per feature in each channel. Additionally, there is an injection layer that injects an artists style by using one of the normalization layers described in Section III-D.

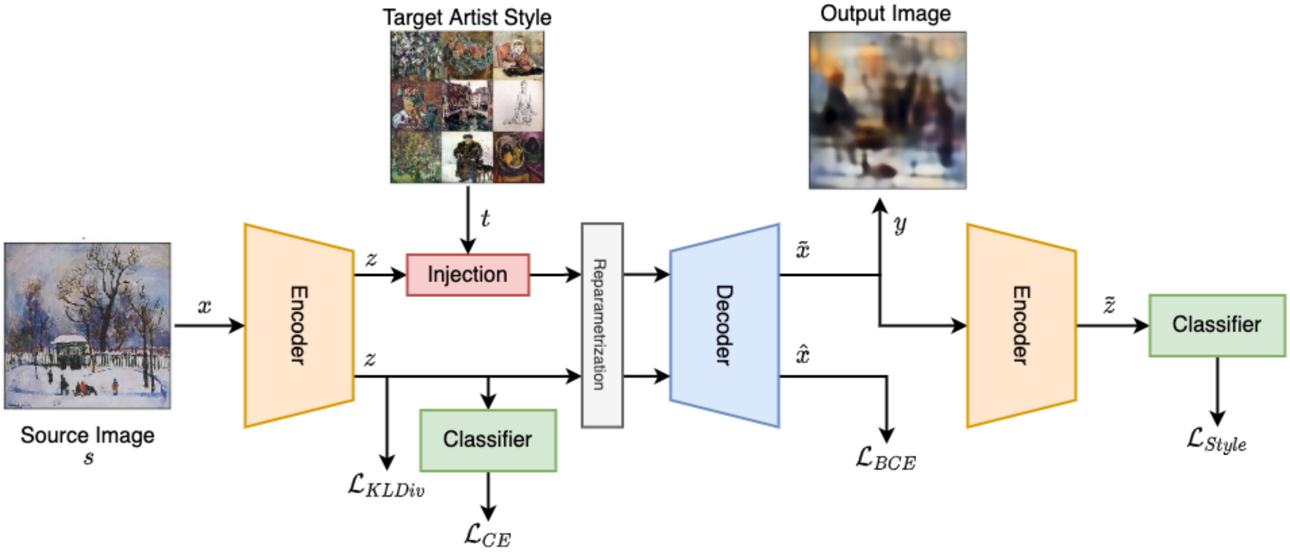

Figure 1. Let ω denote the latent classifier. For a single sample painting $x \in \mathbb{R}^{n \times m}$ of artist s, let z be its latent encoding before reparametrization, $\hat{x}$ the decoded reconstruction from the latent space after reparametrization, $\tilde{x}$ the reconstruction of x after injecting the latent space with a random artist $\tilde{t}$ and let $\tilde{z}$ be the latent encoding of $\tilde{x}$.

Subsequently, we apply the *Reparametrization Trick* [8] on the latent space. The decoder then mirrors the image pyramid of the ResNet encoder backbone. For each scale, it upsamples the feature map of the next lower scale using transpose convolutions and then applies an additional 3×3 convolution blocks to it. Finally, the bilinearly upsampled feature map of the next lower level is added on top of the learned feature map, which adds residual link to the decoding process. In order to regularize the latent space, we introduce deep supervision into our model. The idea is to add a classifier that extracts information about the artist directly from the latent space. The latent space is thus forced to contain information about the artist that drew a specific painting. Lastly, we examine the effect that an adversarial classifier has on the regularisation of the latent space by additionally optimising a style loss on injected latent spaces. See Figure 1 for a schematic of our architecture.

B. Loss

The model needs to be capable of performing the following tasks: 1) Reconstructing an image from it's latent space; 2) Extracting the artist style from a source image such that it encodes the style in the latent space and can be saved in an artist embedding; 3) Injecting a target artist style into the source artists latent space.

