A LEARNED REPRESENTATION FOR ARTISTIC STYLE

Vincent Dumoulin & Jonathon Shlens & Manjunath Kudlur
Google Brain, Mountain View, CA
vi.dumoulin@gmail.com, shlens@google.com, keveman@google.com

ABSTRACT

The diversity of painting styles represents a rich visual vocabulary for the construction of an image. The degree to which one may learn and parsimoniously capture this visual vocabulary measures our understanding of the higher level features of paintings, if not images in general. In this work we investigate the construction of a single, scalable deep network that can parsimoniously capture the artistic style of a diversity of paintings. We demonstrate that such a network generalizes across a diversity of artistic styles by reducing a painting to a point in an embedding space. Importantly, this model permits a user to explore new painting styles by arbitrarily combining the styles learned from individual paintings. We hope that this work provides a useful step towards building rich models of paintings and offers a window on to the structure of the learned representation of artistic style.

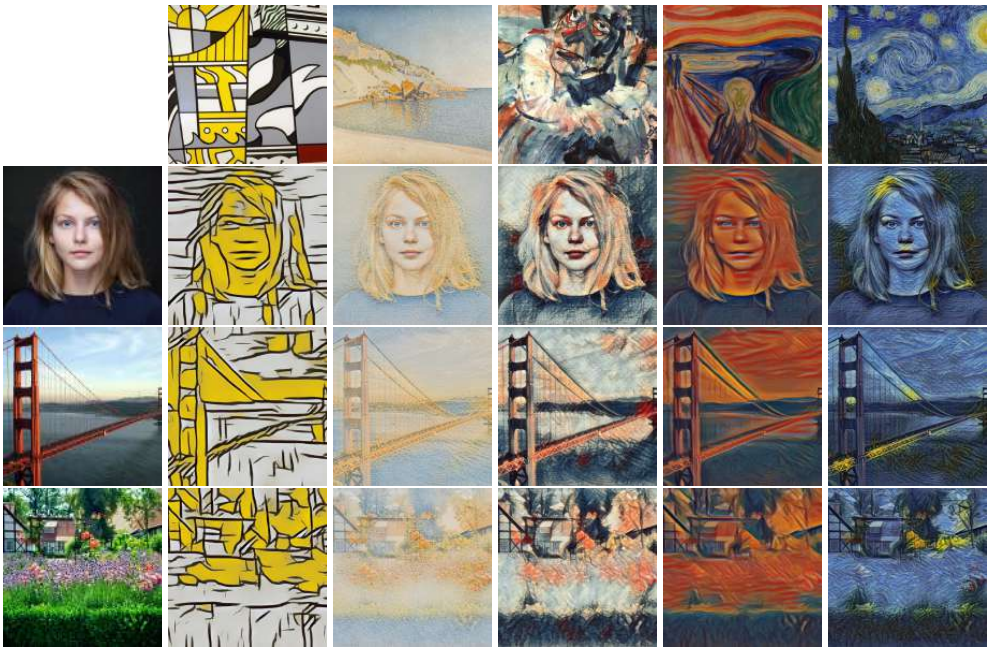
1 INTRODUCTION

A pastiche is an artistic work that imitates the style of another one. Computer vision and more recently machine learning have a history of trying to automate pastiche, that is, render an image in the style of another one. This task is called *style transfer*, and is closely related to the texture synthesis task. While the latter tries to capture the statistical relationship between the pixels of a source image which is assumed to have a stationary distribution at some scale, the former does so while also attempting to preserve some notion of content.

On the computer vision side, Efros & Leung (1999) and Wei & Levoy (2000) attempt to "grow" textures one pixel at a time using non-parametric sampling of pixels in an examplar image. Efros & Freeman (2001) and Liang et al. (2001) extend this idea to "growing" textures one patch at a time, and Efros & Freeman (2001) uses the approach to implement "texture transfer", i.e. transfering the texture of an object onto another one. Kwatra et al. (2005) approaches the texture synthesis problem from an energy minimization perspective, progressively refining the texture using an EM-like algorithm. Hertzmann et al. (2001) introduces the concept of "image analogies": given a pair of "unfiltered" and "filtered" versions of an examplar image, a target image is processed to create an analogous "filtered" result. More recently, Frigo et al. (2016) treats style transfer as a local texture transfer (using an adaptive patch partition) followed by a global color transfer, and Elad & Milanfar (2016) extends Kwatra's energy-based method into a style transfer algorithm by taking content similarity into account.

On the machine learning side, it has been shown that a trained classifier can be used as a feature extractor to drive texture synthesis and style transfer. Gatys et al. (2015a) uses the VGG-19 network (Simonyan & Zisserman, 2014) to extract features from a texture image and a synthesized texture. The two sets of features are compared and the synthesized texture is modified by gradient descent so that the two sets of features are as close as possible. Gatys et al. (2015b) extends this idea to style transfer by adding the constraint that the synthesized image also be close to a content image with respect to another set of features extracted by the trained VGG-19 classifier.

