

Research of Neural Network Image Style Transfer

T.M. Shamsutdinova¹

Bashkir State Agrarian University, Ufa, Russia

¹ ORCID: 0000-0003-1809-3615, tsham@rambler.ru

Abstract

The purpose of this study is to consider theoretical and practical issues of using convolutional neural networks for image style transfer.

The objectives of the study are to analyze literary sources on the problem, to consider, test and compare some neural network models (with open source code) for the purpose of their further study.

In particular, four neural networks that implement style transfer by using the TensorFlow library were selected and studied. In some cases (model 2 and model 4), it was necessary to modify the Python code. The Colab environment was used for the experiments. Fragments of famous paintings, as well as photographs and stylized images of multi-colored geometric figures, were used as test images.

A conclusion is made about the quality of the studied models. It is concluded that despite the current achievements in the field of style transfer, researchers still face a number of challenges, including improving the quality of image generation and reducing computational costs.

Keywords: neural networks, modeling, style, style transfer, Tensorflow.

1. Introduction

Image style transfer using neural networks is a promising area in computer vision and artificial intelligence. This technology enables image transformation, preserving their content but altering the visual style according to a given example, creating unique artistic works.

The paper [1] describes artistic style transfer as a method «used to create art by synthesizing global and local style patterns from a given style image evenly over a content image while maintaining its original structure».

The article [2] notes that artistic style transfer can be defined as the creation of a stylized image from a content image and a style image. Image content and style can be defined as follows:

- two images are similar in content if their high-level features as extracted by an image recognition system are close in Euclidean distance. This definition stems from the observation that high-level features of pre-trained image classification systems are tuned to the semantic information in the image [3];
- two images are similar in style if their low-level features as extracted by an image recognition system share the same spatial statistics. From this perspective, artistic style can be viewed as a visual texture [4].

The article [5] notes that in recent years, neural network technologies known as neural style transfer have transformed the possibilities for creating stylized images. Neural networks are used to extract statistical factors that assess the effectiveness of transfer, related to both content and style.

Neural style transfer (NST) refers to a class of software algorithms that process digital images to adopt the appearance or visual style of another image. A common application of NST

is the creation of artistic works from photographs, for example, by transferring the style of famous paintings.

The problem of style transfer to image content was treated in earlier versions of NST as an image optimization problem, leading to labor-intensive and time-consuming iterations. To address this issue, a more efficient method called "fast style transfer" was introduced. While fast style transfer also uses deep neural networks, it does so by training an independent model that can modify any image in a single feedforward pass.

The purpose of this paper is to examine the theoretical and practical aspects of using convolutional neural networks for image style transfer.

The objectives of the study are to analyze bibliographic sources on the problem under study and to review and test several open-source neural network models for further study, including for use in university teaching.

2. A Review of the Evolution of Image Style Transfer

2.1. Analogies and Non-Photorealistic Rendering

One of the first style transfer methods was the image analogy method.

The image analogy method, described in [6] (2001), is based on multiscale autoregression, which is also used in texture synthesis. By selecting different types of source image pairs as input, the image analogy method supports a wide range of image filter effects, including traditional filters such as:

- blurring;
- embossing;
- super-resolution, in which a higher-resolution image is inferred from a low-resolution source;
- texture transfer, in which images are "texturized" with some arbitrary source texture;
- artistic filters, in which various drawing and painting styles are synthesized based on scanned real-world examples;
- textures created using a simple painting interface, etc.

The next important direction in style transfer is non-photorealistic rendering methods. Non-photorealistic rendering (NPR) is an image-creation tool that doesn't necessarily adhere to the concept of photorealism.

NPR can be considered a computer graphics technique used to create images that don't require maximum realism. Instead of realism, NPR focuses on stylization, artistic expression, and the conveyance of a specific mood or atmosphere. This approach is often used in illustrations and other forms of visual art.

The paper [7] notes that «non-photorealistic rendering has two complimentary goals: the communication of information using images; and rendering images in interesting and novel visual styles which are free of the traditional computer graphics constraint of producing images which are "life-like".

Initially, NPR was based on interactive painting systems using airbrushes and pencils. Brush painting then evolved, incorporating more complex models for brushes, canvas, brushstrokes, and so on. Post-processing systems then emerged, allowing specialized methods to be applied to photographic images to synthesize artistic visualizations of these images in a painting-like style. This involves several elements, such as strokes, tone, and texture, as well as contours and others.

An example of one type of rendering is stroke-based rendering. Stroke-based rendering is a method for creating virtual strokes (e.g., brushstrokes) on a digital canvas to render a photograph in a specific style. Modern rendering methods include a variety of image processing and filtering techniques, texture generation, and more. To achieve a non-photorealistic effect, various techniques are used, such as contour drawing, texturing, color palettes, filters, chiaroscuro, and others.

2.2. Neural Style Transfer Based on the NST Algorithm (Leon A. Gatys, Alexander S. Ecker, Matthias Bethge)

The use of convolutional neural networks has been a real breakthrough in the field of style transfer. The key event in this field was the paper "A Neural Algorithm for Artistic Style" by Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge, published in 2015 [8].

The method proposed in the paper was called NST (Neural Style Transfer).

NST relies on a feature map of object layers, which is used, among other things, in image classification. The paper described the architecture of the VGG-19 network, which was pre-trained to perform object recognition using the ImageNet dataset.

The ImageNet database is a large-scale project to create and maintain a database of annotated images structured for use in computer vision and pattern recognition. As noted on the project's official website [9], the database contains over 14 million image URLs. Each image was manually annotated, including the list and coordinates of objects within the image.