1) As we are training a VAE, we maximize the Evidence Lower BOund (ELBO) of the log-likelihood of our training sample, given by a Binary Cross Entropy (BCE) loss for reconstruction given a coding and the sum of KL divergences between learned $\mathcal{N}(\mu_{cij}, \sigma_{cij})$ per feature and a $\mathcal{N}(0,1)$ prior.

$$\begin{aligned}\mathcal{L}_{BCE} &= \sum_{c,i,j=1} (x_{cij} \log(\hat{x})_{cij}) \\ &\quad + (1 - x_{cij}) \log(1 - \hat{x}_{cij})) \\ \mathcal{L}_{KLDiv} &= \sum_{c,i,j=1} \mathrm{KL}(\mathcal{N}(\mu_{cij}, \sigma_{cij}) || \mathcal{N}(0,1)) \\ \mathcal{L}_{VAE} &= \mathcal{L}_{BCE} + \mathcal{L}_{KLDiv}\end{aligned}$$

2) We encourage style extraction by adding a latent classifier that is trained with the Cross Entropy (CE) loss. Further disentanglement of content and style is aided by the β parameter of the β-VAE. The latent spaces are then normalised by taking advantage of the normalisation methods discussed in section III-D and saved as an artist embedding $Z^{(a)}$.

$$\mathcal{L}_{CE} = -\omega(z)_s + \log(\sum_j \exp(\omega(z)_j))$$

3) Style injection is performed by sampling a random artist during training, applying the aforementioned target artist embedding, decoding the resulting latent space and feeding it through the encoder again. The deep classifier is then trained with the resulting encoding by applying an adversarial separate CE loss. This time with the injected artist as target. We call this additional loss function style loss.

$$\mathcal{L}_{Style} = -\omega(\tilde{z})_t + \log(\sum_j \exp(\omega(\tilde{z})_j))$$

Finally, the total loss amounts to:

$$\mathcal{L}_{TOT} = \mathcal{L}_{VAE} + \mathcal{L}_{CE} + \mathcal{L}_{Style}$$

C. Inference

During inference, we fix paintings and define target artists, whose style we wish to apply to the selected sample. The images are fed through the encoder to get their latent representations. The latent spaces are then transformed according to one of the mechanisms described in section III-D. We hence mapped the coding of the original paintings to the target artists distributions by injecting their statistics into the latent space. Finally, the modified feature map is put through the reparametrization layer and fed through the decoder to arrive at the transferred image *y*.

D. Normalisation Levels

As has been shown with AdaIn [1], one can apply style transfer by swapping and normalising the statistics of source artist s' latent space and target artist t's latent space. In this subsection we introduce the normalisation techniques and statistics with which we performed the injection in the model. We define a latent space injection by:

- Extracting the source mean and variance with one of the described methods, then normalising the latent space by subtracting the mean and dividing by the variance, as in eq. (1).
- Extracting the target mean and variance, then denormalising the acquired representation by multiplying the variance and adding the mean, as in eq. (2).

$$Norm(x, \mu^{(s)}, \sigma^{(s)}) = \frac{(x - \mu^{(s)})}{\sigma^{(s)}} \tag{1}$$

$$DeNorm(N_x, \mu^{(t)}, \sigma^{(t)}) = N_x * \sigma^{(t)} - \mu^{(t)} \tag{2}$$

where multiplication and division executed element-wise and:

x	= Latent space of input image
N_x	= Normalised latent space
$\mu^{(s)}, \mu^{(t)}$	= Source/Target mean
$\sigma^{(s)}, \sigma^{(t)}$	= Source/Target variance

fig. 2 visualises the effect that different normalisation levels have on the injection of the input images latent space. In the following we describe different methods to extract source and target means and variances in order to perform artist injection.

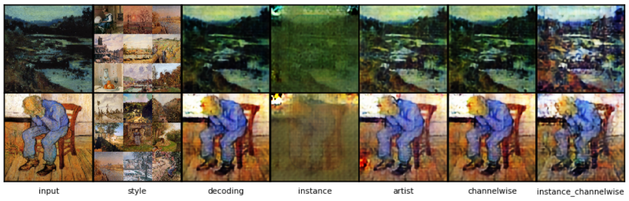

Figure 2. Comparison of the different normalisation levels. Every row shares the same input data.

