Exploring Style Transfer: Extensions to Neural Style Transfer

Noah Makow
Stanford University
nmakow@stanford.edu

Pablo Hernandez
Stanford University
pabloh2@stanford.edu

Abstract

Recent work by Gatys et al. demonstrated the ability to distinguish an image's style from its content, allowing for the generation of new images that replicate the content and style of different images. The process of applying different styles to the same semantic image content through convolutional neural networks is known as Neural Style Transfer. In this paper, we explore several extensions and improvements to the original neural style transfer algorithm. By composing the neural style network with other architectures and by altering the original loss functions, we are able to achieve a number of effects including multiple style transfer, color preserving style transfer, and semantically segmented style transfer. We describe the architectural modifications required to achieve these effects and propose ideas for continued research.

1. Introduction

Human's cognitive ability to separate style from content in the visual field is something that is taken for granted yet enables much of the rich, emotional understanding that comes with vision. While much recent work in computer vision has focused on understanding content, recent research has investigated the ability of computers to understand style in conjunction with, and perhaps separately from, the content of an image.

Neural style transfer enables generating images that replicate the content of one image and the style of another. The input to the original neural style algorithm is two images – one content image and one style image. The output is an imaged generated by the network that minimizes the content loss with respect to the content image and style loss with respect to the style image. In this paper, we explore extensions to the original neural style transfer algorithm described by Gatys *et al.* to achieve a number of novel effects including multiple style transfer, color preserving style transfer, and semantically segmented style transfer [1].

2. Background & Related Work

Recent research has seen the bounds of computer vision grow significantly, with algorithms now surpassing human performance in image classification and improving significantly on tasks such as image segmentation and image captioning [1, 8, 16, 10]. The primary workhorse powering these advances has been the convolutional neural network (CNN), which operates by hierarchically constructing spatially-preserved feature representations of the content of the image. Each layer in a CNN comprises of a number of filters that each slide over the image and produce some activation map on the input to the layer, beginning with the original pixels of the image. By stacking many of these layers on top of one another (i.e. constructing "deep" networks), CNNs can develop abstract, high-level representations of the image content in the deeper layers of the network.

In 2015, Gatys *et al* produced the first work investigating the extraction style representations from images. Further, Gatys *et al* showed that it is possible to extract these style representations *separately* from the content representations of the image [5, 4, 6]. Whereas traditional CNNs hierarchically stack to form content representations, it is possible to form style representations by capturing the correlations of different filter responses within each layer. Given these findings, it is possible to generate new images that replicate the content of one image but the style of another, producing artistic images of high perceptual quality [12]. This method is explored in more detail in the methods section.

More recent work has investigated far more efficient models that produce similar effects to the original neural style approach. These alternate methods provide up to three orders of magnitude speed-up over the original Gatys approach by introducing a pre-trained feed-forward network that performs a forward-pass "image transformation" on the

[1]For our extensions, we build off of the CS231N Assignment 3 codebase in which we implemented the vanilla neural style algorithm.

input image before inputting it into the style transfer loss network. This means that style transfer is realistic for real-time video applications [9, 13, 14, 15].

In addition to work on improving the efficiency of style transfer, much work has been done on exploring the perceptual possibilities of neural style. Novel extensions to the original neural style algorithm have greatly improved the perceptual quality of generated images, as well as introduced a number of new features. These features include multiple style transfer [2], color-preserving style transfer [3], as well as content-aware style transfer [7]. We explore some of these extensions in greater detail below.

3. Methods

The baseline method for fast neural style transfer involves two separate components: an *image transformation network* f_W and a *loss network* ϕ that is used to define several loss functions $l_1, \dots, l_k$. The image transformation network is a deep residual convolutional network parametrized by weights W, transforming images x onto images $\hat{y}$ via the mapping $\hat{y} = f_W(x)$. Each loss function computes a scalar value measuring the difference between the output image $\hat{y}$ and a target image y_i. The image transformation network is trained to minimize a weighted combination of loss functions:

$$W^* = \arg\min_W E_{x,\{y_i\}} \Big[\sum_{i=1} \lambda_i l_i(f_W(x), y_i) \Big].$$

The loss network ϕ is used to define a *feature reconstruction loss* l^{ϕ}_{feat} and a *style reconstruction loss* l^{ϕ}_{style} that measure the differences in content and style per between images. For each input image x we have a content target y_c and a style target y_s. In the case of style transfer, y_c is the input image x and y_s is the stylized or artistic image of which we would like to extract the style. One network is trained per style target.