Leon A. Gatys et al. made the important conclusion that the loss function minimized during stylized image synthesis contains two important entities: content and style.

Let p and x be the original image and the image that is generated, and P^l and F^l be their respective feature representation in layer l .

We then define the squared-error loss between the two feature representations $L_{content}(p, x, l)$ as [8]:

$$L_{content}(p, x, l) = \frac{1}{2} \sum_{i,j} (F_{i,j}^l - P_{i,j}^l)^2 \quad (1)$$

To generate a texture that matches the style of a given image, we perform gradient descent on a white noise image to find another image that matches the feature responses of the original image. This is done by minimising the mean-squared distance between the entries of the Gram matrix G from the original image and the Gram matrix of the image to be generated. In the context of convolutional networks, the Gram matrix is a matrix containing multiple correlations of all features extracted by the layer's filters:

$$G_{i,j}^l = \sum_k F_{ik}^l F_{jk}^l \quad (2)$$

Next, let a and x be the original image and the image that is generated, and A^l and G^l be their respective style representations in layer l .

The loss function for the style representation at layer l with feature map dimensions N_l by M_l is then:

$$L_{style,l}(a, x) = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_{i,j}^l - A_{i,j}^l)^2 \quad (3)$$

The loss for style representation will then be:

$$L_{style}(a, x) = \sum_l w_l L_{style,l}(a, x) \quad (4)$$

And the overall loss function is:

$$L_{sum} = \alpha L_{content}(x, p, l) + \beta L_{style}(a, x) \quad (5)$$

Where w_l are the weighting coefficients for each layer's contribution to the overall style loss, and α and β are the weighting factors for content and style reconstruction respectively.

This approach allows for a greater emphasis on either the transfer of content or the transfer of style. Emphasizing style will produce images that match the texture of the stylized artwork, but with a loss of clarity in the content of the original image. With a significant emphasis on content, the original objects in the image can be clearly identified, but certain aspects of the stylized transferred texture will be lost.

The approach proposed in [8] can be implemented programmatically as the following algorithm:

- loading the VGG-19 network model with ImageNet weights;
- selecting layers of the trained VGG-19 model to represent content and style;
- creating a custom model based on the loaded VGG-19 model;
- creating a content loss function based on formula (1);
- creating a Gram matrix and a function for calculating style loss using formulas (2)-(4);
- adjusting optimizer parameters;
- creating a total loss function by selecting weighting coefficients for the style loss and content loss functions, taking into account formula (5);
- determining the number of transfer epochs (iterations) and/or the generation termination condition;
- starting the iterative style transfer process.

2.3. Evolution of Neural Network Style Transfer Methods

The paper [10] provides an interesting overview of the evolution and classification of style transfer methods. The following neural network style transfer models are distinguished:

- Image-Optimization-Based NST (IOB-NST), which transfer the style by iteratively optimising an image;
- Model-Optimization-Based NST (MOB-NST), which optimize a generative model offline and produces the stylised image with a single forward pass.

Depending on the number of artistic styles the network can generate, MOB-NST algorithms are further divided into the following methods:

- PSPM-MOB-NST (Per-Style-Per-Model) implements a idea, which is to pre-train a feed-forward style-specific network and produce a stylised result with a single forward pass at testing stage;
- MSPM-MOB-NST (Multiple-Style-Per-Model) - neural methods that train multiple styles in a single model. Many paintings (e.g., impressionist paintings) share similar paint strokes and only differ in their colour palettes, which allows for the inclusion of multiple styles in a single model;
- ASPM-MOB-NST (Arbitrary-Style-Per-Model) - neural methods that train an arbitrary number of styles in a model. These methods aims at one-model-for-all, i.e., one single trainable model to transfer arbitrary artistic styles. There are also two types of ASPM, one built upon Non-parametric Texture Modelling and the other one built upon Parametric Texture Modelling with Summary Statistics.

Based on this classification, the following groups of neural network style transfer methods can be distinguished:

- methods based on image optimization, the so-called slow optimization, developed by Leon A. Gatys et al. The key idea behind their algorithm is to iteratively optimise an image with the objective of matching desired feature distributions, which involves both the content information and style information. Their proposed algorithm successfully produces stylised images with the appearance of a given artwork. A drawback is the high computational complexity of the algorithm, requiring a long execution time;
- more recent fast style transfer methods, such as "perceptual loss" methods, which use high-level features to improve quality, while traditional methods train the network by comparing individual pixels;
- combined methods. In particular, work [3] proposes a method in which the authors combine the benefits of feed-forward image transformation tasks and optimization-based methods for image generation by training feed-forward transformation networks with perceptual loss functions. The authors applied this method to style transfer, where they achieved comparable performance and drastically improved speed compared to existing methods, and showed that training with a perceptual loss allows the model to better reconstruct fine details and edges.

Further research into style transfer problems has led to the development of new practice-oriented network architectures. For example, [1, 11] present a style-attention network (SANet), which effectively integrates local style patterns according to the semantic spatial distribution of image content. Article [12] describes a convolutional style transfer network for object detection among other tasks.

3. Research Methods

We will consider and test examples of neural networks capable of performing image style transfer. All models under consideration are open-source projects and can be used to study neural network-based image style transfer.

Specifically, four neural networks (Table 1) implementing style transfer using the TensorFlow library were selected and studied. In some cases (Model 2 and Model 4), modifications to the Python code were required due to the use of outdated (no longer supported) syntax for programming methods from linked libraries in the original sources.