1) Instance Normalisation: During training, the latent space encodings $Z^{(a)}$ of each artist a are extracted and used to compute the standard deviation and the mean per feature and channel. During inference, the standard deviation and mean of the source latent space of the input instance (hence the name *instance*) is calculated as:

$$\mu_{cij}^{(s)} = \frac{1}{CHW} \sum_{c,h,w=1}^{C,H,W} x_{chw}$$

$$\sigma_{cij}^{(s)^2} = \frac{1}{CHW} \sum_{c,h,w=1}^{C,H,W} (x_{chw} - \mu^{(s)})^2$$

$$\mu_{cij}^{(t)} = \frac{1}{|Z^{(s)}|} \sum_{z \in Z^{(s)}} z_{cij}$$

$$\sigma_{cij}^{(t)^2} = \frac{1}{|Z^{(s)}|} \sum_{z \in Z^{(s)}} (z_{cij} - \mu_{cij}^{(t)})^2$$

We then apply the injection procedure described before which yields the injected samples. As can be seen in fig. 2 this normalisation results in very aggressive injections.

2) Artist Normalisation: Opposed to the instance normalisation, we do not extract the mean and variance of the input latent space as in eq. (1), but the mean and variance over all $Z^{(a)}$ of the input artist a. More concretely:

$$\mu_{cij}^{(s)} = \frac{1}{|Z^{(s)}|} \sum_{z \in Z^{(s)}} z_{cij}$$

$$\sigma_{cij}^{(s)^2} = \frac{1}{|Z^{(s)}|} \sum_{z \in Z^{(s)}} (z_{cij} - \mu_{cij}^{(s)})^2$$

The latent space is injected by the mean and variance of the target artists embeddings, i.e. the same way as with *Instance Normalization*. As can be seen in fig. 2, injections are much more subtle.

3) Artist Channelwise Normalisation: This normalisation level is similar to *Artist Normalisation*, but extracts the statistics channelwise. This means that, instead of calculating the mean and variance per feature and channel, we extract the statistics as one scalar value per channel, i.e. for input artist s and target t, with latent feature maps of size $M \times N$

$$\mu_{cij}^{(s)} = \frac{1}{|Z^{(s)}|MN} \sum_{z \in Z^{(s)}} \sum_{m,n=1}^{M,N} z_{cmn}$$

$$\sigma_{cij}^{(s)^2} = \frac{1}{|Z^{(s)}|MN} \sum_{z \in Z^{(s)}} \sum_{m,n=1}^{M,N} (z_{cmn} - \mu_{cij}^{(s)})^2$$

$$\mu_{cij}^{(t)} = \frac{1}{|Z^{(t)}|MN} \sum_{z \in Z^{(t)}} \sum_{m,n=1}^{M,N} z_{cmn}$$

$$\sigma_{cij}^{(t)^2} = \frac{1}{|Z^{(t)}|MN} \sum_{z \in Z^{(t)}} \sum_{m,n=1}^{M,N} (z_{cmn} - \mu_{cij}^{(t)})^2$$

This is essentially a parameter free and non learnable form of instance normalisation as in [9]. The reduction of the injection tensors to a lower dimensional space results in a less aggressive style transfer, as not every single element of the latent space is normalised, but only the channel itself. In the case of *Artist Normalisation* the channelwise approach does not seem to alter the outcome too much, as the input latent space is again normalised by the entire input artists embedding. Thus the artist based normalisation techniques both seem to be too restrictive to perform style transfer.

4) Instance Channelwise Normalisation: Lastly, the very aggressive *Instance Normalisation* and the more subtle approach of the channelwise mean and variance computation are combined. Instead of extracting the source mean and variance from the the input artists embedding, we extract them only from the input latent space x, i.e.

$$\mu_{cij}^{(s)} = \frac{1}{MN}\sum_{m,n=1}^{M,N} x_{cmn}$$

$$\sigma_{cij}^{(s)^2} = \frac{1}{MN}\sum_{m,n=1}^{M,N} (x_{cmn} - \mu_{cij}^{(s)})^2$$

Target mean/variance for denormalisation eq. (2) is then extracted by the channelwise mean and variance of the output artists embedding, as with *Artist Channelwise Normalisation*. Comparing *Instance Normalisation* with the *Instance Channelwise Normalisation*, one notices how the more subtle and regularized channelwise normalisation technique results in a much more subtle style transfer.