3.1. Image Transformation Networks

The image transformation network is a deep residual convolutional network parametrized by weights W, transforming images x onto images $\hat{y}$ via the mapping $\hat{y} = f_W(x)$.

Inputs and Outputs. Both input and output are color images of shape $3 \times 256 \times 256$. The image transformation network is fully convolutional, so at test time it can be applied to images of any resolution.

Downsampling and Upsampling. We make use of downsampling via convolutional layers with stride 2 and upsampling via convolutional layers with stride $1/2$ sandwiching several residual blocks. This provides the computational benefit of operating primarily in a lower dimensional space as well as increased effective receptive field sizes of the neurons in the residual blocks.

3.2. Perceptual Losses

There are two perceptual loss functions of interest: feature reconstruction loss and the style reconstruction loss. Both make use of a pre-trained loss network ϕ trained for image classification on the ImageNet dataset.

Feature Reconstruction Loss. We encourage the pixels of the output image $\hat{y}$ to have similar feature representaitons as computed by the loss network ϕ. Let $\phi_j(x)$ be the activations of the jth layer of the network ϕ when processing x (of shape $C_j \times H_j \times W_j$). The feature reconstruction loss is defined as the squared, normalized Euclidean distance between feature representations:

$$l^{\phi,j}_{feat}(\hat{y}, y) = \frac{1}{C_j H_j W_j} ||\phi_j(\hat{y}) - \phi_j(y)||_2^2.$$

As noted in Johnson et al, minimizing feature reconstruction forces $\hat{y}$ and y to be similar perceptually but not necessarily match exactly.

Style Reconstruction Loss. The style reconstruction loss penalizes differences in style, such as differences in colors, textures, common patterns, etc. As above, let $\phi_j(x)$ be the activations of the jth layer of the loss network ϕ for input x (of shape $C_j \times H_j \times W_j$). Define the Gram matrix $G^{\phi}_j(x)$ to be the $C_j \times C_j$ matrix whose elements are given by:

$$G^{\phi}_j(x)_{c,c'} = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \phi_j(x)_{h,w,c} \phi_j(x)_{h,w,c'}.$$

The style reconstruction loss is then the squared Frobenius norm of the difference between the Gram matrices of the output and target images:

$$l^{\phi,j}_{style}(\hat{y}, y) = ||G^{\phi}_j(\hat{y}) - G^{\phi}_j(y)||_F^2.$$

As demonstrated by Gatys et al. [4], minimizing the style reconstruction loss will result in generating an image that preserves stylistic features from the target image but does not preserve spatial structure. Minimizing the style reconstruction loss at higher layers transfers larger-scale structure. To perform style reconstruction on a set of layers J, we define $l^{\phi,J}_{style}$ to be the sum of losses for each layer $j \in J$.

3.3. Simple Loss Functions

In addition to the perceptual loss functions above, we introduce two more loss functions that depend solely on low-level pixel information.

Pixel loss. The pixel loss is the normalized Euclidean distance between the output image $\hat{y}$ and the target y:

$$l_{pixel} = \frac{1}{CHW}(\hat{y}, y) = ||\hat{y} - y||_2^2.$$

Total Variation Regularization. To encourage spatial smoothness, we follow prior work and make use of a total variation regularizer $l_{TV}(\hat{y})$.