Table 1. Models used

Model No.	Code Source	Pre-trained Network Used
1	Fast Style Transfer for Arbitrary Styles [13]	google/magenta
2	NeuralStyleTransfer_experiments [14] Ilya Ginsburg. Drawing AI: Uncovering the Secrets of Neural Style Transfer [15]	VGG19, weights='imagenet'
3	Picard. How to Copy Warhol's Style with VGG-19 Neural Networks, Transfer Learning, and TensorFlow [16]	VGG19, weights='imagenet'
4	Image Style Transfer with Keras and Tensorflow [17] Neural Style Transfer with Eager Execution. Neural Style Transfer with tf.keras [18]	VGG19, weights='imagenet'

Fragments of paintings were used as test images:

- Vincent van Gogh "Sunflowers (1888)";
- Vincent van Gogh "Wheat Field with Cypresses";
- Edvard Munch "The Scream";
- Ivan Konstantinovich Aivazovsky "The Storm";

as well as the author's own photograph of Minsk views and stylized images of multicolored geometric shapes.

The Colab environment [19] was used for the experiments.

4. Results

A series of computational experiments on image style transfer were conducted using the neural networks described in Table 1. Each of the models examined demonstrated distinct results, generating stylized images of varying quality.

4.1. Model 1

Model 1, which is one of the built-in examples of the Tensorflow library and therefore considered professional-grade, yielded the most artistically interesting results. Unfortunately, however, this model only provides limited code availability, making it difficult to study the style transfer parameter settings.

Figure 1 shows the results of the style transfer, which, as we can see, is distinguished by a high degree of artistic realization. The contours of the original figures are clearly defined (without significant blurring). However, it should be noted that in some cases, the color scheme of the transferred style overlaps with the color of the original objects, which is particularly noticeable in the example of Vincent van Gogh's "Sunflowers (1888)."



Fig. 1. Results of style transfer by model 1

To study the preservation of the original object's shape during style transfer, a series of tests were performed on drawings of geometric primitives (filled and unfilled squares, as well as diamonds, circles, and triangles filled in different colors). The test results are presented in Figure 2, where columns "C" contain images of the original drawings (content), columns "S" contain style drawings, and columns "R" contain the stylized results.

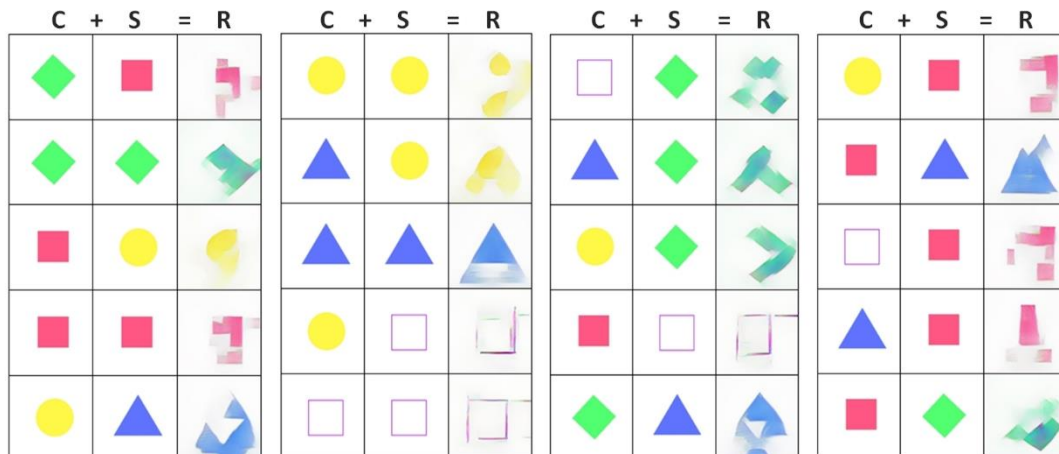


Fig. 2. The result of style transfer, where columns "C" contain the original images (content), "S" are the style images, "R" are the results of stylization

As we can see in Figure 2, in all cases, the color of the original geometric shape (C) was displaced by the color of the style shape (S). The resulting image (R) was based on the color of the applied style (with some blurring from the color palette) and a shape with elements of partially overlapping shapes, stretched and blurred along the outline. Part of the shape's center was replaced with a white background color.

It can be assumed that the results of this test are due to the simple shape of the geometric primitives (apparently not recognized by the model as the main content of the image and taken as a background element) and the specific white background color of the original images.

4.2. Model 2

The second convolutional neural network under study (see Table 1) is based on a pre-trained VGG19 image classifier, accessible via the Keras framework API.

The test was conducted on images scaled to 256x256 pixels. Figure 3 shows the original images for stylized transfer, and Figure 4 shows the result obtained with style transfer (with different iteration values).