IV. Results

A. Dataset

To train our model, we adapted the Kaggle Dataset [10] and trained on the images of the 10 most frequently appearing authors, resulting in a total of 3948 samples for training. Most of the digitalized paintings were originally provided by the WikiArt [11] database.

B. Training

To train the experiment runs described below, we used the Adam Optimizer with a learning rate of 10^{-4}. We trained with a batch size of 8 for 512 epochs in total. Additionally, to make the training generalize better, we applied various data augmentation techniques. All images were first scaled to a size of 512×512 pixels, from which we removed random vertical and horizontal stripes (random cropping) to arrive at a size of 256×256 pixels. Further, we applied random scaling between factors 0.9 and 1.1, randomly reflected the images horizontally and vertically and also allowed for rotations of up to 2π.

C. Experiments

As described in [5], increasing/decreasing the β parameter in β-VAE can result in different levels of style disentenglament since it leads to greater values of the $\mathcal{L}_{KLDiv}$ term in the loss, effectively penalizing information flow from the encoder to the decoder. We thus experiment with different β values in the training. To realize the impact of having a β-VAE, we also compare to a non-VAE by disabling the reparametrization layer. Additionally choosing between normalization levels out of the ones described in section III-D influences the style loss, thus having impact on the effectiveness of the injection model. Since *Instance* and *Instance Channelwise* normalization levels seem to have the most impact on the decoded image, we choose these two for our experiments. We evaluate the effectiveness of the injection model, by first reporting precision/recall of the prediction of the latent classifier when encoding the input image; then compare it to the precision/recall of the latent predictions when encoding the injected image. Note that precision/recall is computed for each class separately and then averaged over all classes. The results can be found in table I and a comparison of the instance normalised networks is visualised in fig. 3.

Norm Level	β	Precision Input	Recall Input	Precision Injection	Recall Injection
Instance	No VAE	0.086	0.114	0.098	0.029
Instance	1	0.053	0.094	0.087	0.029
Instance	5	0.064	0.095	0.121	0.046
Instance	20	0.058	0.100	0.101	0.035
Instance Channel	No VAE	0.0479	0.133	0.0938	0.0499
Instance Channel	1	0.0586	0.157	0.108	0.0724
Instance Channel	5	0.046	0.0769	0.0765	0.0205
Instance Channel	20	0.0686	0.082	0.0757	0.0182

Table I
RESULTS OF EXPERIMENT WITH DIFFERENT β AND NORMALIZATION LEVELS.

D. Comparison with AdaIn method

In this subsection, we compare our approach with the AdaIn [1] that is constrained within the bounds of the abilities of our encoder and decoder. Further it is to mention, that the AdaIn encoder is trained with a different style loss function, which our implementation does not consider. As our approach deviates with the common methodology of style transfer, namely that AdaIn (and others) use an input image and a target image, contrary to our approach of defining the target style as an artist embedding, the comparison of the methods has to be taken with a grain of salt, as they try to answer different questions. Specifically AdaIn poses the question: *How would this content image*

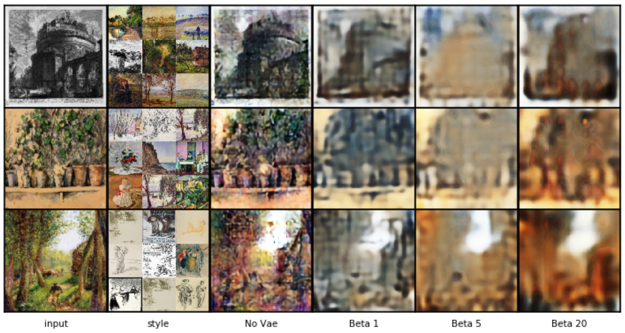

Figure 3. Visualisation of the instance normalised networks from table I and the effect of the β parameter.

look like in this style images style? While our approach gives rise to the question: *What would Picasso have done with this input image?* Thus our approach is a broader generalisation.

