Multi-Style Generative Network with Transfer Learning

Sanjay Reddy Banda — Suprith Reddy Gurudu — Chandra Sekhar Devarakonda

sbanda@ufl.edu — sgurudu@ufl.edu — sdevrakonda@ufl.edu
Dept. of CISE, University of Florida
December 10, 2020

Abstract

Although there is a quick development in the style transfer, current methodologies for multi-style or arbitrary style transfer are using feed-forward network that are commonly produces low quality and less flexible images. It is profoundly tricky to accomplish style transfer using one dimensional representation. To overcome this challenge, we used a Co-Match layer which matches the second order statistics with style images. We also built a Multi-style Generative Network (MSG-Net) which improves the performance and an up-sampled convolution that avoids checkerboard artifacts created by fractional stride. This MSG-Net for style transfer is well-suited for most of the applications such as color preservation, spatial and brush stroke size control, and content-style interpolation [1]. We have used Torch, Torchvision, PIL, and Numpy frameworks for our implementation.

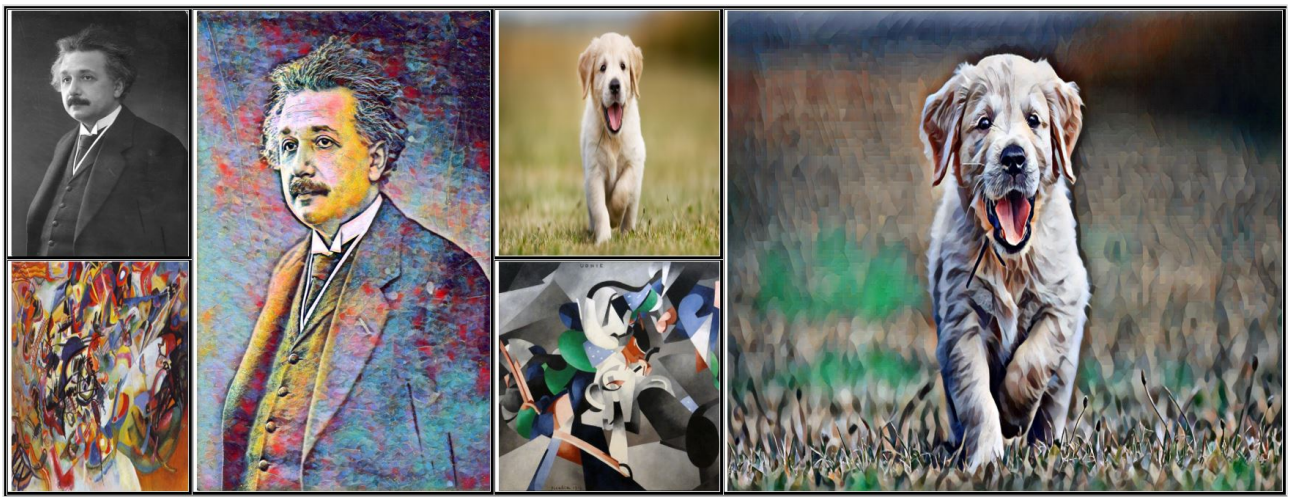


Figure 1: Examples of stylized images and the corresponding content and styles using the MSG-Net.

1 Introduction

Multi-Style Generative transfer network can be represented as texture synthesis based on the semantics of the content image [2]. Most of the primary works involve traditional synthesizing techniques for texture that includes pixel re-sampling [3] and/or multi-scale matching of feature statistics [4]. Image styles can be captured well using Gram Matrix of Convolutional Neural Networks (CNN) according to *Gatys*[5]. There is a growing demand for style transfer and texture synthesis using feed-forward networks[6] and other optimization techniques [5]. In extension, multi-style or arbitrary style transfer can be achieved [7]. Although, representation of an image style to one-dimensional space is complicated, these techniques transform the style images to 1D embedding by feature-map (i.e., mean and variance of various styles) tuning.

Generally, input images with high resolution do not produce satisfying results and there is no specific method to adjust the size of the brush strokes. A fully convolutional neural network (FCN) will take all sizes of input image and the brush stroke size changes with the style image resize. Hence, we need a control mechanism for brush size at run-time. Additionally, there is a limitation of finer performance for style representations in one-dimensional embedding, so a two-dimensional model representation for style images is desirable. To achieve this, we introduced a layer called Co-Match that sets the image style with a two-dimensional representation along with matching Gram

Matrix aka second-order feature statistics of the image styles while training the model. This co-match layer facilitates generation of multi-style images on a single feed-forward network.