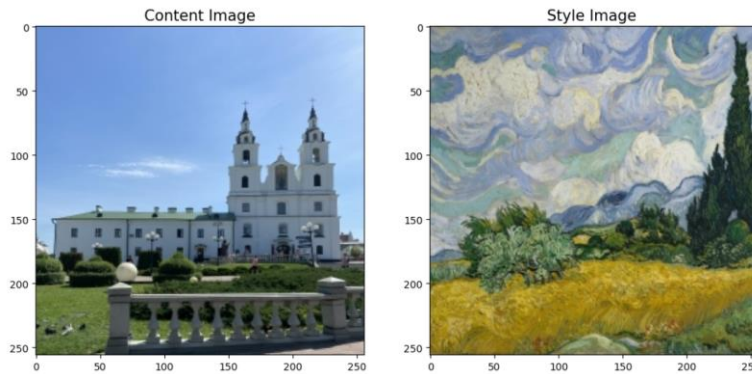


Fig. 3. Source images for style transfer

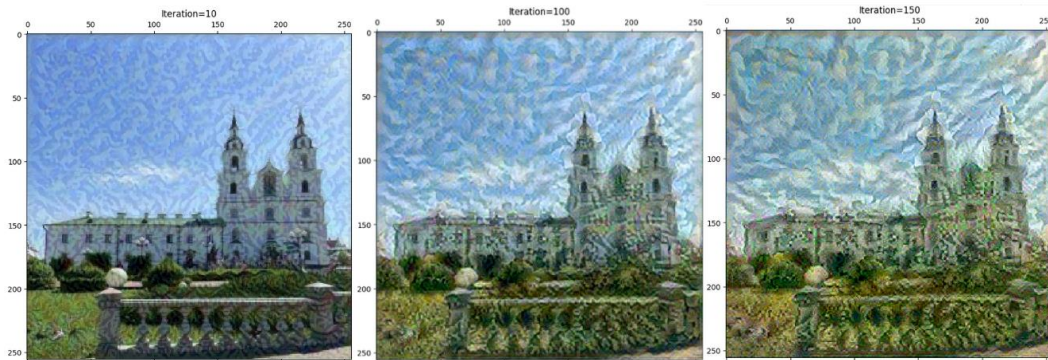


Fig. 4. Result of the style transfer process with 10, 100, and 150 iterations

The VGG-19 network, with 19 layers, is designed for object recognition in images. It has 16 convolution layers and three fully connected layers (another implementation of VGG-16 exists, containing 13 convolution layers and three fully connected layers).

The VGG series of models are deep neural networks consisting of common modules [20]:

- convolutional modules. A full-color RGB image with three channels is fed to the network's input. The image is then passed through convolutional layers, each with filters with 3x3 kernels;
- max-pooling layers. After some convolutional modules, max-pooling layers are added with a 2x2 filter and a stride of 2 to downsample the feature maps. The filter reduces the width and height by a factor of two, but maintains the same number of channels;
- fully connected layers. The network terminates with three fully connected layers. The last layer has 1,000 channels, corresponding to the 1,000 classes in ImageNet;
- softmax layer, which outputs the probability distribution over classes.

Based on the concept of the network structure, [8] concludes that such a network (without the final, fully-connected classifying part) can be used to extract features from objects whose contours must be considered during style transfer.

Examples of the application of VGG networks to style transfer problems are given, for example, in [21-22] and others.

Using the code presented in Model 2, one can obtain a feature map corresponding to the representation of the VGG19 network layers (Figs. 5a, 5b).

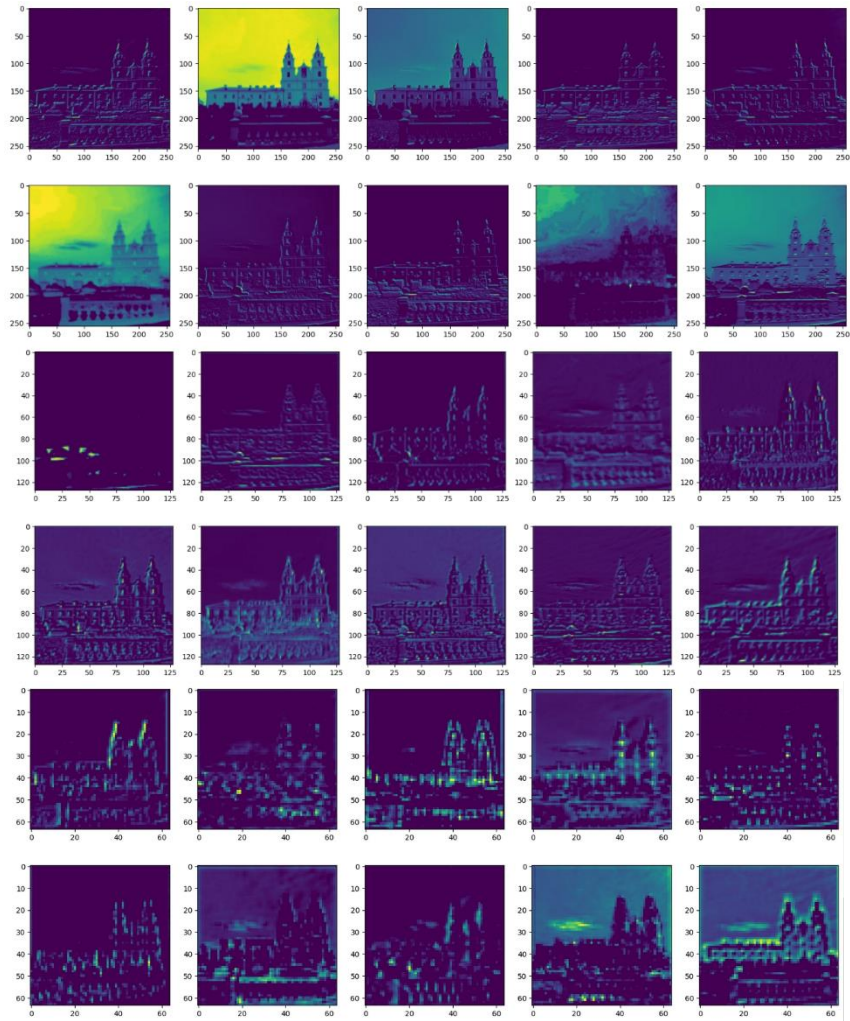


Fig. 5a. Display of the original photograph by various VGG19 convolutional modules (beginning of figure)

