CS 461/661 Final Report
Style Transfer with Spatial and Strength Control

Hsi-Wei Hsieh (hhsieh9), Cong Mu (cmu2), Yicheng Hu (yhu70)
Minmin Hu (mhu15), Haoran Li (hli143)

I. Project goals

A. Problem Statement and Initial Goals

Style transfer refers to an active research problem which develops algorithms to transfer the style of one image (style image) onto another one (content image). Our goal for this project is to to build a pipeline for fast style transfer with spatial and strength control. We expect our pipeline could allow changing styles and strength simultaneously and could also do spatial control guided by masks generated via semantic image segmentation techniques [1].

B. Goals Achieved

We ended up with a pipeline (trained networks) which transfers two distinct styles onto two regions of the input content image guided by masks. The input content image, style strength and the masks can be changed for real time but the two styles are fixed.

II. Approaches

There are a lot of ways that can address style transfer problem, but many of them are inspired by the framework of Gatys et al [2], which relies on the features captured by pre-trained VGG net. One advantage of Gatys approach is that it is straight forward to conduct spatial control by manipulating the features correspond to specific regions of content images. Therefore, we choose to explore several approaches based on the idea of [2] (including the Gatys' original framework) to achieve our goals. All approaches we explored were implemented in **PyTorch, Numpy** and **Scipy**.

A. Gatys Optimization Approach [2]

Let us denote by $\mathbf{F}_\ell(\mathbf{x}) \in \mathbb{R}^{(H_\ell(\mathbf{x})\times W_\ell(\mathbf{x}))\times N_\ell}$ the feature representation of an image $\mathbf{x}$ in layer ℓ of a pre-trained VGG 16 network and $\mathbf{G}_\ell(\mathbf{x}) \doteq \frac{\mathbf{F}_\ell(\mathbf{x})^T\mathbf{F}_\ell(\mathbf{x})}{H_\ell(\mathbf{x})W_\ell(\mathbf{x})}$ the Gram matrix of feature map at layer ℓ, where N_ℓ, $H_\ell(\mathbf{x})$ and $W_\ell(\mathbf{x})$ are the number of feature maps, height and width of each feature map at layer ℓ. It was showned that Gram matrices can capture the style of an image. The idea is to generate an new image $\hat{\mathbf{x}}$, given content image $\mathbf{x}_C$ and style image $\mathbf{x}_S$ which minimizes the following loss function:

$$\mathcal{L}(\mathbf{x}) = \alpha\mathcal{L}_{content}(\mathbf{x}) + \beta\mathcal{L}_{style}(\mathbf{x}),$$

where the first term compares features at a single layer ℓ_C through Frobenius norm:

$$\mathcal{L}_{content}(\mathbf{x}) \doteq \frac{\|\mathbf{F}_{\ell_C}(\mathbf{x}) - \mathbf{F}_{\ell_C}(\mathbf{x}_C)\|_F^2}{(H_\ell(\mathbf{x}_C)\times W_\ell(\mathbf{x}_C))\times N_\ell}$$

and the second term compares Gram matrices:

$$\mathcal{L}_{style}(\mathbf{x}) \doteq \sum_\ell \frac{w_\ell}{N_\ell^2}\|\mathbf{G}_\ell(\mathbf{x}) - \mathbf{G}_\ell(\mathbf{x}_S)\|_F^2.$$

. In this project, we use pre-trained VGG-16 network and include "`relu3_3`" as layer ℓ_C for content loss and "`relu1_2`", "`relu2_2`", "`relu3_3`", "`relu4_3`" for style comparison. We adopt the *L-BFGS* algorithm implemented in **Scipy** to solve the the optimization problem.

B. Spatial Control via Guidance Masks [3]

Suppose we want to transfer R distinct styles to R disjoint regions onto the content image. For each region, we generate a mask $\mathbf{T}^r$, which is a binary image indicating the region of interest in content image. We then propagate $\mathbf{T}^r$ to each layer ℓ of VGG net to obtain guiding channels $\mathbf{T}_\ell^r$

by indicating spatial region of feature maps only on the neurons whose receptive field is contained in the guidance region in the content image. We normalize $\mathbf{T}_\ell^r$ so that $\|\mathbf{T}_\ell^r\|_F = 1$.

We define the spatially guided feature map and guided Gram matrix as

$$\mathbf{F}_\ell^r(\mathbf{x})_{[:,i]} = \mathbf{T}_\ell^r \circ \mathbf{F}_\ell(\mathbf{x})_{[:,i]}$$
$$\mathbf{G}_\ell^r(\mathbf{x}) \doteq \mathbf{F}_\ell^r(\mathbf{x})^T \mathbf{F}_\ell^r(\mathbf{x}),$$

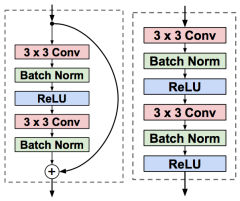
where $\mathbf{F}_\ell^r(\mathbf{x})_{[:,i]}$ is the vectorized i-th feature map of layer ℓ and $\circ$ denote the entry-wise multiplication. Then by comparing the guided Gram matrix with the one from corresponding style, we can conduct spatial control in the setting of section II-A.