3.4. Multiple Style Transfer

An extension to the vanilla neural style transfer algorithm is to allow for the styles of multiple images to be transferred to a single content image. Achieving this effect requires a simple modification to the style loss function posed above to be a weighted sum over multiple style images. Let $l_{style}^{\phi,J}(\hat{y}, y)$ be the total style loss for output image $\hat{y}$ and style image y. Then, for a set of style images $\{y_1, y_2, \ldots, y_n\}$, we can define

$$l_{multi} = \sum_{i=1}^{n} w_i l_{style}^{\phi,J_i}(\hat{y}, y_i),$$

where J_i is the set of layers corresponding to style image y_i and w_i is the weight corresponding to style image y_i. This formulation allows us to flexibly choose the style layers and style layer weights independently for each style image in order to achieve the best results for each style image. We are then able to weight the overall effect of each style image on the generated image via the style image weights.

3.5. Color Preserving Style Transfer

To achieve color preserving style transfer, we run the original style transfer algorithm and apply a luminance-only transfer from the content image to the output image. First, we extract the luminance channels L_S and L_C from style and transfer images. The neural style algorithm is run to produce an output image with luminance channels L_T. In the YIQ color space, the color information of the content image are represtend by the I and Q channels and can be combined the L_T to produce the final image with the original colors preserved. Thus, the final resulting image uses the Y channel from the output image L_T, and the I and Q channels from the original content image L_C.

3.6. Semantically Segmented Style Transfer

Semantic segmentation involves clustering parts of an input image together that belong in the same object class. It entails classifying each pixel in the input image to ultimately recognize the objects present in any image. Achieving semantically segmented style transfer is similar to the process of vanilla style transfer with the exception of first generating a mask for each input image. The mask is of shape $H \times W$ and contains ones in the pixel locations where would like to apply the gradient and zeros in the areas where we do not want to apply the gradient. We compute the same content loss when conducting segmented style transfer, but the style loss has a slight variation. Here, we choose to only apply the style gradients to the desired segments of the image - using the input mask as a filter. This entails a simple element-wise multiplication with the mask. Figure 1 shows examples of our generated input masks used for segmented style transfer.

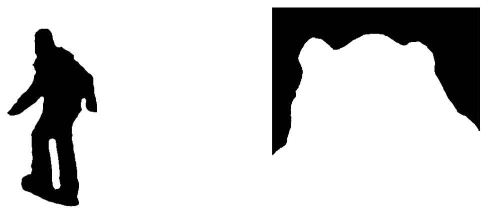


Figure 1: Example image masks for segmented style transfer.

4. Dataset & Features

4.1. Dataset

We chose a random sample of images from the Microsoft COCO dataset [11] as well as 5 different style images from various artists. We also selected additional visually pleasing input content images for the purposes of this report (Figure 2), and five art pieces to be used as input style images (Figure 3).

(a) Stanford (b) New York skyline

Figure 2: Example content images.

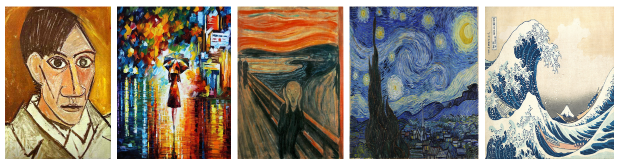
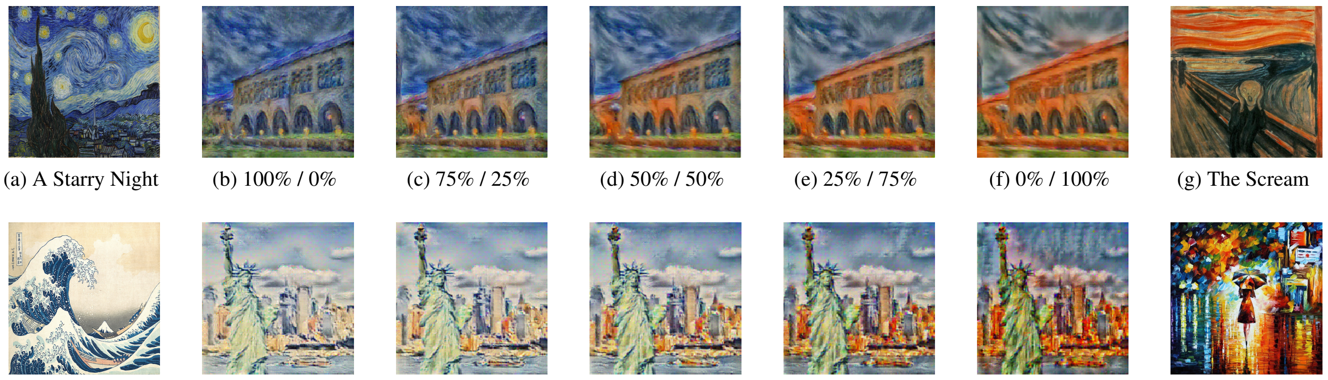