We also built a Multi-style Generative Network or MSG-Net along with the co-match layer and conventional CNN with an up-sampling layer. This network generally runs after training acting as a feed-forward layer and includes a decoder to retrieve the details of a down-sampled image. The up-sampled layer eschews any checkerboard artifacts because of using integer stride instead of a fractional stride [8] during learning. We used an up-sampling residual block to decrease the complexity of computation preserving the flexibility of styles through a greater number of channels.

2 Background Work

Pyramid Matching.

In conventional methods, multi-scale image pyramids [4] were used for synthesizing texture where the white noise images were manipulated and matched with features statistics at each level of pyramid of target image to generate texture images which are realistic. Our strategy replicates the matching method of feature statistics in the feed-forward neural network and utilizes the networks of deep learning that reduces the cost of computations during training.

Fusion Layers.

Contemporary work related to fusion layers contains concatenation of feature maps and element-wise summation [9]. As these methods are not easily applicable because we cannot separate style from content and output images should neither contain any semantic data of the style image nor style data of the content images for the style transfer. So, we used a Co-Match layer in place of fusion layer which accepts two inputs, content image and style image.

Multiple or Arbitrary Style Transfer.

In a lot of current works of multiple or arbitrary style transfer [10], a swap layer is used which is slow and produces low quality images in comparison to feed-forward neural network methods. Instead, we used an adaptive instance normalization that matches mean and variance and a co-match layer that matches the second order statistics of Gram Matrices of the feature maps.

Generative Adversarial Network.

The Generative Adversarial Network(GAN) [11], which comprises of an adversarial generator and discriminator, both are trained to check each other's work, has led to the surge of interest in study of image generation. Most of the recent works in the image-to-image GANs adopt conditional GAN so as to approach the solution with ground truth image pairs. But when it comes to style transfer problem, there is no particular gold standard to measure the performance of generator, in other words, there is no defined loss function for validation. Therefore, we rely on the work to adopt a loss network that minimize the perceptual variation between the synthesized images with style and content images. The loss network also provides the supervision of the adversarial network learning. The use of Gram Matrix to trigger style synthesis is inspired by a recent work that suggests using an encoder in GAN framework.

3 Style and Content Representation

Models pre-trained on large datasets can be useful for Style transfer as they tend to identify both the semantic and perceptual features in the Images. They provide explicit representations of both content and style independently. Each activation of the network represents a feature map of the i^{th} scale $\mathcal{F}^i(x) \in R^{C_i \times H_i \times W_i}$ with a given input image X, where C_i, H_i and W_i are number of feature map channels, height and width respectively. The texture or style of the image can be extracted by computing the gram matrix which serves as a representation of distribution of features. Gram matrix is equivalent to covariance matrix for zero-centered data. Gram matrix can be calculated by first reshaping the feature map $\mathcal{F}^i(x) \in R^{C_i \times H_i \times W_i}$, where $\Phi()$ is a reshaping operation. Then the Gram Matrix can be written as

$$\mathcal{G}(\mathcal{F}^i(x)) = \sum_{h=1}^{H^i} \sum_{w=1}^{W^i} \mathcal{F}^i_{h,w}(x) \mathcal{F}^i_{h,w}(x)^T \qquad (1)$$

can also be represented as

$$\mathcal{G}(\mathcal{F}^i(x)) = \Phi(\mathcal{F}^i(x)) \Phi(\mathcal{F}^i(x))^T. \qquad (2)$$

4 Co-Match Layer

This layer is specifically made to extract the second order feature statistics from the given style images. Xc represents content target and Xs represent style target. For a given i-th scale in the descriptive network, content and style can be represented as Fi(xc) and G(Fi(xs)) respectively. Yˆi is the desired output which preserves the target style feature statistics and matches the semantic content of the input image.

$\hat{Y}^i = argmin ||Y^i - \mathcal{F}^i(x_c)||^2 + \alpha ||\mathcal{G}(Y^i) - \mathcal{F}^i(x_s)||^2_F$

where is a trade-off parameter that balancing the contribution of the content and style targets. The optimization of the above problem can be tackled with an iterative approach, however it is highly infeasible in real-time or making the model differentiable. We, therefore, approximate the solution by placing the computational burden on training stage. Introducing approximation that can tune the feature map based on

the target style.