While very flexible, this algorithm is expensive to run due to the optimization loop being carried. Ulyanov et al. (2016a) and Johnson et al. (2016) tackle this problem by introducing a *feedforward style transfer network*, which is trained to go from content to pastiche image in one pass. However, in doing so some of the flexibility of the original algorithm is lost: the style transfer network is tied to a single style, which means that separate networks have to be trained for every style being



(a) With conditional instance normalization, a single style transfer network can capture 32 styles at the same time, five of which are shown here. All 32 styles in this single model are in the Appendix. Golden Gate Bridge photograph by Rich Niewiroski Jr.

(b) The style representation learned via conditional instance normalization permits the arbitrary combination of artistic styles. Each pastiche in the sequence corresponds to a different step in interpolating between the γ and β values associated with two styles the model was trained on.

Figure 1: Pastiches produced by a style transfer network trained on 32 styles chosen for their variety.

modeled. Subsequent work has brought some performance improvements to style transfer networks, e.g. with respect to color preservation (Gatys et al., 2016) or style transfer quality (Ulyanov et al., 2016b), but to our knowledge the problem of the single-purpose nature of style transfer networks remains untackled.

We think this is an important problem that, if solved, would have both scientific and practical importance. First, style transfer has already found use in mobile applications, for which on-device processing is contingent upon the models having a reasonable memory footprint. More broadly, building a separate newtork for each style ignores the fact that individual paintings share many common visual elements and a true model that captures artistic style would be able to exploit and learn from such regularities. Furthermore, the degree to which an artistic styling model might generalize across painting styles would directly measure our ability to build systems that parsimoniously capture the higher level features and statistics of photographs and images (Simoncelli & Olshausen, 2001).

In this work, we show that a simple modification of the style transfer network, namely the introduction of *conditional instance normalization*, allows it to learn multiple styles (Figure 1a).We demonstrate that this approach is flexible yet comparable to single-purpose style transfer networks, both qualitatively and in terms of convergence properties. This model reduces each style image into a point in an embedding space. Furthermore, this model provides a generic representation for artistic styles that seems flexible enough to capture new artistic styles much faster than a single-purpose net-

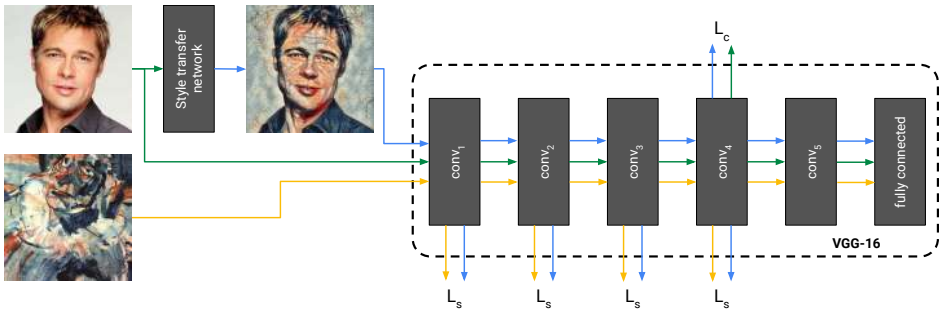


Figure 2: Style transfer network training diagram. A pastiche image is produced by feeding a content image through the style transfer network. The two images, along with a style image, are passed through a trained classifier, and the resulting intermediate representations are used to compute the content loss $\mathcal{L}_c$ and style loss $\mathcal{L}_s$. The parameters of the classifier are kept fixed throughout training.

work. Finally, we show that the embeddding space representation permits one to arbitrarily combine artistic styles in novel ways not previously observed (Figure 1b).

2 Style transfer with deep networks

Style transfer can be defined as finding a pastiche image p whose content is similar to that of a content image c but whose style is similar to that of a style image s. This objective is by nature vaguely defined, because similarity in content and style are themselves vaguely defined.

The neural algorithm of artistic style proposes the following definitions:

- Two images are similar in content if their high-level features as extracted by a trained classifier are close in Euclidian distance.
- Two images are similar in style if their low-level features as extracted by a trained classifier share the same statistics or, more concretely, if the difference between the features' Gram matrices has a small Frobenius norm.

The first point is motivated by the empirical observation that high-level features in classifiers tend to correspond to higher levels of abstractions (see Zeiler & Fergus (2014) for visualizations; see Johnson et al. (2016) for style transfer features). The second point is motivated by the observation that the artistic style of a painting may be interpreted as a visual texture (Gatys et al., 2015a). A visual texture is conjectured to be spatially homogenous and consist of repeated structural motifs whose minimal sufficient statistics are captured by lower order statistical measurements (Julesz, 1962; Portilla & Simoncelli, 1999).

In its original formulation, the neural algorithm of artistic style proceeds as follows: starting from some initialization of p (e.g. c, or some random initialization), the algorithm adapts p to minimize the loss function

$$\mathcal{L}(s, c, p) = \lambda_s \mathcal{L}_s(p) + \lambda_c \mathcal{L}_c(p), \tag{1}$$

where $\mathcal{L}_s(p)$ is the style loss, $\mathcal{L}_c(p)$ is the content loss and λ_s, λ_c are scaling hyperparameters. Given a set of "style layers" $\mathcal{S}$ and a set of "content layers" $\mathcal{C}$, the style and content losses are themselves defined as

$$\mathcal{L}_s(p) = \sum_{i \in \mathcal{S}} \frac{1}{U_i} \mid\mid G(\phi_i(p)) - G(\phi_i(s)) \mid\mid_F^2 \tag{2}$$

$$\mathcal{L}_c(p) = \sum_{j \in \mathcal{C}} \frac{1}{U_j} \mid\mid \phi_j(p) - \phi_j(s) \mid\mid_2^2 \tag{3}$$

where $\phi_l(x)$ are the classifier activations at layer l, U_l is the total number of units at layer l and $G(\phi_l(x))$ is the Gram matrix associated with the layer l activations.