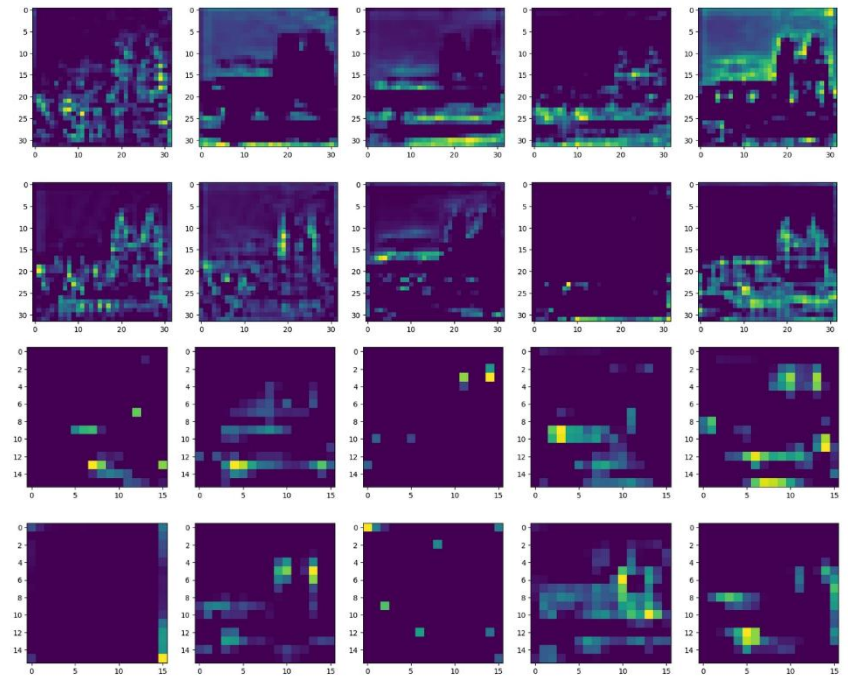


Fig. 5b. Display of the original photograph by various VGG19 convolutional modules (continued)

4.3. Model 3

Model 3 (see Table 1) is also based on the pre-trained VGG19 network with the 'imagenet' weight set.

The model's performance was studied using small graphical images, which were then transformed to dimensions of $\text{max_dim} = 128$ and $\text{max_dim} = 64$. The network's performance was also examined with varying parameters such as the `tf.optimizers.Adam` optimizer learning rate and the number of iterations.

Figure 6 shows the network's performance with the following parameters:

- `max_dim = 128`;
- `style_weight=1e-2`;
- `content_weight=1e4`;
- `tf.optimizers.Adam(learning_rate=0.01, beta_1=0.99, epsilon=1e-1)`;
- `tf.optimizers.Adam(learning_rate=0.02, beta_1=0.99, epsilon=1e-1)`;
- The number of iterations was 200, 400, and 1000.

It should be noted that further increases in the number of iterations resulted in increased blurriness and loss of clarity in the resulting image.

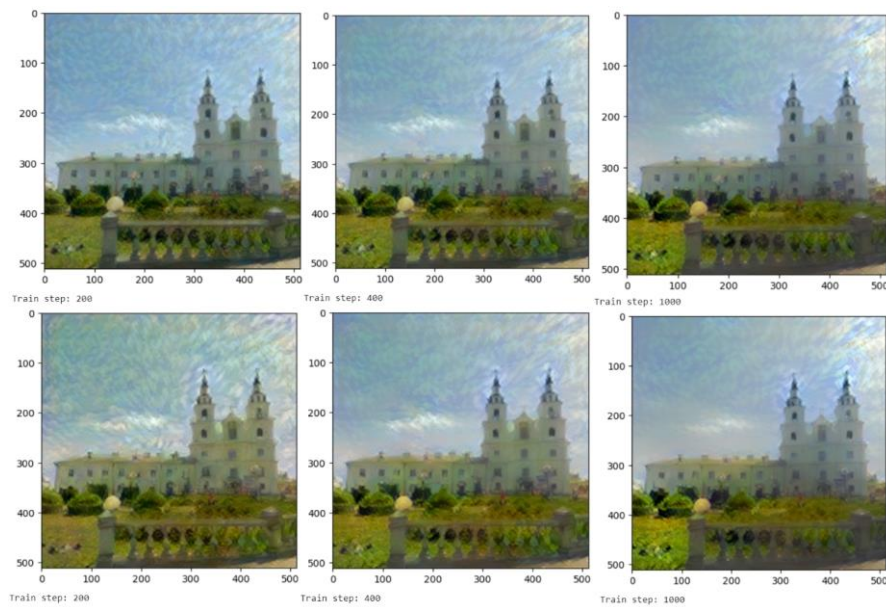


Fig. 6. Image stylization results with the following parameters: $\text{max_dim} = 128$, Adam optimizer learning rate = 0.01 (top row), learning_rate = 0.02 (bottom row) for 200, 400, and 1000 iterations.

The network's performance results for $\text{max_dim} = 64$ are shown in Figures 7-8.