$$\hat{y}^i = \Phi^{-1}[\Phi(\mathcal{F}^i(x_c))^T W \mathcal{G}(\mathcal{F}^i(x_s))]^T, (3)$$

where $W \in R^{C_i \times C_i}$ is a learnable weight matrix and $\Phi()$ is a reshaping operation to match the dimension, so that $\Phi(\mathcal{F}^i(x_c)) \in R^{C_i \times (H_i W_i)}$ To gain an intuition on the functionality of W, we suppose $W = \mathcal{G}(\mathcal{F}^i(x_s))^{-1}$then the first term in Equation 2 (content term) is minimized. Now let $W = \Phi(\mathcal{F}^i(x_c)^{-T} \mathcal{L}(\mathcal{F}^i(x_s))^{-1}$ obtained by the Cholesky Decomposition of $\mathcal{G}(\mathcal{F}^i(x_s)) = \mathcal{L}(\mathcal{F}^i(x_s))\mathcal{L}(\mathcal{F}^i(x_s))^T$the second term of Equation 2 (style term) is minimized. W learns directly from the loss function to balance the trade-off dynamically. The co-match layer is differentiable and can be placed in the generative network. Therefore it can be directly learned from the loss function without any need for additional super vision.

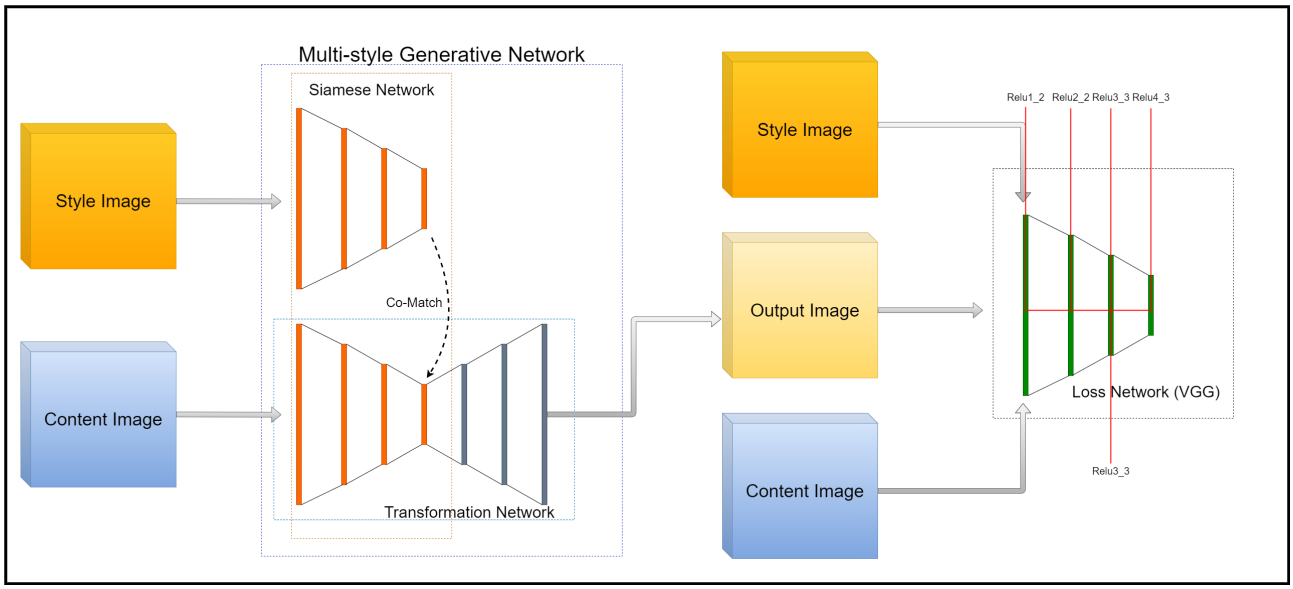


Figure 2: MSG-Net Architecture

5 MSG-Network

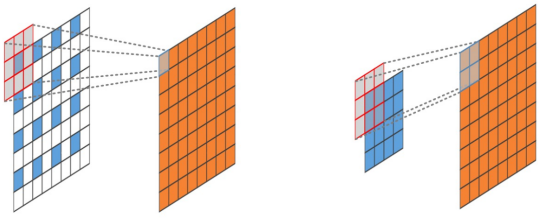
5.1 Architecture

Architecture

In traditional methods, a single-style transfer accepts only one input – content image and produces the output image by employing a loss function to learn the feature statistics of the image style as $G(x_c)$. In this paper, we use an MSG-Net that takes two input parameters, content image and style image as x_c and x_s. We have also used a Siamese network that includes an encoder which shares the weights of the transformation network with different scales of feature statistics i.e., x_s. It outputs the Gram Matrices $\{\mathcal{G}(\mathcal{F}^i(x_s))\}(i = 1.., K)$. Now the content image x_c is passed to the transformation network that co-matches the feature statistics of image style of different scales.