(a) (b) (c) (d) (e)

Figure 3: We used (a) Picasso's Self-Portrait by Pablo Picasso, (b) Rain Princess by Leonid Afremov, (c) The Scream by Edvard Munch, (d) Starry Night by Vincent Van Gogh, and (e) The Great Wave off Kanagawa by Hokusai as our artistic style images.


(a) A Starry Night (b) 100% / 0% (c) 75% / 25% (d) 50% / 50% (e) 25% / 75% (f) 0% / 100% (g) The Scream

(h) The Great Wave (i) 100% / 0% (j) 75% / 25% (k) 50% / 50% (l) 25% / 75% (m) 0% / 100% (n) Rain Princess

Figure 4: Blending between Starry Night and The Scream for the Stanford image and between The Great Wave and Rain Princess for the skyline image via multiple style transfer.

5. Results

5.1. Multiple Style Transfer

Our method proposed above allows us to readily generate images that blend the styles of multiple images. Figure 4 shows the ability to blend between two styles seamlessly. We see that adding multiple styles can somewhat lessen the impact of each individual style, especially when the styles are somewhat different.

As with the basic neural style algorithm, we train with the Adam optimizer. We are able to converge relatively quickly to minimize the loss functions for each style image. Predictably, when forced to blend between multiple styles we arrive at a higher style loss for that image than when operating with just one style image at a time.

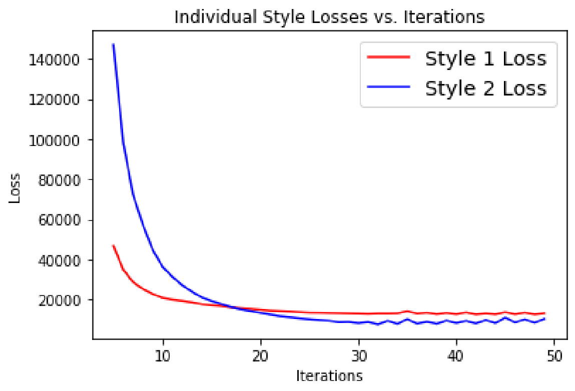


Figure 5: Style loss versus iteration for each style image when blending 50%/50% between two style images.

5.2. Color Preserving Style Transfer

Luminance-only transfer works very well and takes only a simple transformation at the end of the normal style transfer algorithm. Below we compare the results between color-preserving and normal style transfer with all other hyperparameters held constant.

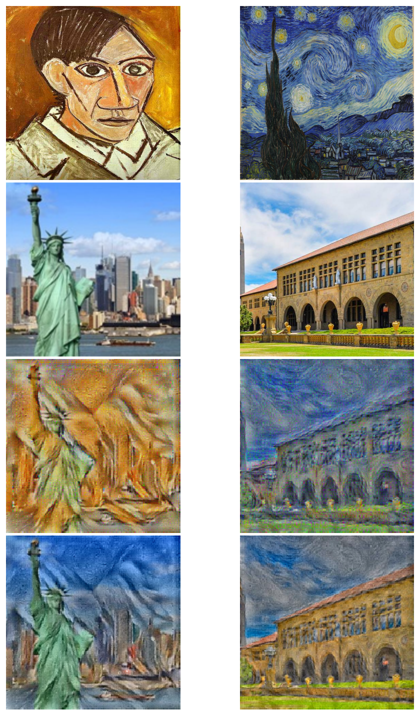


Figure 6: A demonstration of color preserving style transfer, with the first row being the style image, the second being normal style transfer, and the last being color-preserving style transfer.