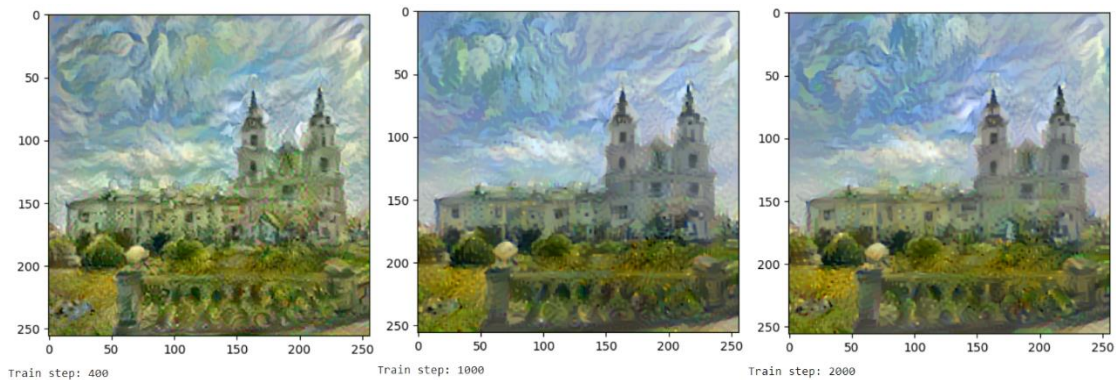


Fig. 7. The result of image stylization with the parameters: $\text{max_dim} = 64$, the learning rate of the Adam optimizer learning_rate = 0.01 for 400, 1000 and 2000 iterations



Fig. 8. Initial data for stylization (top row) and style transfer results (bottom row) with $\text{max_dim} = 64$ and iterations of 300 and 2000.

From these tests, we can conclude that the model under consideration is more suitable for small thumbnail images. Overall, the model demonstrates fairly high (fast) performance. However, as the image size increases, the contours begin to blur and the image becomes out of focus.

For each loaded image, you must manually adjust the settings to create the desired artistic effect.

4.4. Model 4

Model 4 (see Table 1) was run on 224x224 pixel images.

Results for different source images are shown in Figures 9 and 11.

The following settings were used:

- `tf.compat.v1.train.AdamOptimizer(learning_rate=0.5, beta1=0.99, epsilon=1e-1);`
- `content_weight=1e3;`
- `style_weight=1e-2.`

The existing open-source model code was supplemented by plotting the loss function during network optimizer training (Figures 10 and 12).