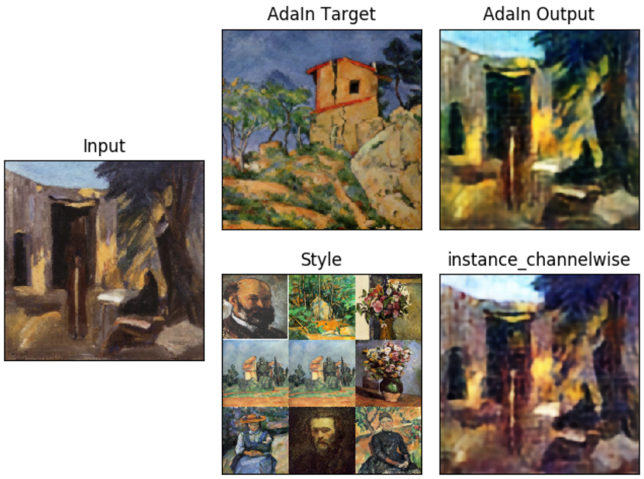

Figure 4. Visualisation of the difference between the AdaIn approach by [1] in contrast to our approach with the same input image.

fig. 4 Shows how the aforementioned AdaIn implementation in section IV-D and our approach differ in results, by inputting the same input image for both approaches. Due to the different nature of the methods, AdaIn gets a single style image, that is within the embedding of our target artists embedding.

V. Discussion

From the results seen in table I, we can make out a trend that the precision of our classifier seems to increase, when we inject an artist into an input image, while recall seems to decrease. We can thus follow, that the correct (injected) artist is classified with more certainty after injection, even though less frequently as suggested by the recall. This implies, that injection does not always work. But if it does, it is able to convince our model that the injected image was in fact painted by the injected artist. Due to the fact that the recall and precision values are rather low and decoded images of the β-VAE seem to be pretty blurry, it is hard to draw a definitive conclusion. Interestingly, the images of the non-VAE are reconstructed much more accurately and judging from eye alone, the injected images seem to mimic the injected artists style slightly better. Further investigating into non-variational models might thus be worthwhile. It would also be interesting to see how the results change, if our model was trained on a larger dataset. Unfortunately, because of the nature of our task, it is hard to find an adeuqate dataset. Artists usually only create a limited amount of paintings during their lifetime and taking even more artists into our dataset would actually increase the difficulty of the learning task instead of leveraging it.

VI. Summary

In this project, we present a novel approach to image style transfer by regularizing the encoded latent spaces of images, instead of transferring the style of a single image onto another one. Our method captures the general style of an artist and is able to transfer it onto an image painted by a different artist. Thanks to the deeply supervised AE architecture and the novel losses we introduced, the artist style representations could be disentangled. We experimented both with normal AE and β-VAEs and were able to show that while β-VAE are an interesting approach due to their inherent disentanglement capabilities, the reconstructed images are too blurry and are not able to properly represent the style of the target artist. Further we proposed and compared various normalization methods, which produced different but nevertheless interesting results.

In future work, we will try to examine how the intra-class variance influences the extraction and injection occurring in the latent space. As, over the lifetime of an artist, its individual style might evolve significantly which would produce ambiguous results. Further, our architecture still has room for improvements and modifications, since the time spent on training our network was rather short due to limited time and resources.

Acknowledgements

The authors thank the reviewers of our proposal for their helpful feedback and guiding us in the right direction. The training of our network was carried out on the Leonhard GPU cluster managed by the HPC team at ETH Zurich.

References

[1] X. Huang and S. J. Belongie, “Arbitrary style transfer in real-time with adaptive instance normalization,” *CoRR*, vol. abs/1703.06868, 2017. [Online]. Available: http://arxiv.org/abs/1703.06868

[2] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[3] J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution,” in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Springer International Publishing, 2016, pp. 694–711.

[4] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua, “Stylebank: An explicit representation for neural image style transfer,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[5] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, “Understanding disentangling in β-vae,” 2018.

[6] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[7] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[8] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” 2014.

[9] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis,” 2017.

[10] Kaggle, “https://www.kaggle.com/c/painter-by-numbers/overview.”

[11] WikiArt, “https://www.wikiart.org/.”