In practice, we set $\lambda_c = 1.0$ and and leave λ_s as a free hyper-parameter.

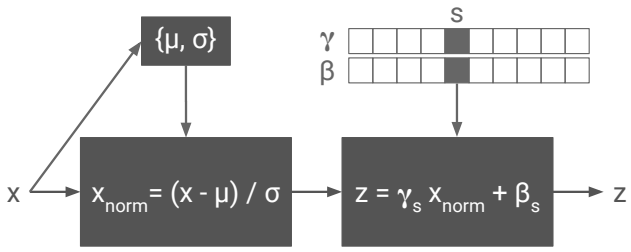


Figure 3: Conditional instance normalization. The input activation x is normalized across both spatial dimensions and subsequently scaled and shifted using style-dependent parameter vectors γ_s, β_s where s indexes the style label.

2.1 FEEDFORWARD STYLE TRANSFER NETWORKS

In order to speed up the procedure outlined above, a style transfer network T is introduced (Figure 2). It takes as input a content image c and outputs the pastiche image p directly. The network is trained on many content images using the same loss function as above, i.e.

$$\mathcal{L}(s, c) = \lambda_s \mathcal{L}_s(T(c)) + \lambda_c \mathcal{L}_c(T(c)). \tag{4}$$

While feedforward style transfer networks solve the problem of speed at test-time, they also suffer from the fact that they are tied to one specific style. This means that a separate network has to be trained for *every* style to be imitated. The real-world impact of this limitation is that it becomes prohibitive to implement a style transfer application on a memory-limited device, such as a smartphone.

2.2 N-STYLES FEEDFORWARD STYLE TRANSFER NETWORKS

Our work stems from the intuition that many styles probably share some degree of computation, and that this sharing is thrown away by training N networks from scratch when building an N-styles style transfer system. For instance, many impressionist paintings share similar paint strokes but differ in the color palette being used. In that case, it seems very wasteful to treat a set of N impressionist paintings as completely separate styles.

To take this into account, we propose to train a single conditional style transfer network $T(c, s)$ for N styles. The conditional network is given both a content image and the identity of the style to apply and produces a pastiche corresponding to that style. While the idea is straightforward on paper, there remains the open question of how conditioning should be done. In exploring this question, we found a very surprising fact about the role of normalization in style transfer networks: to model a style, it is sufficient to specialize scaling and shifting parameters after normalization to each specific style. In other words, all convolutional weights of a style transfer network can be shared across many styles, and it is sufficient to tune parameters for an affine transformation after normalization for each style.

We call this approach *conditional instance normalization*. Building off the instance normalization technique proposed in Ulyanov et al. (2016b), we augment the γ and β parameters so that they're $N \times C$ matrices, where N is the number of styles being modeled and C is the number of output feature maps. Conditioning on a style is achieved as follows:

$$z = \gamma_s \left(\frac{x - \mu}{\sigma} \right) + \beta_s \tag{5}$$

where μ and σ are x's mean and standard deviation taken across spatial axes and γ_s and β_s are obtained by selecting the row corresponding to s in the γ and β matrices (Figure 3). One added benefit of this approach is that one can stylize a single image into N painting styles with a single feed forward pass of the network. In constrast, a single-style network requires N feed forward passes to perform N style transfers (Johnson et al., 2016; Ulyanov et al., 2016a).

Because conditional instance normalization only acts on the scaling and shifting parameters, training a style transfer network on N styles requires much less parameters than the naive approach of

Figure 4: A single style transfer network was trained to capture the style of 10 Monet paintings, five of which are shown here. All 10 styles in this single model are in the Appendix. Golden Gate Bridge photograph by Rich Niewiroski Jr.

training N separate networks. In a typical network setup, the model consists of roughly 1.6M parameters, only around 3K (or 0.2%) of which specify individual artistic styles. In fact, because the size of γ and β grows linearly with respect to the number of feature maps in the network, this approach requires $O(N \times L)$ parameters, where L is the total number of feature maps in the network.

In addition, as is discussed in subsection 3.4, conditional instance normalization presents the advantage that integrating an $N + 1^{th}$ style to the network is cheap because of the very small number of parameters to train.

3 Experimental results

3.1 Methodology

Unless noted otherwise, all style transfer networks were trained using the hyperparameters outlined in the Appendix's Table 1.

We used the same network architecture as in Johnson et al. (2016), except for two key details: zero-padding is replaced with mirror-padding, and transposed convolutions (also sometimes called *deconvolutions*) are replaced with nearest-neighbor upsampling followed by a convolution. The use of mirror-padding avoids border patterns sometimes caused by zero-padding in SAME-padded convolutions, while the replacement for transposed convolutions avoids checkerboard patterning, as discussed in in Odena et al. (2016). We find that with these two improvements training the network no longer requires a total variation loss that was previously employed to remove high frequency noise as proposed in Johnson et al. (2016).