Up-sampled Convolution

To preserve semantic coherence of content images, we need a larger receptive field and computationally heavy operations for switching styles. So, standard CNN uses encoder and decoder framework for increasing efficiency. We used integer strides for ignoring checkerboard artifacts. For instance, with a factor for upsampling as 2, we will produce 2x2 outputs for each window as shown in the below figure.



Right: Up-sampled convolution (Integer Stride). Left: fractionally-strided convolution.

Up-sample Residual Block

Residual blocks are useful when there is need for low computational complexity with no lose of diversity. It preserves a great number of feature map channels that helps in decreasing the complexity. It also utilizes reflective padding in order to remove artifacts in a multi-style generative process. This architecture extends deeper and converges faster by allowing an identity to pass through the network.

5.2 Learning

As there is no gold-standard in style transfer problem, we would try to balance the weighted combination of content and style differences of the generator network

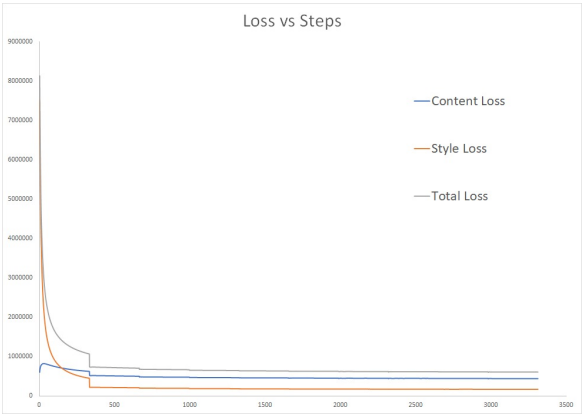
outputs and the targets of VGG (pretrained) loss network. Let $G(x_c, x_s)$ denotes the generator network parameterized by weights W_G. Learning starts by first sampling content images $x_c X_c$ and style images $x_s X_s$ and then reweighting the parameters W_G of the generator $G(x_c, x_s)$ in order to minimize the loss

$$\hat{W}_G = argminE_{x_c,x_s}\{\lambda_c||\mathcal{F}^c(G(x_c,x_s)) - \mathcal{F}^c(x_c)||_F^2$$

$$+\lambda_s \sum_{i=1}^{K} ||\mathcal{G}(\mathcal{F}^i(G(x_c,x_s))) - \mathcal{G}(\mathcal{F}^i(x_s))||_F^2$$

$$+\lambda_T V l_T V(G(x_c,x_s))\}$$

where c and s are for balancing weights appropriately for content and style losses. We consider image content at scale c and image style at scales $i \in \{1, ...K\}$. $l_{TV}()$ is the total variation regularization as used prior work for encouraging the smoothness of the generated images [6].



Plot for Mean Squared Error Loss over batches during training MSG-Net on COCO Dataset[12]

5.3 Pseudo Code

Algorithm 1: Training of MSG-Net

Result: Trained GAN Model

1. Load model weights W_G;

for *number of training epochs* **do**

1. Load target style image x_s;
2. Normalize according to network specifications;

$$3.\mathcal{G}(\mathcal{F}^i(x_s)) = \sum_{h=1}^{H^i}\sum_{w=1}^{W^i} \mathcal{F}^i_{h,w}(x_s)\mathcal{F}^i_{h,w}(x_s)^T$$

for *number of batches* **do**

1. Load batch content images x_c and normalize;

$$2.\hat{Y}^i = argmin||Y^i - \mathcal{F}^i(x_c)||^2 + \alpha||\mathcal{G}(Y^i) - \mathcal{F}^i(x_s)||_F^2$$

$$3.\hat{W}_G = argminE_{x_c,x_s}\{\lambda_c||\mathcal{F}^c(G(x_c,x_s)) - \mathcal{F}^c(x_c)||_F^2$$

$$+\lambda_s \sum_{i=1}^{K} ||\mathcal{G}(\mathcal{F}^i(G(x_c,x_s))) - \mathcal{G}(\mathcal{F}^i(x_s))||_F^2$$

$$+\lambda_T V l_T V(G(x_c,x_s))\}$$

4. compute MSE losses;

end

end

Algorithm 2: Generation of Stylized Images

Result: Stylized Image of the given content image and style target

1. Load model weights W_G;
2. Transform and normalize given content image input;
3. Input the normalized content image to the MSG-Net;

6 Experimental Results

Figure 3: Example I - shows the difference between the stylized images generated by the MSG-Net during training with different number of epochs. Left most image displays the content. Top row includes style targets. Second row consists of stylized images of obtained on the model with one epoch. Similarly, the last row has the stylized images after ten epochs.

Figure 4: Example II - following the example I, we can infer that style transfer can be obtained after one epoch, but the granularity and quality of the stylized image is directly proportional to the number of epochs. If we train the model with more number of epochs, we get better results.