5.3. Semantically Segmented Style Transfer

Upon generating the individual masks for the original content images, and tweaking the loss function as to only transfer style to the areas where we would like to apply the gradient, the following images are able to be produced.

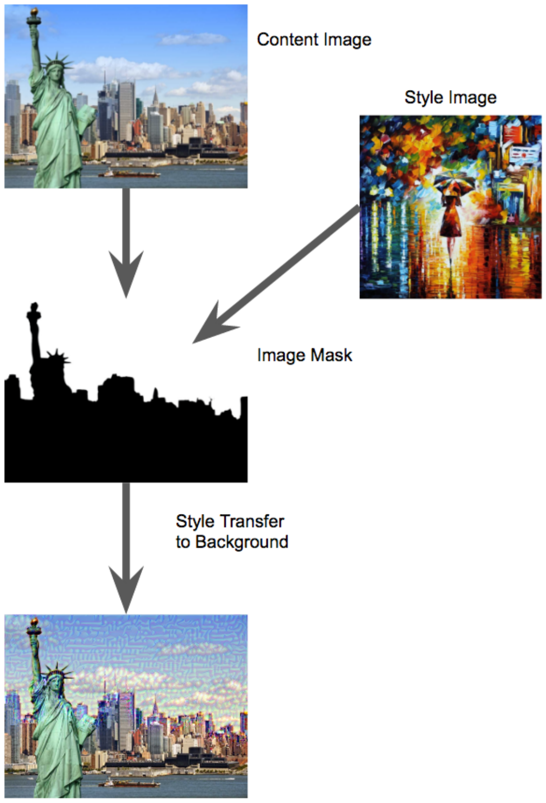


Figure 7: Flow of segmented style transfer.

Figure 8 shows a demonstration of segmented style transfer applied to two different content images. Semantic segmentation can be selectively applied to either objects in the foreground or the background as a whole.

6. Conclusion & Future Work

Our experiments with the neural style algorithm show the flexibility and potential further uses cases for style transfer. The fast variant on the neural style algorithm presented by Johnson [9] make these sorts of effects possible to achieve in real-time video processing applications. Already we are seeing style transfer being implemented in products such as Facebook's messenger service (see Figure 8). The extensions that we have presented – namely multiple style transfer, color-preserving style transfer, and semantically segmented style transfer – are simple extensions that will continue to improve the perceptual quality and novel applications achievable through style transfer.

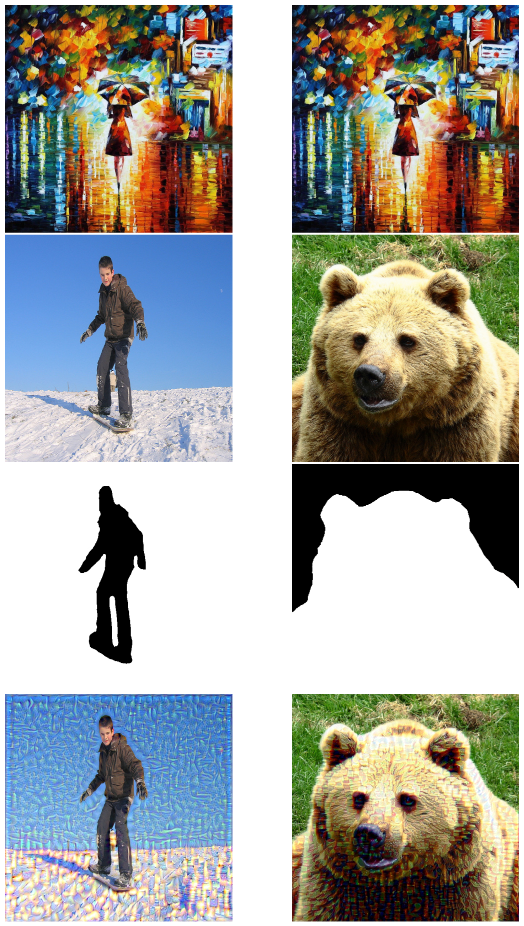


Figure 8: A demonstration of segmented style transfer, with the first row being the style image, the second being the content image, third being the image mask, and the last being the segmented style transfer.

