IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Revue d'Intelligence Artificielle
Vol. 33, No. 4, August, 2019, pp. 299-304
Journal homepage: http://iieta.org/journals/ria

Improved Artistic Images Generation Using Transfer Learning

Bulla Premamayudu[1*], Peram Subbarao[1], Koduganti Venkata Rao[2]

[1] Department of Information Technology, VFSR Deemed to be University, Vadlamudi, Guntur 522213, Andhra Pradesh India
[2] Department of Computer Science and Engineering, VIIT, Gajuwaka, Visakhaptnam (Dt) 530049, Andhra Pradesh, India

Corresponding Author Email: drbpm_it@vignan.ac.in

https://doi.org/10.18280/ria.330406

Received: 12 March 2019
Accepted: 5 June 2019

Keywords:
neural style transfer, transfer learning, convolutional neural networks, deep learning, transfer learning

ABSTRACT

The existing methods for photographic image generation have defects in both content preservation and style transform, which suppress the accuracy of the generated images. This paper attempts to improve the quality of photographic images generated based on inputted content images and style images. The transfer learning with VGG-19 model, a convolutional neural network (CNN), was adopted to extract the features from inputted style image and apply them to the content image. Then, a loss function was defined based on the ImageNet model, and used to capture the difference between the images generated based on the content image and the style image. In addition, the VGG-19 model was trained on a very large ImageNet database, aiming to improve its ability to identify image features of any dimension. Finally, several experiments were conducted to compare our method and several existing methods. The results show that the photographic images generated by our image retain the features of inputted content and style images, and minimizes the discrepancy between content and style.

1. INTRODUCTION

Artistic style transfer has introduced a new way of image generation and manipulation [1]. Gatys et al. [2] has proposed a neural style transformation method. This is the biggest achievement in computer vision research area. It has created a different style of image generation with convolutional neural networks. Therefore, many recent works [3-9] have been done with the idea of Gatys et al. [2]. These methods produce reasonable results in the area of transferring photographic features from style painting into the content photographic. Since, distortion is a main problem in artistic style transfer methods [10]. The generated style results have a painting like looking even both content and reference images are photographic. In style transfer methods, the painting look is referred as distortion problem [11]. Luan et al. [12] has identified that the distortions appear only at style transformation process. He proposed a process of photorealism regularization. This process creates on local affine color transformation to reconstruct file content details. Luan et al. [12] as eliminated the unexpected geometric matching problem. He added integrated semantic segmentation masks to Gatys et al's proposal [2]. But the semantic segmentation takes an enormous amount of computational power and a lot of extra time to produce better quality. The distortions can be created at two phases: during content preserving process from content image and geometric matching during the style transformation. This paper improves photorealism by introducing a transfer learning method. This concept measures the corresponding loss function to include retaining content and transforming styles. It minimizes similarity loss and other loss functions discussed in fast neural style method [13].

Our proposed work introduced two stages: reconstruction process and style transfer process. We investigate the real problem of Gatys et al's method. It finds the distortions occurred points at content preservation and style transformation to avoid a lost photorealism of stylized results [14]. It has the capability to improve the photorealism. The loss function using VGG-19, a 19-layer version of the VGG network is used to optimize content details and avoid geometric mismatching. Deeper layers find to detect higher level features like complex textures and object classes [15], [16]. In deeper layer architecture, we have picked a specified hidden layer to use. The input image applied into pretrained picked hidden layer of VGG-19 Network and run forward propagation and backward propagation to finalize the hyperparameters of the model [17].

2. RELATED WORK

Gatys et al. [2] made breaking performance of neural style transfer. This work concentrated on the photographic style transfer. It focuses on the preservation of photorealistic attribute. The results of artistic style are not considered distortion problem. Recently, Luan et al. [12] proposed an improved photographic style transfer method by using semantic segmentation and post- processing. This two-stage method is solved some level of distortion. The post processing step in Luan et al. [12] is replaced with screened poison equation by Mechrez et al. [18] to preserve more precise content details. Liao et al. [3] introduce a method for sophisticated images. This proposal is based on deep features extraction from CNN to find the nearest neighbour field. They improved from the neural style transfer [2] and noted good results than aforementioned method. Inspired by that, correlation between feature maps can effectively describe image texture, and they design various correlations and transform them into style vectors, and investigate

classification performance brought by different variants [19]. In addition to intralayer correlation, interlayer correlation is proposed as well, and its effectiveness is verified. They [20] propose to pre-train the GoogLeNet architecture on ImageNet dataset and fine-tune on our fine-grained fashion dataset based on design attributes.