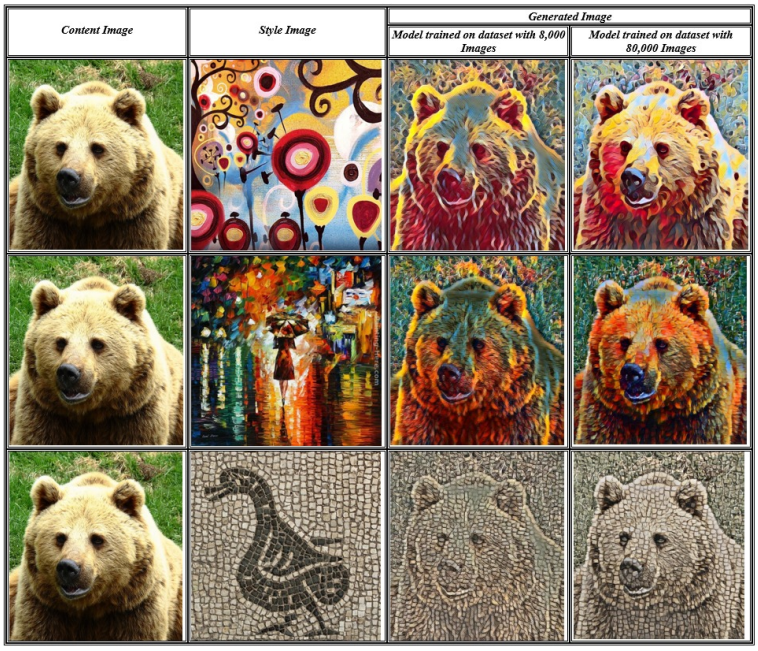


Figure 5: Example III - Left most column contains the content images. Second left column consists of the style targets. Third column has the generated stylized images of MSG-Net trained on dataset of 8K images from COCO dataset [12]. Right most column shows the generated stylized images of MSG-Net trained on dataset of 80K images from COCO dataset [12].

7 Conclusion

We used a modern Co-Match layer which learns the second order statistics or Gram Matrix as style target embedding. We have utilized this technique to style transfer where we attain comparable performance and extremely improved speed compared to prevailing approaches. The MSG-Net has accomplished better results compared to conventional strategies. Moreover, this MSG-Net for style transfer is congruent for most of the applications such as color preservation, spatial and brush stroke size control, and content-style interpolation. We conclude that adding co-match layer to the MSG-Net greatly improves the style transfer.

8 Acknowledgement

This work was supported by Dr. Corey Toler-Franklin, Assistant Professor, CISE Dept., University of Florida. A GPU cluster used for this research was provided by the HiperGator, managed by University of Florida.

References

[1] Multi-style Generative Network for Real-time Transfer, Hang Zhang, Kristin Dana. [arXiv:1703.06953].
[2] C. Li and M. Wand. Combining markov random fields and convolutional neural networks for image synthesis. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.
[3] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pages 341–346. ACM, 2001.
[4] J. S. De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In Proceedings of the 24th annual conference on Computer graphics and interactive techniques, pages 361–368. ACM Press/AddisonWesley Publishing Co., 1997.
[5] L. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In Advances in Neural Information Processing Systems, pages 262–270, 2015
[6] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In European Conference on Computer Vision, 2016.
[7] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua. Stylebank: An explicit representation for neural image style transfer. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
[8] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3431–3440, 2015.
[9] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1933–1941, 2016.
[10] T. Q. Chen and M. Schmidt. Fast patch-based style transfer of arbitrary style. arXiv preprint arXiv:1612.04337, 2016.
[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.
[12] Microsoft COCO: Common Objects in Context. http://msvocds.blob.core.windows.net/coco2014

9 How to run the code

Dataset

Run the following commands :

```
$ cd dataset
$ wget http://msvocds.blob.core.windows.net/coco2014/train2014.zip
$ unzip train2014.zip
```

Training the model

Run the following command :

```
$ python main.py train
  --dataset <dataset_dir>
  --style-image <target_styles_image_dir>
  --save-model-dir <final_model_save_dir>
  --checkpoint-model-dir <checkpoint_model_save_dir>
  --epochs <#epochs>
  --cuda <1 if gpu available else 0>
  --batch-size <batch_size>
```

Generation of Stylized Images

Run the following command :

```
$ python main.py eval
  --content-image <content_image_path>
  --model <model_path>
  --output-image <output_image_path>
  --cuda <1 if gpu available else 0>
  --style-num 19
  --style-id <style_image_id>
```