The evaluation images used for this work were resized such that their smaller side has size 512. Their stylized versions were then center-cropped to 512x512 pixels for display.

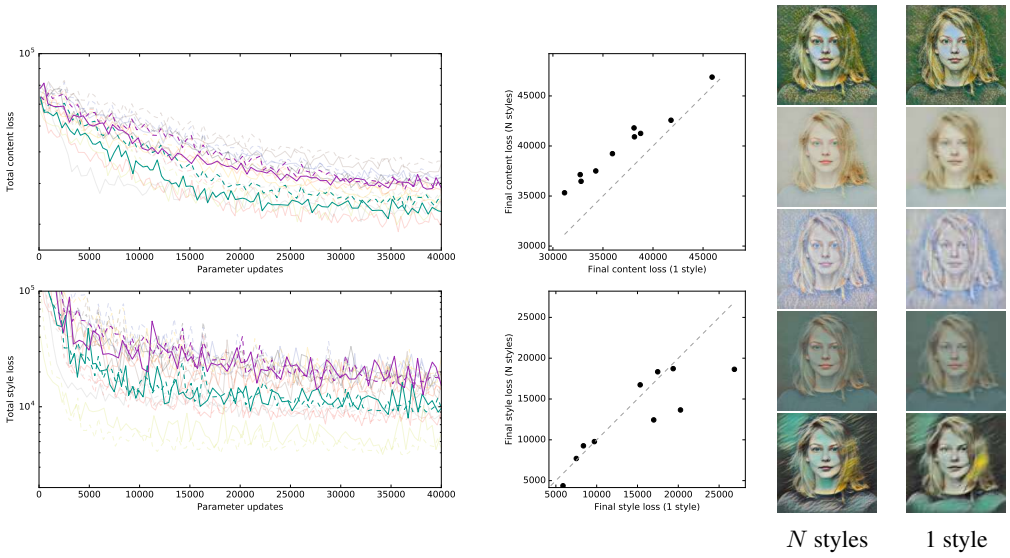


Figure 5: The N-styles model exhibits learning dynamics comparable to individual models. (Left column) The N-styles model converges slightly slower in terms of content loss (top) and as fast in terms of style loss (bottom) than individual models. Dashed curves represent the N-styles model, and full curves represent individual models. Emphasis has been added on two arbitrary styles for visualization purposes. (Center column) The N-styles model reaches a slightly higher final content loss than (top, $8.7 \pm 3.9\%$ increase) and a final style loss comparable to (bottom, $8.9 \pm 16.5\%$ decrease) individual models. (Right column) Pastiches produced by the N-styles network are qualitatively comparable to those produced by individual networks.

3.2 Training a single network on N styles produces stylizations comparable to independently-trained models

As a first test, we trained a 10-styles model on stylistically similar images, namely 10 impressionist paintings from Claude Monet. Figure 4 shows the result of applying the trained network on evaluation images for a subset of the styles, with the full results being displayed in the Appendix. The model captures different color palettes and textures. We emphasize that 99.8% of the parameters are shared across all styles in contrast to 0.2% of the parameters which are unique to each painting style.

To get a sense of what is being traded off by folding 10 styles into a single network, we trained a separate, single-style network on each style and compared them to the 10-styles network in terms of style transfer quality and training speed (Figure 5).

The left column compares the learning curves for style and content losses between the single-style networks and the 10-styles network. The losses were averaged over 32 random batches of content images. By visual inspection, we observe that the 10-styles network converges as quickly as the single-style networks in terms of style loss, but lags slightly behind in terms of content loss.

In order to quantify this observation, we compare the final losses for 10-styles and single-style models (center column). The 10-styles network's content loss is around $8.7 \pm 3.9\%$ higher than its single-style counterparts, while the difference in style losses ($8.9 \pm 16.5\%$ lower) is insignificant. While the N-styles network suffers from a slight decrease in content loss convergence speed, this may not be a fair comparison, given that it takes N times more parameter updates to train N single-style networks separately than to train them with an N-styles network.

The right column shows a comparison between the pastiches produced by the 10-styles network and the ones produced by the single-style networks. We see that both results are qualitatively similar.