3. METHOD

Artistic image creation is one of the most fun techniques in the research area of Convolutional Neural Networks (CNN) and Computer Vision. This is one of the applications of CNN in deep neural networks. The main idea of artistic image creation is generating an image(G) from content image (C) and style image(S). In other words, using style image to the photo image and generating photo image with the style of style image.

We used the feature space provided by a standard version of the 19-layer VGG network's 16 convolutional and 5 pooling layers. We normalized the network by scaling weights so that the mean activation over images and locations of each convolutional filter is equal to one. This can be done for the VGG network without changing its performance, as it contains only rectification of linear activation functions and no standardization or pooling of feature maps. At each processing stage of the CNN, a given input image is represented as a series of filtered images. Although the number of different filters increases along the processing hierarchy, some down-sampling method (e.g. max-pooling) reduces the size of the filtered frames, resulting in a decrease in the total number of units per network layer.

3.1 Architecture

Our method uses the trained CNN for features extraction and identification. We have used a network that is trained with large ImageNet database. VGG-19 is a 19-layer version trained model on large ImageNet database. Hence, it is more useful to extract the features of the image. The features of image are extracted at the earlier and deeper layers of the network in our method. The Figure 1 describes the VGG-19 network architecture.

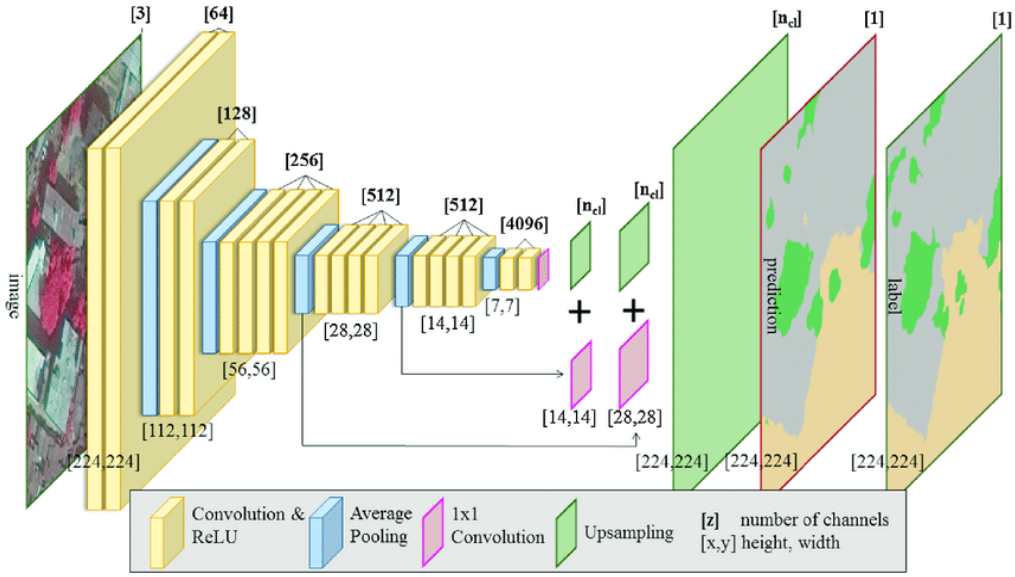

Figure 1. VGG-19 architecture model

Our method consisting three phases:

1. Computing cost function $J_{content}(C,G)$
2. Computing the style cost $J_{style}(S,G)$
3. Defining the total cost to optimize $J(G)=\alpha J_{Content}(C,G)+\beta J_{Style}(S,G)$

3.2 Build the content cost function

Now create forward propagation from the pretrained VGG Network by giving the image C as the input. Similarly, run forward propagation on image G. $a^{(C)}$ and $a^{(G)}$ are the corresponding hidden layer activation. These activation values are diverge based on chosen hidden layer. Hence, the content cost function is:

$$J_{Content}(C,G) = \frac{1}{4 \times n_H \times n_W \times n_C} \sum_{all\ entries} (a^{(c)} - a^{(a)})^2 \quad (1)$$

where, n_H, n_W, n_C are the height, weight and number of channels of chosen hidden layer in the network VGG-19. Figure 2 describe content cost calculation and unroll the activation values.