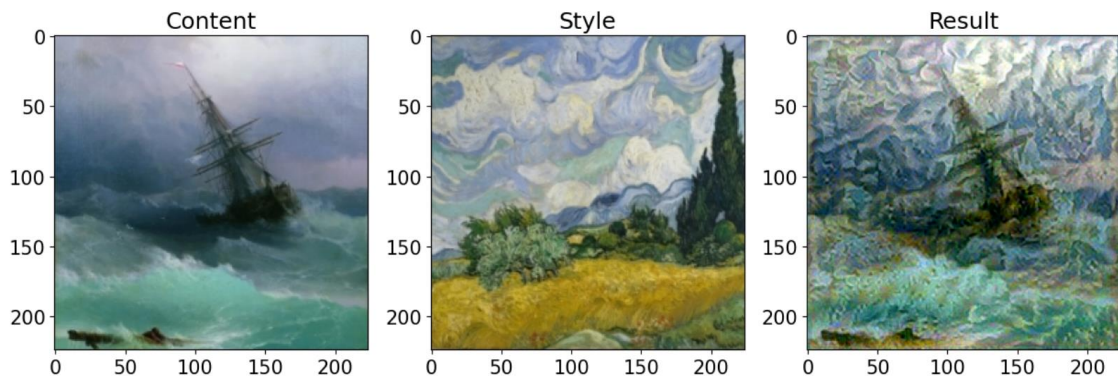


Fig. 9. Stylization result for 600 iterations

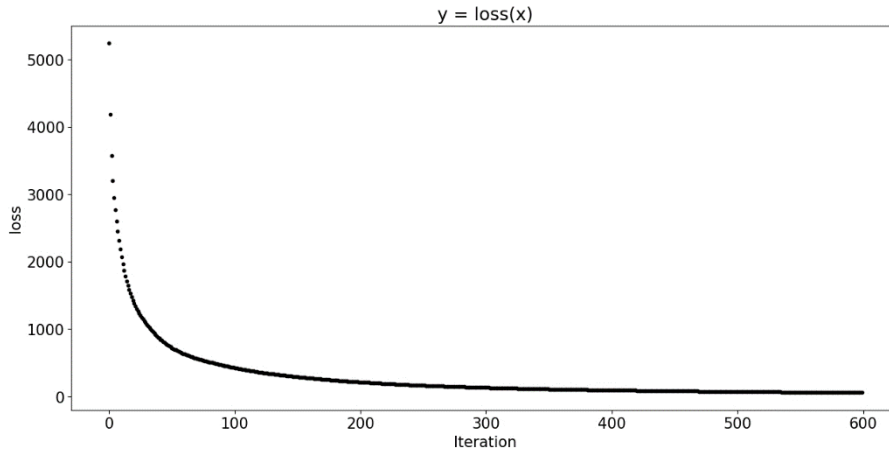


Fig. 10. Scaled loss function plot for 600 iterations

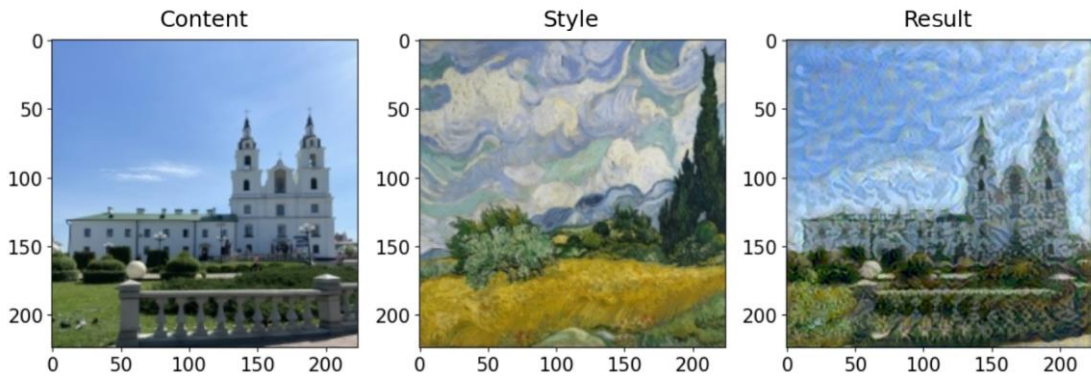


Fig. 11. Stylization result for 200 iterations

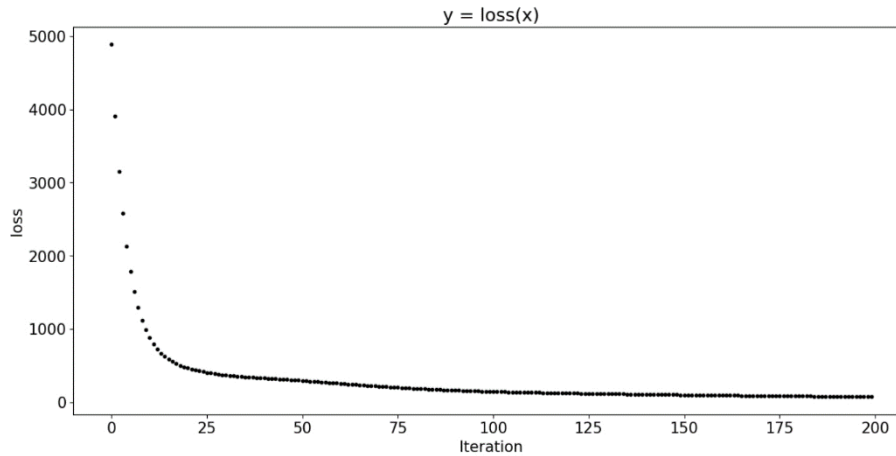


Fig. 12. Loss function plot for 200 iterations

Overall, Model 4 demonstrated a sufficient degree of stability to changes in network parameter settings. As can be seen from the loss function plots (Figs. 10, 12), the model exhibits stable, monotonic convergence as the number of iterations increases.

5. Conclusion

In conclusion, it can be noted that image style transfer using neural networks represents a revolutionary direction in the field of computer vision and artificial intelligence at the intersection of art and technology.

One of the goals of this article is to popularize the use of neural networks for image processing, as using libraries with pre-trained models requires only a moderate level of programming knowledge. Modern environments such as Colab, with the ability to connect to re-

more computing resources, allow computational experiments to be conducted on home computers. Working with the neural network models discussed above can also be implemented, for example, during laboratory work in regular university computer labs.

Comparing models 1, 2, 3, and 4, we can conclude that models 2, 3, and 4, while exploratory in nature and naturally inferior in visual results to model 1, nonetheless provide a comprehensive understanding of the principles of working with convolutional neural networks. In particular, the models under study enable learning the specifics of image processing and network parameter selection. Models 2 and 4 demonstrated a sufficient degree of stability and robustness to changes in network parameter settings. Model 3 demonstrated the least stable results, manifested in frequent blurriness of the resulting images, and required the most experimental tuning of individual settings for optimal style transfer for each loaded image.

Model 1, which is one of the built-in examples of the Tensorflow library and, therefore, is considered professional-grade, yielded the most artistically interesting results. However, unfortunately, this model only provides limited code availability, making it difficult to study the style transfer parameter settings.

Stylish images that produce more artistically successful transfer options are styles associated with broad brushstrokes without fine detail (impressionism, post-impressionism, expressionism, etc.). Works in these styles have fairly free contours, allowing for the transfer of content with a certain degree of distortion of objects without compromising their non-photorealistic quality.

Photorealistic images with a smooth, uniform background perform well as content photographs when conveying the style of paintings. In this case, the background readily accepts the transferred large-texture style and produces an artistically interesting effect on the model's output.

To summarize, the following conclusions can be drawn. Image style transfer creates new boundaries for the interaction between computer technology and art. However, despite the current achievements in the field of style transfer, researchers still face a number of challenges, including improving the quality of image generation and reducing computational costs.