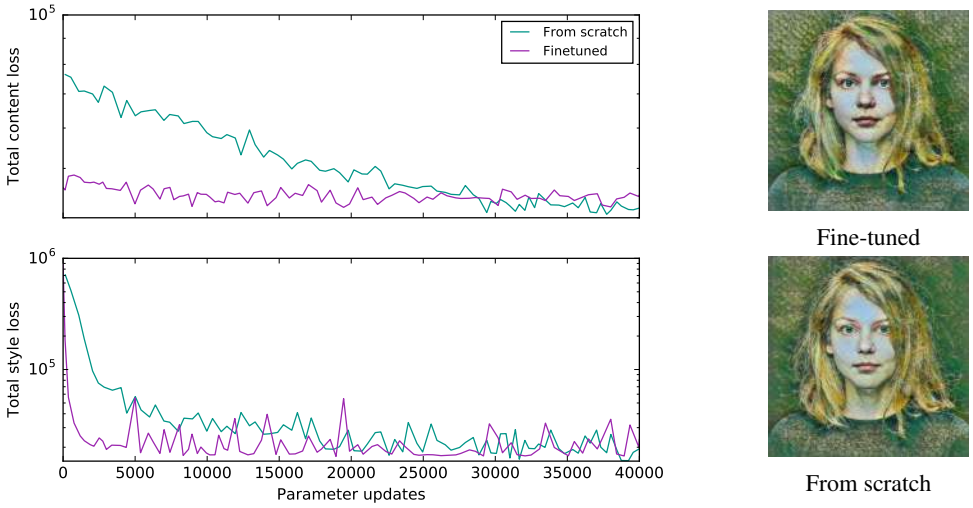


Figure 6: The trained network is efficient at learning new styles. (Left column) Learning γ and β from a trained style transfer network converges much faster than training a model from scratch. (Right) Learning γ and β for 5,000 steps from a trained style transfer network produces pastiches comparable to that of a single network trained from scratch for 40,000 steps.

3.3 THE N-STYLES MODEL IS FLEXIBLE ENOUGH TO CAPTURE VERY DIFFERENT STYLES

We evaluated the flexibility of the N-styles model by training a style transfer network on 32 works of art chosen for their diversity. Figure 1a shows the result of applying the trained network on evaluation images for a subset of the styles. Once again, the full results are displayed in the Appendix. The model appears to be capable of modeling all 32 styles in spite of the tremendous variation in color palette and the spatial scale of the painting styles.

3.4 THE TRAINED NETWORK GENERALIZES ACROSS PAINTING STYLES

Since all weights in the transformer network are shared between styles, one way to incorporate a new style to a trained network is to keep the trained weights fixed and learn a new set of γ and β parameters. To test the efficiency of this approach, we used it to incrementally incorporate Monet's *Plum Trees in Blossom* painting to the network trained on 32 varied styles. Figure 6 shows that doing so is much faster than training a new network from scratch (left) while yielding comparable pastiches: even after eight times fewer parameter updates than its single-style counterpart, the fine-tuned model produces comparable pastiches (right).

3.5 THE TRAINED NETWORK CAN ARBITRARILY COMBINE PAINTING STYLES

The conditional instance normalization approach raises some interesting questions about style representation. In learning a different set of γ and β parameters for every style, we are in some sense learning an embedding of styles.

To probe the utility of this embedding, we tried convex combinations of the γ and β values of very distinct styles (Figure 1b; Figure 7, left column). Employing a single convex combination produces a smooth transition from one style to the other. Suppose (γ_1, β_1) and (γ_2, β_2) are the parameters corresponding to two different styles. We use $\gamma = \alpha \times \gamma_1 + (1-\alpha) \times \gamma_2$ and $\beta = \alpha \times \beta_1 + (1-\alpha) \times \beta_2$ to stylize an image. Figure 7 (right column) shows the style loss from the transformer network for a given source image, with respect to the *Bicentennial Print* and *Head of a Clown* paintings, as we vary α from 0 to 1. As α increases, the style loss with respect to *Bicentennial Print* increases, which explains the smooth fading out of that style's artifact in the transformed image.

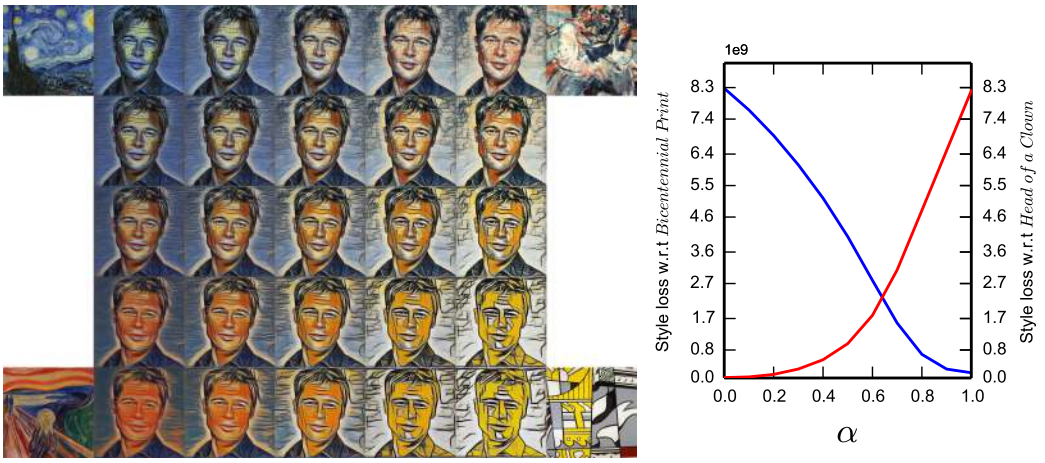


Figure 7: The N-styles network can arbitrarily combine artistic styles. (Left) Combining four styles, shown in the corners. Each pastiche corresponds to a different convex combination of the four styles' γ and β values. (Right) As we transition from one style to another (*Bicentennial Print* and *Head of a Clown* in this case), the style losses vary monotonically.

4 Discussion

It seems very surprising that such a small proportion of the network's parameters can have such an impact on the overall process of style transfer. It could be that the network architecture is overspecified for the task. Another interpretation could be that the convolutional weights of the style transfer network encode transformations that represent "elements of style". The scaling and shifting factors would then provide a way for each style to inhibit or enhance the expression of various elements of style to form a global identity of style. While this work does not attempt to verify this hypothesis, we think that this would constitute a very promising direction of research in understanding the computation behind style transfer networks as well as the representation of images in general.