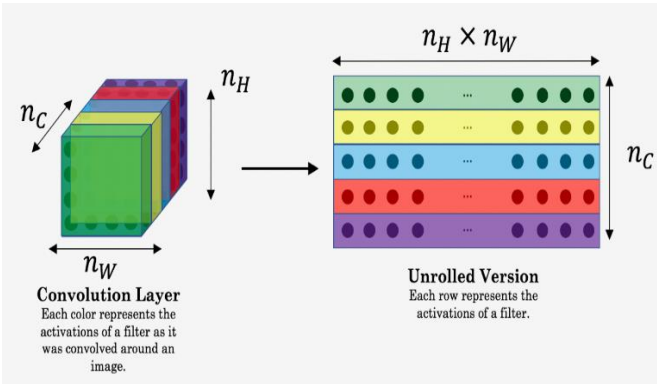

Figure 2. Content cost calculation

It is a good practice to unroll these 3D volumes into a 2D

matrix. This functionality carries out a similar operation for computing the style cost. The main objective of first phase is to measure difference between $a^{(C)}$ and $a^{(G)}$. If we minimize the content cost than G is very much like C.

3.3 Building the style cost

This phase aims at to minizine the distance between Gram matrix of the image S and image G. Gram (GR) matrix is a style matrix. GR is a set of vectors ($V_1, V_2, V_3, \ldots, V_n$). it is the matrix of dot products. The entries of GR matrix are $GR_{ij}=V_i^T V_j$. It means, G_{ij} compares how V_i and V_j are similar. The large value of dot product indicates the highly similar otherwise moderate. The dimension of Gram matrix is (n_c,n_c) where n_c is the number of filters. The diagonal elements of Gram matrix are the important part of this matrix. Suppose, the element G_{ii} indicates how active filter I is. If the i^{th} filter is related to vertical textures in the image than G_{ii} shows how common vertical textures are in image. The large value of G_{ii} means that the image has a lot of vertical texture. Figure 3 describes correlation between content and style filters.

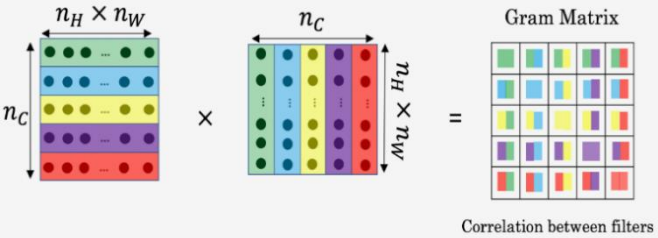

Figure 3. Gram matrix generation

Now, compute the style cost for single hidden layer ($a^{[l]}$). This calculation produces the difference between style matrix (Gram matrix) of image S and image G. The chosen hidden layers style cost is defined as:

$$J_{Style}^{[l]}(S,G) = \frac{1}{4\times n_C^2\times (n_H\times n_W)^2}\sum_{i=1}^{nc}\sum_{j=1}^{nc}(G_{ij}^{(S)} - G_{ij}^{(G)})^2 \quad (2)$$

where, $G_{ij}^{(S)}$ and $G_{ij}^{(G)}$ are Gram matrices of image S and image G. These matrices are computed using hidden layer activation of chosen layer in VGG-19 network. The merge of style costs of several different layers can improve the results of photorealism.

The image style costs of different layer are defined as:

$$J_{style}(S,G) = \sum_l \lambda^{[l]} J_{Style}^{[l]}(S,G) \quad (3)$$

where, $\lambda^{[l]}$ presents different layers.

3.4 Defining total cost function

Finally, define a cost function for style and content cost. This cost function will minimize both style and content cost.

The definition of cost function is as:

$$J(G)=\alpha\, J_{Content}(C,G)+\beta\, J_{Style}(S,G) \quad (4)$$

where, α and β are hyperparameters. The hyperparameters will control the relative weights between content and style images. The total style cost is a linear combination of the content and style cost.

4. IMPLEMENTATION

The implementation details of our methods described in this section. We have selected the pre trained VGG-19 model to extract the textural features from the content and style images. The steps involved in our approach is as follows:

1. Load, reshape and normalize content image.
2. Load, reshape and normalize style image.
3. Initialize the generated image with noise from the content image. This process creates more match between content image and generated image.
4. Load the VGG-19 model
5. Compute the content and style cost by giving content and style image as input to VGG-19 model.
6. Compute total cost
7. Set the optimizer and learning rate hyperparameters.
8. Update the generated image at every step by ruing large number iterations.