We've seen that these extensions enable interesting new perceptual effects. However, a number of other interesting extensions can continue to improve the quality of style transfer. Specifically, content-aware style transfer attempts to spatially target certain areas of the style image to be applied to certain areas of the content image [7, 17]. Further, region-based methods seek to achieve a similar goal by replicating specific styles in certain regions of the content image.

Further, we can produce any arbitrary mask image rather than masks produced via semantic segmentation, such as a gradient mask that linearly interpolates from 0 to 1 from one region of the image to another. With no alteration to algorithm, we can apply this mask to achieve effects such as applying different styles to different regions of the image, as well as blending between styles within a single output image.

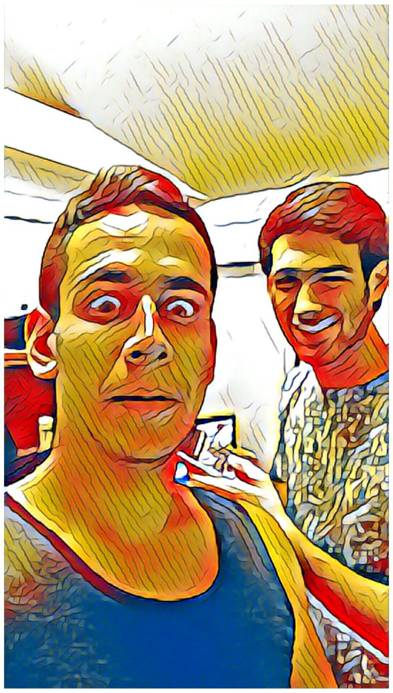


Figure 9: An example of Facebook's implementation of fast neural style in their Messenger app applied to the two models that wrote this paper.

Photorealistic style transfer is another area of potential future research. Rather than using artistic images as style images, we can simply use another photograph. This allows for interesting effects such as season transfer, where we attempt to transition a photo of nature from one season to another. Combining photorealistic style transfer with semantic segmentation may allow for a primitive form of image-to-image translation, which is often performed via adversarial networks [18]. For example, we may be able to convert an image of horses to an image of zebras.

Each of these extensions provide paths forward for improving the perceptual quality of style transfer. While style transfer can be applied to a number of interesting applications such as graphics, art, and real-time video processing, it is also provides interesting perspective into how humans perceive style in visual content. As we create agents capable of recognizing stylistic content in addition to semantic content, we will achieve intelligent agents that have a deeper understanding of the world around them.

References

[1] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[2] V. Dumoulin, J. Shlens, and M. Kudlur. A learned representation for artistic style. 2016.

[3] L. A. Gatys, M. Bethge, A. Hertzmann, and E. Shechtman. Preserving color in neural artistic style transfer. *arXiv preprint arXiv:1606.05897*, 2016.

[4] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[5] L. A. 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] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016.

[7] L. A. Gatys, A. S. Ecker, M. Bethge, A. Hertzmann, and E. Shechtman. Controlling perceptual factors in neural style transfer. *arXiv preprint arXiv:1611.07865*, 2016.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[9] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[11] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[12] Y. Nikulin and R. Novak. Exploring the neural algorithm of artistic style. *arXiv preprint arXiv:1602.07188*, 2016.

[13] M. Ruder, A. Dosovitskiy, and T. Brox. Artistic style transfer for videos. In *German Conference on Pattern Recognition*, pages 26–36. Springer, 2016.

[14] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *Int. Conf. on Machine Learning (ICML)*, 2016.

[15] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[16] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.

[17] R. Yin. Content aware neural style transfer. *arXiv preprint arXiv:1601.04568*, 2016.

[18] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.