The question of how predictive each style image is of its corresponding style representation is also of great interest. If it is the case that the style representation can easily be predicted from a style image, one could imagine building a transformer network which skips learning an individual conditional embedding and instead learn to produce a pastiche directly from a style and a content image, much like in the original neural algorithm of artistic style, but without any optimization loop at test time.

Finally, the learned style representation opens the door to generative models of style: by modeling enough paintings of a given artistic movement (e.g. impressionism), one could build a collection of style embeddings upon which a generative model could be trained. At test time, a style representation would be sampled from the generative model and used in conjunction with the style transfer network to produce a random pastiche of that artistic movement.

In summary, we demonstrated that conditional instance normalization constitutes a simple, efficient and scalable modification of style transfer networks that allows them to model multiple styles at the same time. We showed that despite its simplicity, the method is flexible enough to capture very different styles while having very little impact on training time and final performance of the trained network. Finally, we showed that the learned representation of style is useful in arbitrarily combining artistic styles. This work suggests the existence of a learned representation for artistic styles whose vocabulary is flexible enough to capture a diversity of the painted world.

Acknowledgments

We would like to thank Fred Bertsch, Douglas Eck, Cinjon Resnick and the rest of the Google Magenta team for their feedback; Peyman Milanfar, Michael Elad, Feng Yang, Jon Barron, Bhavik Singh, Jennifer Daniel as well as the the Google Brain team for their crucial suggestions and advice. Finally, we would like to thank the Google Cultural Institute, whose curated collection of art photographs was very helpful in finding exciting style images to train on.

REFERENCES

Alexei A Efros and William T Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 341–346. ACM, 2001.

Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pp. 1033–1038. IEEE, 1999.

Michael Elad and Peyman Milanfar. Style-transfer via texture-synthesis. *arXiv preprint arXiv:1609.03057*, 2016.

Oriel Frigo, Neus Sabater, Julie Delon, and Pierre Hellier. Split and match: Example-based adaptive patch sampling for unsupervised style transfer. 2016.

Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 262–270, 2015a.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015b.

Leon A Gatys, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. Preserving color in neural artistic style transfer. *arXiv preprint arXiv:1606.05897*, 2016.

Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 327–340. ACM, 2001.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *arXiv preprint arXiv:1603.08155*, 2016.

Bela Julesz. Visual pattern discrimination. *IRE Trans. Info Theory*, 8:84–92, 1962.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Vivek Kwatra, Irfan Essa, Aaron Bobick, and Nipun Kwatra. Texture optimization for example-based synthesis. *ACM Transactions on Graphics (ToG)*, 24(3):795–802, 2005.

Lin Liang, Ce Liu, Ying-Qing Xu, Baining Guo, and Heung-Yeung Shum. Real-time texture synthesis by patch-based sampling. *ACM Transactions on Graphics (ToG)*, 20(3):127–150, 2001.

Augustus Odena, Christopher Olah, and Vincent Dumoulin. Avoiding checkerboard artifacts in neural networks. *Distill*, 2016.

Javier Portilla and Eero Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40:49–71, 1999.

Eero Simoncelli and Bruno Olshausen. Natural image statistics and neural representation. *Annual Review of Neuroscience*, 24:1193–1216, 2001.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. *arXiv preprint arXiv:1603.03417*, 2016a.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016b.

Li-Yi Wei and Marc Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 479–488. ACM Press/Addison-Wesley Publishing Co., 2000.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pp. 818–833. Springer, 2014.

APPENDIX

HYPERPARAMETERS

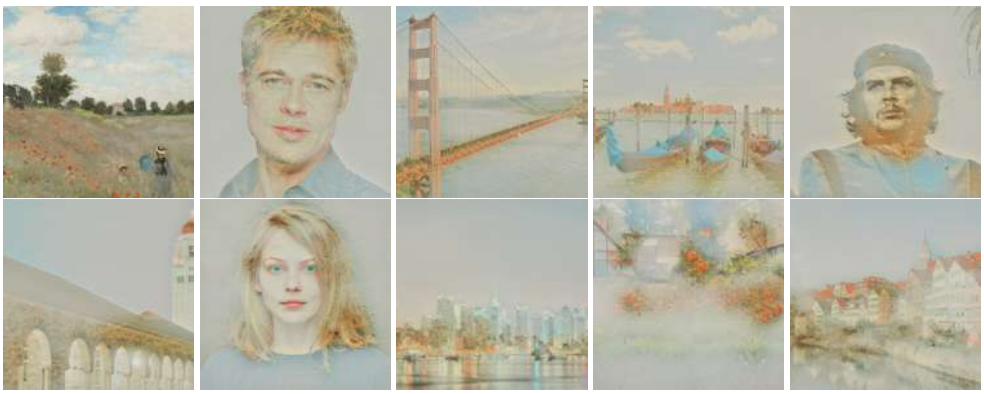
Operation	Kernel size	Stride	Feature maps	Padding	Nonlinearity
Network – $256 \times 256 \times 3$ input					
Convolution	9	1	32	SAME	ReLU
Convolution	3	2	64	SAME	ReLU
Convolution	3	2	128	SAME	ReLU
Residual block			128		
Residual block			128		
Residual block			128		
Residual block			128		
Residual block			128		
Upsampling			64		
Upsampling			32		
Convolution	9	1	3	SAME	Sigmoid
Residual block – C feature maps					
Convolution	3	1	C	SAME	ReLU
Convolution	3	1	C	SAME	Linear
	Add the input and the output				
Upsampling – C feature maps					
	Nearest-neighbor interpolation, factor 2				
Convolution	3	1	C	SAME	ReLU
Padding mode	REFLECT				
Normalization	Conditional instance normalization after every convolution				
Optimizer	Adam (Kingma & Ba, 2014) ($\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$)				
Parameter updates	40,000				
Batch size	16				
Weight initialization	Isotropic gaussian ($\mu = 0$, $\sigma = 0.01$)				