Implementation in Tensor Flow Framework:

```
tf.reset_default_graph()
sess = tf.InteractiveSession()
1. Load, reshape and normalize content image:
content_image= cipy.misc.imread("images/louvre_small.jpg")
content_image=reshapeand_normalize_image(content_image)
2. Load, reshape and normalize style image:
style_image = scipy.misc.imread("images/monet.jpg")
style_image = reshape_and_normalize_image(style_image)
3.Initialize the generated image with noise from the content image. This process creates more match between content image and generated image:
generated_image = generate_noise_image(content_image)
imshow(generated_image[0])
4. load the VGG-19 Model:
model = load_vgg_model("pretrained-model/imagenet-vgg-verydeep-19.mat")
5. Compute the content and style cost using VGG-19 model.
sess.run(model['input']. assign(content_image))
out = model['conv4_2']
a_C = sess.run(out)
a_G = out
J_content = compute_content_cost(a_C, a_G)
sess.run(model['input'].assign(style_image))
J_style = compute_style_cost(model, STYLE_LAYERS)
6. Compute total cost:
J = total_cost(J_content, J_style)
7. Set the optimizer and learning rate hyperparameters.
optimizer = tf.train.AdamOptimizer(2.0)
train_step = optimizer.minimize(J)
8. Update the generated image at every step by ruing large number iterations:
def model_nn(sess, input_image, num_iterations = 200):
   sess.run(tf.global_variables_initializer())
   generated_image =
sess.run(model['input'].assign(input_image))
   for i in range(num_iterations):
         sess.run(train_step)
          generated_image = sess.run(model['input'])
          if i%20 == 0:
          Jt, Jc, Js = sess.run([J, J_content, J_style])
          print("Iteration " + str(i) + " :")
          print("total cost = " + str(Jt))
          print("content cost = " + str(Jc))
          print("style cost = " + str(Js))
```

```
        save_image("output/" + str(i) + ".png", generated_image)
    save_image('output/generated_image.jpg',
generated_image)
        return generated_image
```

Output of the Model:

```
model_nn(sess, generated_image)
Iteration 0:
total cost = 5.04752e+09
content cost = 7865.72
style cost = 1.26186e+08
Iteration 20:
total cost = 9.44829e+08
content cost = 15237.8
style cost = 2.36169e+07
Iteration 40:
total cost = 4.80348e+08
content cost = 16714.2
style cost = 1.20045e+07
Iteration 60:
total cost = 3.10248e+08
content cost = 17445.0
style cost = 7.75183e+06
```