C. Fast Neural Style Transfer (FNST) [4]

The approach in [2] is computationally expensive since it requires solving optimization problem for each pair of content and style image. To achieve the real-time style transfer, we follow the idea in [4], which utilized the loss function introduced in section II-A to train a image transformation network to solve the optimization problem proposed in [2] at test time. In particular, figure 1 shows the architecture of image transform network

Layer	Activation size
Input	3 × 256 × 256
Reflection Padding (40 × 40)	3 × 336 × 336
32 × 9 × 9 conv, stride 1	32 × 336 × 336
64 × 3 × 3 conv, stride 2	64 × 168 × 168
128 × 3 × 3 conv, stride 2	128 × 84 × 84
Residual block, 128 filters	128 × 80 × 80
Residual block, 128 filters	128 × 76 × 76
Residual block, 128 filters	128 × 72 × 72
Residual block, 128 filters	128 × 68 × 68
Residual block, 128 filters	128 × 64 × 64
64 × 3 × 3 conv, stride 1/2	64 × 128 × 128
32 × 3 × 3 conv, stride 1/2	32 × 256 × 256
3 × 9 × 9 conv, stride 1	3 × 256 × 256

(a) Architecture



(b) Res & Conv block

Fig. 1: Image transformation network architecture adopted from [4].

As shown in figure 2, the spatial control techniques in [3] can be easily adopted to the fast neural style transfer framework. Specifically, the transformation net now takes content image along with the fixed guiding masks as input. During the training, the masks are also passed to loss network as mentioned in section II-B. Once such net is trained, any content image and guiding masks can be input to obtain styled image for fixed styles on guided regions to achieve the real-time spatial control.

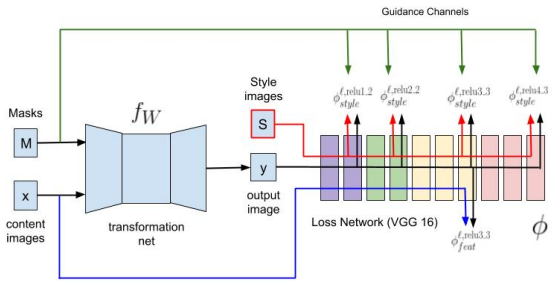


Fig. 2: Fast neural style Transfer with spatial control.

D. Fast Neural Style Transfer with Strength Control (FNSTSC) [5]

Furthermore, the strength control modifications introduced in [5] can be adopted to this framework. In particular, for the basic framework as in [4], we need to re-train the transform net each time if we want to adjust the strength by using different α and β in the total loss. By modifying the residual blocks in the basic framework, we can achieve real-time strength control. Specifically, the output of residual block $i \in \{1, \cdots, 5\}$ given input u is now given by

$$g_i(u) = u + \gamma_i f_i(u) \text{ with } \gamma_i = 2\frac{|\alpha\beta_i|}{1 + |\alpha\beta_i|}.$$

Note that here β_i is a trainable parameter of block i. Re-normalization is also applied to ensure that the non-linear factor γ_i stays within a reasonable range $[0, 2)$ for any stylization strength α. Intuitively, the impact of the non-linear transformation $f_i(u)$ will increase as γ_i increases. Thus α can control the amount of style added to the generated image via γ_i. During training, strength α is sampled uniformly from the grid $[0, 0.1, \cdots, 10.0]$ so that the transform network can learn to generate images with different stylization strengths. At test time, we can achieve real-time strength control by using different strength factor as input.

With our discussion in previous sections, we now have the fast neural style transfer framework with spatial and strength control as illustrated in figure 3.

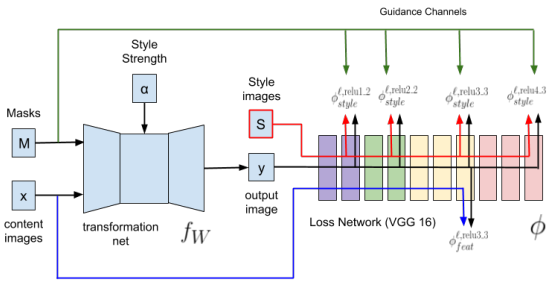


Fig. 3: Fast neural style Transfer with spatial and strength control.

III. Model Training, Data Set and Running Time

Our training data contains 10,000 natural images, in which 5000 were taken from MS-COCO `2017 Val images` and 5000 from `2017 Test images` [6]. As for style images, we mainly use collection of style images used in the previous style transfer works.

The models were trained and tested on a a desktop computer equipped with a NVIDIA Quadro P5000 graphics card. We resize training images to 256×256 and train the models with the batch size 10 for 5 epochs (each epoch takes about 500 seconds). Adam was used with learning rate 10^{-3}. We trained the model with strength control using style weight $5e4$, so that the trained models can cover strength from $5e3$ to $5e5$.