Table 1: Style transfer network hyperparameters.

MONET PASTICHES

Claude Monet, *Grainstacks at Giverny; the Evening Sun* (1888/1889).

Claude Monet, *Plum Trees in Blossom* (1879).



Claude Monet, *Poppy Field* (1873).

Claude Monet, *Rouen Cathedral, West Façade* (1894).

Claude Monet, *Sunrise (Marine)* (1873).

Claude Monet, *The Road to Vétheuil* (1879).

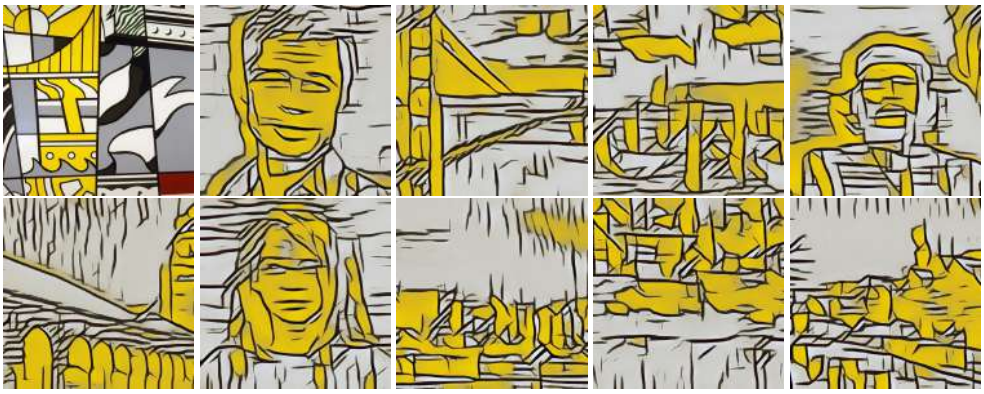
Claude Monet, *Three Fishing Boats* (1886).

Claude Monet, *Vétheuil* (1879).

Claude Monet, *Vétheuil* (1902).

Claude Monet, *Water Lilies* (ca. 1914-1917).

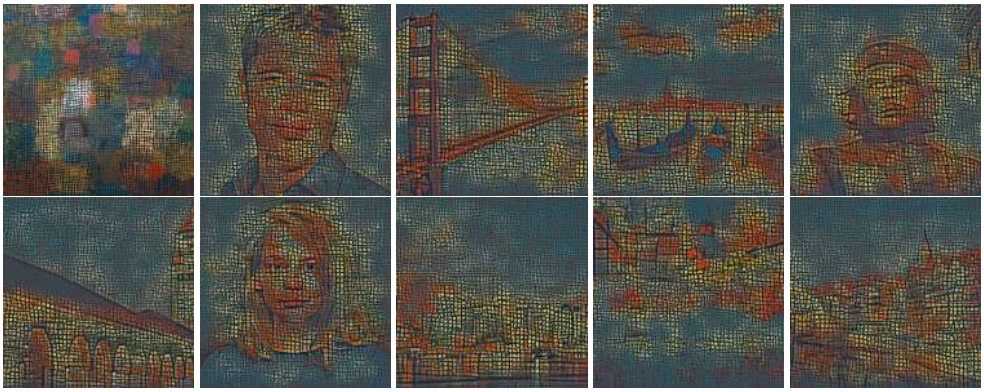
VARIED PASTICHES



Roy Lichtenstein, *Bicentennial Print* (1975).

Ernst Ludwig Kirchner, *Boy with Sweets* (1918).

Paul Signac, *Cassis, Cap Lombard, Opus 196* (1889).

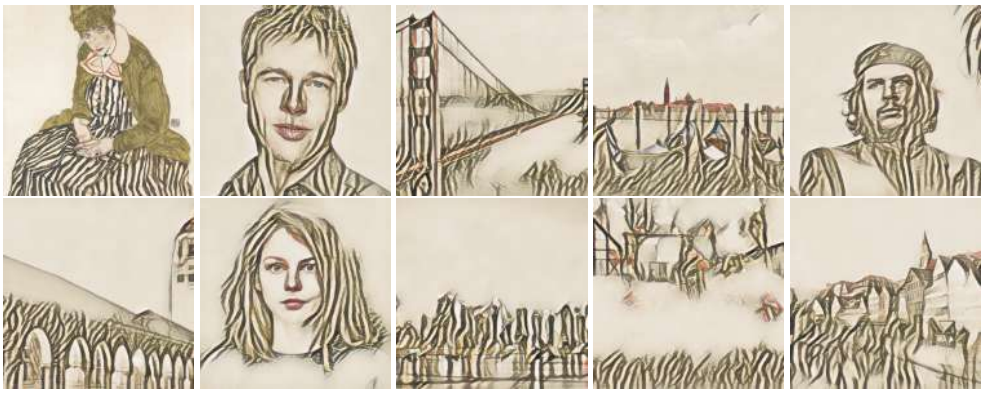


Paul Klee, *Colors from a Distance* (1932).