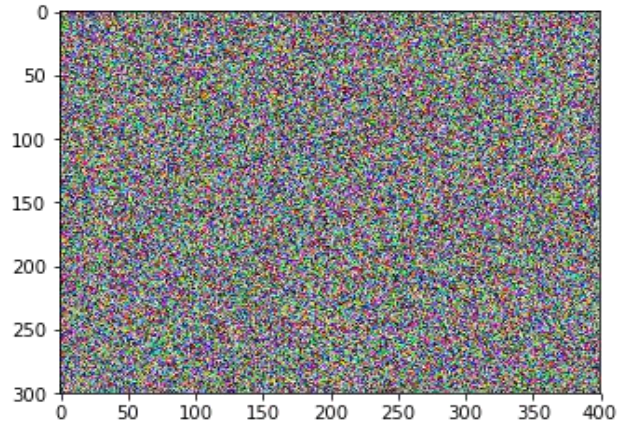

Figure 4. Initial generated image with random noise in tune with content image

Figure 5. Artistic image generation from content and style image

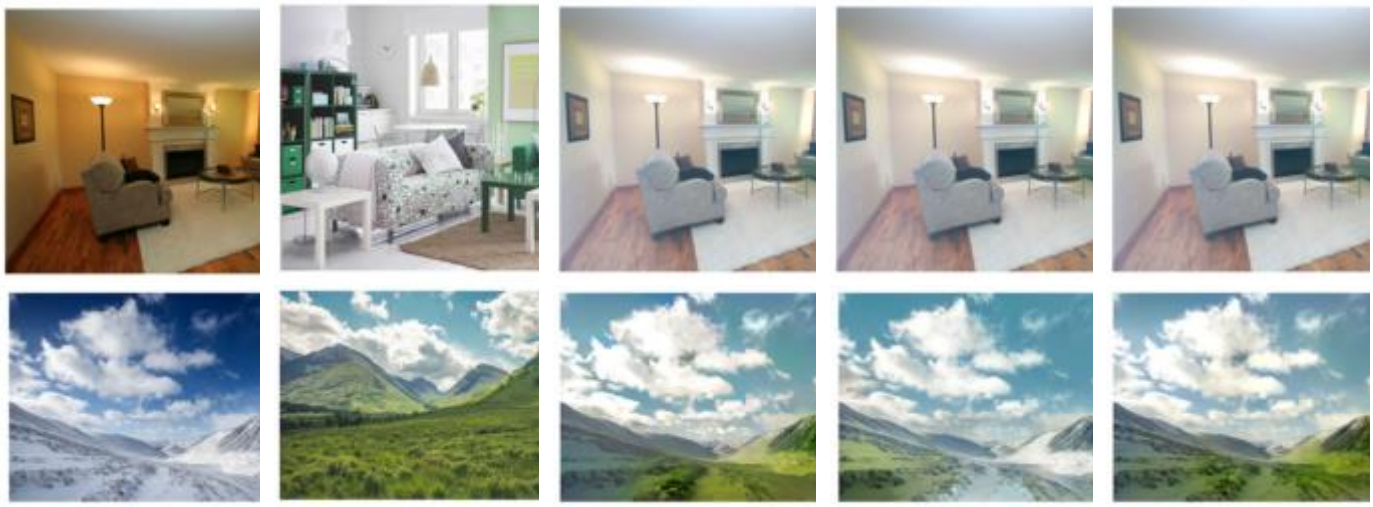

Figure 6. Comparison between State-of-the-art Methods and Our Method

Figure 4 shows the generation of initial content image. As a first step, create the generated image as a noisy image from the content image. By initializing the pixels of the generated image to be mostly noise but still slightly correlated with the content image, this will help the content of the generated image more rapidly match the content of the content image.

Figure 5 shows example of stylizing photo by artwork. To get the best-looking results, running the optimization algorithm longer (and perhaps with a smaller learning rate) might work better result. In results, the style cost of the image improved for every iteration. It shows that, if we increase the number of iterations then the quality of the generated image can be improved. By initializing the random parameters and applying weight decay methods there is a possibility to get the better results.

The goal of style transfer is to generate an image combining the content of an image of the target content with the style image. We demonstrate that our style transfer network allows users to monitor the degree of stylization, interpolate between different styles, pass styles while retaining colors, and use different styles in different spatial regions to further illustrate the versatility of our process. Remember that all of these controls are only implemented with the same network at runtime, without any change to the training model. Training a new network for each painting is inefficient as painting styles share common visual textures, color palettes and semantics to encode an image's scene. It would provide a rich vocabulary to represent any painting by building a style transfer network that shares its representation across many paintings. A simple trick recognized is to build a network of style transfer as a standard encoder / decoder architecture but to customize the parameters of normalization unique to each style of painting.

5. RESULTS

The comparison between Gats et al. [2], Luan et al. [12], Liao et al. [7], and our method presents in this section. Figure 6 shows comparison between existing methods which are presented in related work section and our method. Our technique reproduces almost everything about the picture of the material and accurately moves the type of shading. Our technique shows our strong ability to reduce distortions and reserve image structures of content. Figure 6 provides the comparison with our method and the existing neural style transfer methods. We demonstrated our method in tensor flow framework to show the style loss function computation.

It is clear from these results that the trained transfer style network is aware of the semantic content of images. The people are clearly identifiable in the transformed picture, for example, in the beach images in Figure 6, but the landscape is distorted beyond recognition. Another reason is that the VGG-16 loss network has selective features for humans and animals as these items are present in the classification dataset it was trained on. Our style transfer networks are equipped to conserve the features of VGG-16, and in doing so they learn to preserve more than background artifacts for people and animals.

6. CONCLUSIONS

The main drawback in the existing methods presented in this paper's related work section is that both stages of content preservation and style transformation distort images in order to lose photorealism. This paper examined the reasons for distortion in photographic image. We therefore used the VGG-19 model, which is trained for similarity loss function in the large ImageNet database. In the output image the exact function information and object content structures are covered. Our method is to protect the image output photorealism. It also prevents the discrepancy between images of content and style.

There are still some interesting questions to study further. For example, the auto-encoder can incorporate semantic segmentation as additional oversight in the region's decomposition, helping to create more impressive region-specific transfer. Therefore, our learned representation does not make full use of all image channels, which may require a more compact representation.