In table I, we present the running time comparison for three approaches with guidance masks to show the real time transfer availability.

Image Size	Gatys et al	FNST	FNSTSC
256	25.83	0.31	1.43
512	92.67	0.61	1.44
1024	376.67	2.6	1.56

TABLE I: Running time comparison in sec. The optimization approach ran with 200 iterations

IV. Results

In this section, we present the result generated by the models we explored. Figure 4 shows the style transfer and spatial control obtained by Gatys optimization approach and guiding channels.

We have trained multiple models for 4 pairs of two styles. Comparisons of images generated from the fast neural style transfer with strength control [5] and the basic framework as in [4] are given below. Figures 5 and 6 shows the style transfer guided by masks generated from [1] and figures 8, 7 present the results obtained from simple masks.

(a) Content (b) 1 Style (c) 2 Styles

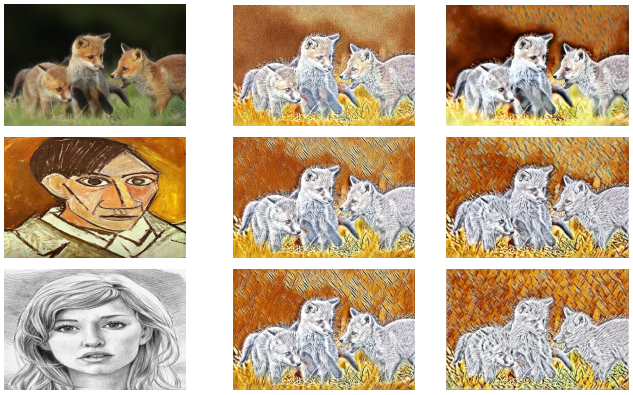
Fig. 4: Style transfer and spatial control using framework of [2].

Content & Style FNSTSC FNST

Fig. 5: Style transfer with model of combination of style feathers and candy has ratio of style weights and content weights set to 1×10^4, 5×10^4 and 1×10^5.

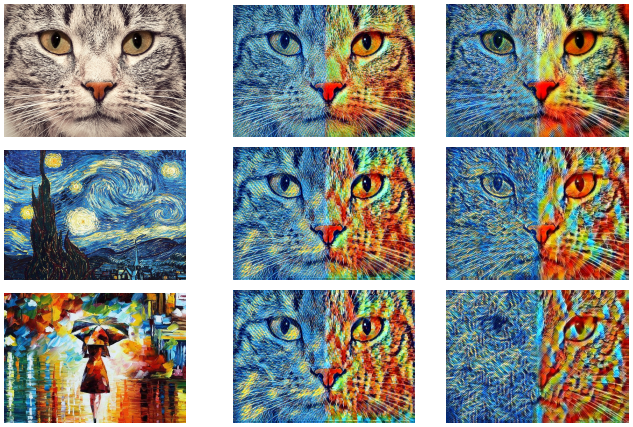
V. Limitations and Possible Future Extensions

After the the transformation network is trained, we are unable to change the styles chosen in advance. Also, we can only control the style strength globally on the entire image. In the future, we hope



Content & Style FNSTSC FNST

Fig. 6: Style transfer with model of combination of style picasso and pencil has ratio of style weights and content weights set to 1×10^4, 5×10^4 and 1×10^5.



Content & Style FNSTSC FNST

Fig. 7: Style transfer with model of combination of style starry night and rain princess with ratio of style weights and content weights set to 1×10^4, 5×10^4 and 1×10^5.

we can come up with a new architecture based on the current one, to achieve changing styles for real time and to control the strengths separately for different regions.

VI. The two or three things you learned during this project

Our project focuses on Style transfer, which is a emerging area in Computer Vision. When searching publications in its related field, we have explored many of them to come with unique one

Content & Style FNSTSC FNST

Fig. 8: Style transfer with model of combination of style mosaic and feathers has ratio of style weights and content weights set to 1×10^4, 5×10^4 and 1×10^5.

fitting our goal with spatial and strength control. We learnt from exploring various approaches by scrutinizing their distinct strengths and weaknesses. Also, we all learnt to be team players in this project. All team members discussed the model selection together, paying much attention to the details of our results.

VII. Advice you would give to next year's CV students (and instructors)

The homework arrangement could be more interesting, e.g. if all combined, the 3 homework sets could form an example course project. In this way we will understand more about the goal of each algorithm in this course.

References

[1] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Google inc.: Rethinking atrous convolution for semantic image segmentation. 2017.

[2] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[3] Leon A Gatys, Alexander S Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. Controlling perceptual factors in neural style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3985–3993, 2017.

[4] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[5] Victor Kitov. Real-time style transfer with strength control. In *International Conference on Computer Analysis of Images and Patterns*, pages 206–218. Springer, 2019.

[6] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.