References

1. Park D. Y., Lee K. H. Arbitrary style transfer with style-attentional networks // arXiv preprint, Submitted on 6 Dec 2018 (doi:10.48550/arXiv.1812.02342) (<https://arxiv.org/abs/1812.02342>)
2. Ghiasi G., Lee H., Kudlur M., Dumoulin V., Shlens J. Exploring the structure of a real-time, arbitrary neural artistic stylization network // arXiv preprint, Submitted on 18 May 2017 (doi:10.48550/arXiv.1705.06830) (<https://arxiv.org/abs/1705.06830>)
3. Johnson J., Alahi A., Fei-Fei L. Perceptual losses for real-time style transfer and super-resolution // arXiv preprint, Submitted on 27 Mar 2016 (doi:10.48550/arXiv.1603.08155) (<https://arxiv.org/abs/1603.08155>)
4. Efros A.A., Freeman W.T. Image Quilting for Texture Synthesis and Transfer // SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, 2001, pp. 341 – 346. (doi: 10.1145/383259.383296) (<https://dl.acm.org/doi/10.1145/383259.383296>)
5. Ulyanov D., Vedaldi A., Lempitsky V. Improved Texture Networks: Maximizing Quality and Diversity in Feed-Forward Stylization and Texture Synthesis // 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 4105-4113 (doi:10.1109/CVPR.2017.437) (<https://ieeexplore.ieee.org/document/8099920>)
6. Hertzmann A., Jacobs C.E., Oliver N., Curless B., Salesin D. H. Image Analogies // SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, 2001, pp. 327 – 340 (doi: 10.1145/383259.383295) (<https://dl.acm.org/doi/10.1145/383259.383295>)

7. Sayeed R., Howard T. State of the Art Non-Photorealistic Rendering (NPR) Techniques // EG UK Theory and Practice of Computer Graphics, 2006, pp. 1–10 (doi: 10.2312/LocalChapterEvents/TPCG/TPCGo6/o89-o98) (<https://diglib.eg.org/items/7ba3e4eb-b58d-477a-a8dc-70bfbece3448>) (<https://www.cs.princeton.edu/courses/archive/spring15/cos426/papers/Sayeedo6.pdf>)
8. Gatys L.A., Ecker A.S., Bethge M. A Neural Algorithm of Artistic Style // arXiv preprint, Submitted on 26 Aug 2015 (doi:10.48550/arXiv.1508.06576) (<https://arxiv.org/abs/1508.06576>)
9. ImageNet [Electronic resource]. URL: <https://www.image-net.org/> (accessed 30.11.2025)
10. Jing Y., Yang Y., Feng Z., Ye J., Yu Y., Song M. Neural Style Transfer: A Review // arXiv preprint, Submitted on 11 May 2017 (doi:10.48550/arXiv.1705.04058) (<https://arxiv.org/abs/1705.04058>)
11. Berezin S.A., Volkova V.M. Neurosetevoi perenos proizvol'nogo stilya na portretnye izobrazheniya s ispol'zovaniem neurosetei s mekhanizmom vnimaniya [Neural arbitrary style transfer for portrait images using the attention mechanism] // Sbornik nauchnykh trudov Novosibirskogo gosudarstvennogo tekhnicheskogo universiteta – Transaction of scientific papers of the Novosibirsk state technical university, 2019, №. 3–4 (96), pp. 96–105. (doi: 10.17212/2307-6879-2019-3-4-96-105) [in Russian].
12. Karachev D.K., Shtekhin S.E., Tarasyan V.S., Smolin I.U., Isakov M.V. Style Transfer as a Way to Improve the Generalization Ability of a Neural Network in an Object Detection Task // Proceedings of the Institute for System Programming of the RAS (Proceedings of ISP RAS, 2023, 35(6), pp. 247–264. (doi: 10.15514/ISPRAS-2023-35(6)-16) [in Russian].
13. Fast Style Transfer for Arbitrary Styles [Electronic resource]. URL: https://www.tensorflow.org/hub/tutorials/tf2_arbitrary_image_stylization (accessed 30.11.2025)
14. NeuralStyleTransfer_experiments [Electronic resource]. URL: <https://www.kaggle.com/code/shashank069/neuralstyletransfer-experiments> (accessed 30.11.2025)
15. Ginsburg I. Drawing AI: Uncovering the Secrets of Neural Style Transfer [Electronic resource]. URL: <https://proglab.io/p/ii-dlya-risovaniya-raskryvaem-sekrety-neyronnogo-perenosa-stilya-2022-08-29> (accessed 30.11.2025)
16. Picard. How to Copy Warhol's Style with VGG-19 Neural Networks, Transfer Learning, and TensorFlow [Electronic resource]. URL: <https://habr.com/ru/companies/skillfactory/articles/541758/> (accessed 30.11.2025)
17. Image Style Transfer with Keras and Tensorflow [Electronic resource]. URL: https://propoprops.ru/neural_network/delaem-perenos-stiley-izobrazheniy-s-pomoshchyu-keras-i-tensorflow (accessed 30.11.2025)
18. Neural Style Transfer with Eager Execution. Neural Style Transfer with tf.keras [Electronic resource]. URL: https://colab.research.google.com/github/tensorflow/models/blob/master/research/nst_blogpost/4_Neural_Style_Transfer_with_Eager_Execution.ipynb (accessed 30.11.2025)
19. Colab [Electronic resource]. URL: <https://colab.research.google.com/> (accessed 30.11.2025)
20. Meena G., Mohbey K. K., Indian A., Kumar S. Sentiment Analysis from Images using VGG19 based Transfer Learning Approach // Procedia Computer Science, 2022, Vol.204, pp. 411–418 (doi: 10.1016/j.procs.2022.08.050)
21. Tao Y. Image Style Transfer Based on VGG Neural Network Model // 2022 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA), Dalian, China, 2022, pp. 1475–1482 (doi: 10.1109/AEECA55500.2022.9918891)
22. Mouale M.N.B. Using a Pre-Trained Neural Network (VGG 16) to Solve the Image Style Transfer Problem // Modern Information Technologies and IT-Education, [S.l.], 2022, Vol. 18, №. 2, pp. 241–248 (doi:10.25559/SITITO.18.202202.241-248) [in Russian].