REFERENCES

[1] Li, Y., Zhang, T., Han, X., Qi, Y. (2018). Image style transfer in deep learning networks. 2018 5th Int. Conf. Syst. Informatics, pp. 660-664.

https://doi.org/10.1109/ICSAI.2018.8599501
[2] Gatys, L.A., Ecker, A.S., Bethge, M. (2016). Image style transfer using convolutional neural networks. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 2414-2423. https://doi.org/10.1109/CVPR.2016.265
[3] Chen, D., Yuan, L., Liao, J., Yu, N., Hua, G. (2017). StyleBank: An explicit representation for neural image style transfer. Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, pp. 2770-2779. https://arxiv.org/abs/1703.09210
[4] Chen, T.Q., Schmidt, M. (2016). Fast patch-based style transfer of arbitrary style. https://arxiv.org/abs/1612.04337
[5] Ghiasi, G., Lee, H., Kudlur, M., Dumoulin, V., Shlens, J. (2017). Exploring the structure of a real-time, arbitrary neural artistic stylization network. pp. 1-27. https://arxiv.org/abs/1705.06830
[6] Huang, X., Belongie, S. (2017). Arbitrary style transfer in real-time with adaptive instance normalization. Proc. IEEE Int. Conf. Comput. Vis., pp. 1510-1519. https://arxiv.org/abs/1703.06868
[7] Li, C., Wand, M. (2016). Combining markov random fields and convolutional neural networks for image synthesis. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 2479-2486. https://arxiv.org/abs/1601.04589
[8] Wang, X., Oxholm, G., Zhang, D., Wang, Y.F. (2017). Multimodal transfer: A hierarchical deep convolutional neural network for fast artistic style transfer. Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, pp. 7178-7186. https://arxiv.org/abs/1612.01895
[9] Zhang, H., Dana, K. (2019). Multi-style generative network for real-time transfer. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), pp. 349-365. https://arxiv.org/abs/1703.06953
[10] Georgievski, B. (2019). Image augmentation with neural style transfer. Big Data Processing and Mining. ICT Innovations 2019. Communications in Computer and Information Science, 1110: 212-224. https://doi.org/10.1007/978-3-030-33110-8_18
[11] Min, K.Y. (2019). Feature visualization in comic artist classification using deep neural networks. J. Big Data, 6: 59. https://doi.org/10.1186/s40537-019-0222-3
[12] Luan, F., Paris, S., Shechtman, E., Bala, K. (2017). Deep photo style transfer. Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, pp. 6997-7005. https://arxiv.org/abs/1703.07511
[13] Johnson, J., Alahi, A., Li, F.F. (20160. Perceptual losses for real-time style transfer and super-resolution. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 9906 LNCS: 694-711. https://arxiv.org/abs/1603.08155
[14] Wang, W., Shen, W. (2018). Image artistic style migration based on convolutional neural network. 2018 5th Int. Conf. Syst. Informatics, pp. 967-972. https://ieeexplore.ieee.org/document/8599512, accessed on January 12, 2019.
[15] Cui, W. (2018). Image style transfer with multi-target loss for loT applications. 2018 15th Int. Symp. Pervasive Syst. Algorithms Networks, pp. 296-299. https://ieeexplore.ieee.org/document/8636295, accessed on January 12, 2019.
[16] Kim, K.S., Kim, D., Kim, J., Member, S. (2019). Hardness on style transfer deep learning for rococo painting masterpieces. 2019 Int. Conf. Artif. Intell. Inf. Commun., pp. 452-454. https://arxiv.org/pdf/1804.03189.pdf
[17] Zhao, H., Rosin, P.L., Kai, Y.K., Lin, M.G., Liu, Q.Y. (2019). Image neural style transfer with global and local optimization fusion. IEEE Access, 7: 85573-85580. https://doi.org/10.1109/ACCESS.2019.2922554
[18] Mechrez, R., Shechtman, E., Zelnik-Manor, L. (2017). Photorealistic style transfer with screened poisson equation. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8736245, accessed on January 12, 2019.
[19] Chu, W.T., Wu, Y.L. (2018). Image style classification based on learnt deep correlation features. IEEE Trans. Multimed., 20(9): 2491-2502. https://doi.org/10.1109/TMM.2018.2801718
[20] Seo, Y., Shin, K.S. (2018). Image classification of fine-grained fashion image based on style using pre-trained convolutional neural network. 2018 IEEE 3rd Int. Conf. Big Data Anal., pp. 387-390. https://doi.org/10.1109/ICBDA.2018.8367713