Frederic Edwin Church, *Cotopaxi* (1855).

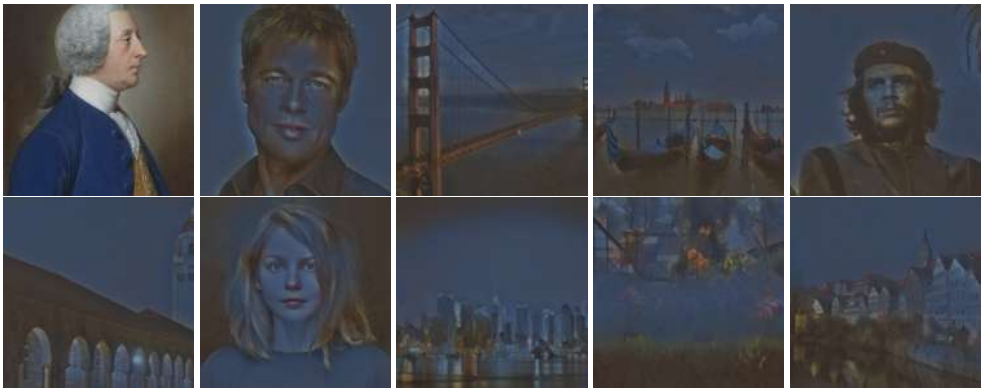
Jamini Roy, *Crucifixion*.

Henri de Toulouse-Lautrec, *Divan Japonais* (1893).



Egon Schiele, *Edith with Striped Dress, Sitting* (1915).

Georges Rouault, *Head of a Clown* (ca. 1907-1908).



William Hoare, *Henry Hoare, "The Magnificent", of Stourhead* (about 1750-1760).

Giorgio de Chirico, *Horses on the seashore* (1927/1928).

Vincent van Gogh, *Landscape at Saint-Rémy (Enclosed Field with Peasant)* (1889).

Nicolas Poussin, *Landscape with a Calm* (1650-1651).

Bernardino Fungai, *Madonna and Child with Two Hermit Saints* (early 1480s).

Max Hermann Maxy, *Portrait of a Friend* (1926).

Juan Gris, *Portrait of Pablo Picasso* (1912).

Severini Gino, *Ritmo plastico del 14 luglio* (1913).

Richard Diebenkorn, *Seawall* (1957).

Alice Bailly, *Self-Portrait* (1917).

Grayson Perry, *The Annunciation of the Virgin Deal* (2012).

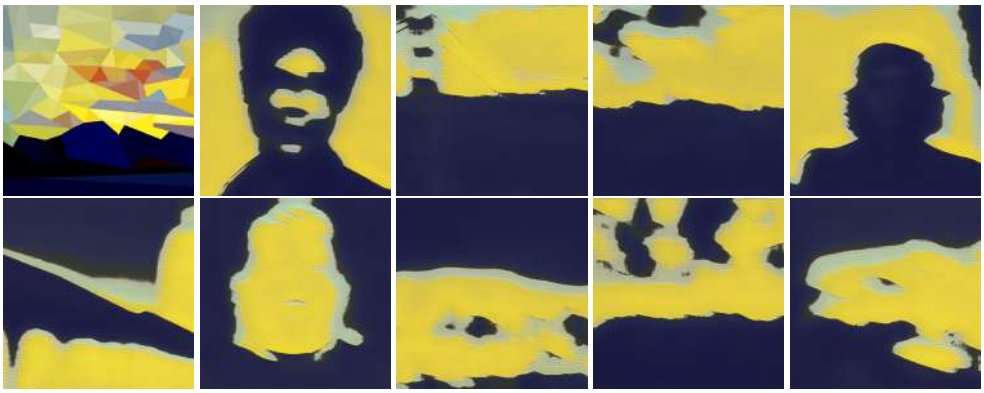
William Glackens, *The Green Boathouse* (ca. 1922).

Edvard Munch, *The Scream* (1910).

Vincent van Gogh, *The Starry Night* (1889).

Pieter Bruegel the Elder, *The Tower of Babel* (1563).

Wolfgang Lettl, *The Trial* (1981).



Douglas Coupland, *Thomson No. 5 (Yellow Sunset)* (2011).

Claude Monet, *Three Fishing Boats* (1886).

John Ruskin, *Trees in a Lane* (1847).

Giuseppe Cades, *Tullia about to Ride over the Body of Her Father in Her Chariot* (about 1770-1775).

Berthe Morisot, *Under the Orange Tree* (1889).

Giulio Romano (Giulio Pippi), *Victory, Janus, Chronos and Gaea* (about 1532-1534).

Wassily Kandinsky, *White Zig Zags* (1922).