Fast Patch-based Style Transfer of Arbitrary Style

Tian Qi Chen and Mark Schmidt
University of British Columbia
Artistic Style Transfer

Combining a picture with Vincent van Gogh’s *The Starry Night*:
Artistic Style Transfer

Combining a picture with Vincent van Gogh’s *The Starry Night*:

A painting takes days to complete.
Can a computer be used to transfer the style of a painting onto another image?
Visual Quality comes from use of Convolutional Neural Nets

Processing at pixel level $\rightarrow$ Processing at the activations level

Success in visual quality has created a market for mobile and web applications.

Gatys et al. (2015), Li and Wand (2016), Ulyanov et al. (2016), Johnson et al. (2016), Dumoulin et al. (2016)
How to define “style transfer”?

$$\arg\min_{\mathcal{I}} L(\mathcal{I}, \text{Content}, \text{Style})$$

Requires hundreds of forward and backward passes through the CNN.

**Slow**

e.g. Gatys et al. (2015), Li and Wand (2016)
Existing Approaches: Feedforward Style Network

\[ \min_\theta L(StyleNet(\text{Content}; \theta), \text{Content}, \text{Style}) \]

\[ StyleNet(\text{Content}; \theta^*) = I \]

Train a neural network to approximate the optimization result.

Limited in Style

e.g. Ulyanov et al. (2016), Johnson et al. (2016), Dumoulin et al. (2016)
Motivation

Existing optimization-based approaches:

adaptable to any style image but slow

Existing feedforward approaches:

fast but limited

We present an approach that is:

feedforward, fast, and adaptable to any style image
Our Approach

- restrict to the use of just a single layer
- directly construct target activations
- an inverse network that is not style-dependent

Differences from existing works:
- Decoupling of the style transfer process and image generation
- Constructive procedure instead of defining style transfer as an optimization
Our Approach

- restrict to the use of just a single layer
Our Approach

- restrict to the use of just a single layer
- directly construct target activations
Our Approach

- restrict to the use of just a single layer
- directly construct target activations
- an inverse network that is not style-dependent
Our Approach

- restrict to the use of just a single layer
- directly construct target activations
- an inverse network that is not style-dependent

Differences from existing works:

- Decoupling of the style transfer process and image generation
- Constructive procedure instead of defining style transfer as an optimization
Patch-based Replacement (Style Swap)

For every content patch, swap it with the best matching style patch.
Patch-based Replacement (Style Swap)

For every content patch, swap it with the best matching style patch.

$$\text{BestMatch}(c) = \arg \max_{s \in S} \frac{\langle c, s \rangle}{\|c\| \cdot \|s\|}$$

*(Li and Wand (2016) uses this measure, but inside an optimization procedure.)*
Patch-based Replacement (Style Swap)

For every content patch, swap it with the best matching style patch.

$$\text{BestMatch}(c) = \arg\max_{s \in S} \frac{\langle c, s \rangle}{\|c\| \cdot \|s\|}$$

*(Li and Wand (2016) uses this measure, but inside an optimization procedure.)*
Patch-based Replacement (Style Swap)

For every content patch, swap it with the best matching style patch.  

$$\text{BestMatch}(c) = \arg \max_{s \in S} \frac{\langle c, s \rangle}{||c|| \cdot ||s||}$$

Other names for the transposed convolution: fractionally-strided convolution, backward convolution, upconvolution, or “deconvolution”.

2D Convolution With Normalized Style Patches as Filters

Channel-wise Argmax

2D Transposed Convolution With Style Patches as Filters
Properties of Style Swap: RGB vs. Activations

Content

Style

Style Swap RGB

Style Swap Activations
Properties of Style Swap: Intuitive Tuning Parameter

Control level of abstraction using a single discrete parameter:

patch size.

3 × 3 Patches    7 × 7 Patches    12 × 12 Patches
Inverting Activations: Option I

Pretrained CNN

Style Swap
Inverting Activations: Option I

Pretrained CNN

Style Swap
We do unsupervised training of the inverse network. We train using 80,000 photos \(^1\) as content and 80,000 paintings \(^2\) as style.

\(^1\) Microsoft COCO
\(^2\) Painter by Numbers (public; hosted on kaggle.com)
We do unsupervised training of the inverse network. We train using 80,000 photos\(^1\) as content and 80,000 paintings\(^2\) as style.

\(^1\)Microsoft COCO  
\(^2\)Painter by Numbers (public; hosted on kaggle.com)
Comparison with existing methods that can handle arbitrary style images:

<table>
<thead>
<tr>
<th>Method</th>
<th>N. Iters.</th>
<th>Time/Iter. (s)</th>
<th>Total (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gatys et al.</td>
<td>500</td>
<td>0.1004</td>
<td>50.20</td>
</tr>
<tr>
<td>Li and Wand</td>
<td>200</td>
<td>0.6293</td>
<td>125.86</td>
</tr>
<tr>
<td><strong>Style Swap</strong> (Optim)</td>
<td>100</td>
<td>0.0466</td>
<td>4.66</td>
</tr>
<tr>
<td><strong>Style Swap</strong> (InvNet)</td>
<td>1</td>
<td>1.2483</td>
<td>1.25</td>
</tr>
</tbody>
</table>

**Table 1**: Computation time on $500 \times 300$ size images.
Comparison with existing methods that can handle arbitrary style images:

<table>
<thead>
<tr>
<th>Method</th>
<th>N. Iters.</th>
<th>Time/Iter. (s)</th>
<th>Total (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gatys et al.</td>
<td>500</td>
<td>0.1004</td>
<td>50.20</td>
</tr>
<tr>
<td>Li and Wand</td>
<td>200</td>
<td>0.6293</td>
<td>125.86</td>
</tr>
<tr>
<td><strong>Style Swap</strong> (Optim)</td>
<td>100</td>
<td>0.0466</td>
<td>4.66</td>
</tr>
<tr>
<td><strong>Style Swap</strong> (InvNet)</td>
<td>1</td>
<td>1.2483</td>
<td>1.25</td>
</tr>
</tbody>
</table>

Table 1: Computation time on 500 × 300 size images.

- Computation time for our method can be significantly reduced if number of style patches is reduced.
- Can scale to large content sizes if style image is kept at a manageable size.
Empirically, we observe:

- Similar images $\rightarrow$ similar style transferred results.
- Consecutive frames of a video $\rightarrow$ consistent results.
Frame-by-frame Application of Our Method

Timelapse Video of Vancouver, BC

Original video credit to TimeLapseHD.
We present the first feedforward method for style transfer that can adapt to arbitrary style.

Our method for style transfer has the following properties:

- Tunable with a discrete intuitive tuning parameter
- Consistent and allows frame-by-frame application to videos
- Gives a degree of control over the style transfer result
Fast Patch-based Style Transfer of Arbitrary Style
Tian Qi Chen and Mark Schmidt

source code: github.com/rtqichen/style-swap