Multi-style Transfer: Generalizing Fast Style Transfer to Several Genres

Brandon Cui, Calvin Qi, Aileen Wang
Stanford University, CS 231N Spring 2017

Motivation

Style transfer has seen its advent in recent years due to the intriguing visual results of being able to render images in a style of choice. Many current implementations of style transfer are well documented and produce good results using convolutional neural networks (CNN), but have drawbacks in performance and are limited to learning a style from just one image and producing a single pre-trained style network. We hope to implement style transfer in a more generalized form that is fast to run and also capable of combining various styles.

Problem Definition

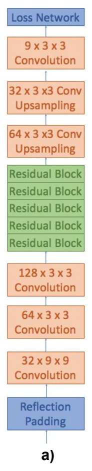
The goal of our project is to:
1. Implement fast feed forward style transfer network as proposed in Johnson et al. 2016. The framework comprises of a CNN transfer network, that is then fed through a VGG loss network that is pretrained off an image dataset.
2. Extend to multiple style transfer by incorporating multiple style losses into the VGG loss calculation.
3. Improve on existing multiple style transfer by having the network learn the blending weights of multi-style per iteration, rather than predefining the blending weights.

Approaches

1. Baseline: Gayts et al. 2015

- Uses *Perceptual Loss Functions* to define high level differences and iteratively optimize a single image.
- This produces high quality images, but is slow.

2. Fast Style Transfer Architecture (Johnson et al. 2016)



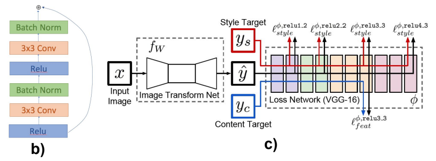


Figure 1: a) Image Transform Net, **b)** Residual Connections, **c)** Loss Network (*Johnson et al. 2016)*

- We design a feed-forward CNN that outputs an image of the same size, but in the style of an input image
- Can stylize images in less than a second, which is much faster than naive style transfer.

3. Nearest Neighbor Upsampling (Google, 2016)

- Convolution Transposes led to checkerboard patterns.
- Used nearest neighbor upsampling to produce smoother images.

4. Multiple Styles

- Simultaneously train multiple networks for multiple styles.
- Train networks to blend combinations of styles.
- This blending can be made into learnable parameters

Approaches (cont.)

5. Optimizing Total Loss

For all cases, we utilized the Adam optimizer to minimize total loss. The loss is defined as follows:

i) Feature Loss: $\ell_{feat}^{\phi,j}(\hat{y},y) = \frac{1}{C_j H_j W_j}\|\phi_j(\hat{y}) - \phi_j(y)\|_2^2$ where Φ_j is the output of the jth layer in the Loss network.

ii) Single-style Loss: $\ell_{style}^{\phi,j}(\hat{y},y) = \|G_j^\phi(\hat{y}) - G_j^\phi(y)\|_F^2$. where G_j is the Gram-matrix at the jth layer.

iii) TV Loss: $TV_\beta(x) = \sum_{i,j}\left((x_{i,j+1} - x_{ij})^2 + (x_{i+1,j} - x_{ij}))^2\right)^{\frac{\beta}{2}}$, here β=1. wiggles or "total variation" in the pixel values.

iv) Multi-Style Loss:Weighted average of all the input style loss for blending different styles.

6. Multi-Style Blending Weights

The loss of each input style (Si) can be blended as part of total multi-style (S) loss as follows:

$$L(S) = \sum_{i=1}^{n} w_i \cdot L(S_i)$$

The blending weights can be calculated by $w_i = L(S_i) / \sum_{i=1}^{n} L(S_i)$

7. Data source: We will choose different type of datasets to test and validate our multi-style transfer algorithm:
i) SqueezeNet for naive style transfer baseline
ii) VGG-16 and associated pre-trained ImageNet weights for loss network
iii) Microsoft COCO Image Dataset 2014 (80,000 images) for full network training

Results

We use the same picture of Tübingen as our content image in these results

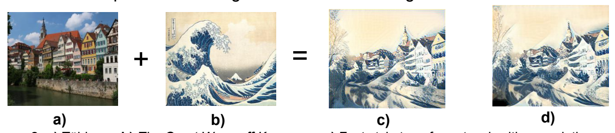


Figure 2: a) Tübingen **b)** *The Great Wave off Kanagawa* **c)** Fast style transfer network with convolution transposes **d)** Fast style transfer with nearest neighboring upsampling

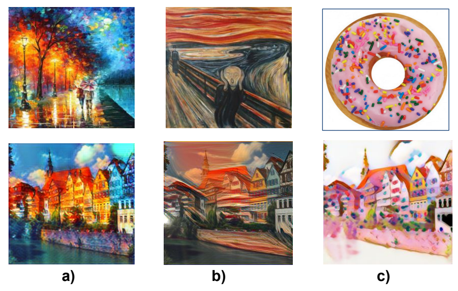


Figure 3: Original style images and output style images for **a)** *Alley by the Lake* **b)** *The Scream* **c)** A Donut

Results (cont.)

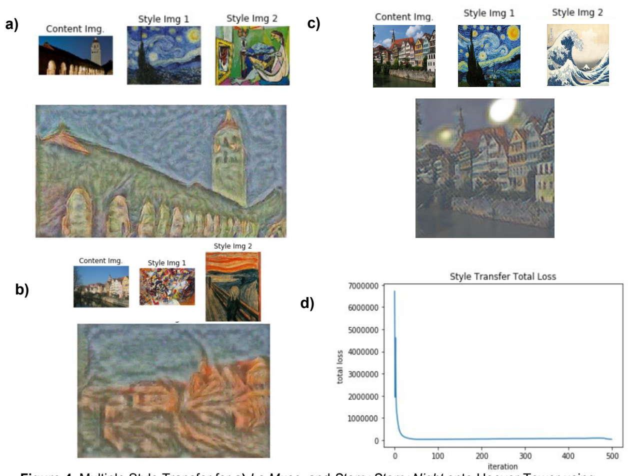


Figure 4. Multiple Style Transfer for **a)** *La Muse* and *Starry Starry Night* onto Hoover Tower using iterative optimization **b)** *Composition vii* and *The Scream* onto Tübingen **c)** *Starry Starry Night and The Great Wave onto* Tübingen **d)** Loss curve for multi-style loss

Analysis

Single Style Transfer

- After training the single-style transfer network, we can perform style transfer with **one fast forward pass** taking less than a second.
  - Much **faster than naive style transfer** which iterates on the image
  - Qualitatively the images are equally convincing, and quantitatively (based on values for style/feature loss) **comparable in performance**
  - Disadvantages: training time is much longer, and each model is restricted to only a limited set of styles

Multi-Style Transfer

- After running our multi-style transfer algorithm on two sets of examples (Figure 4 a, b, and c), we obtain **high quality visual results** combining the desired styles.
- Figure 4 d shows the decay pattern of total loss after 500 iterations, and indicates that our **optimization converges quite quickly.**
- Our multi-style transfer algorithm uses **trainable weights**, so it doesn't require hand-picking any hyper parameter scales for weighing different styles and automatically chooses the best combination
- This can be extended to an arbitrary number of styles

Future Work

In the future, we would like to expand our work to incorporate other aspects of image understanding. We could apply image segmentation so that our algorithm identifies different components of a picture as it applies styles to them, leading to more visually organized and consistent results.