Fast Patch-based Style Transfer of Arbitrary Style

Tian Qi Chen and Mark Schmidt

University of British Columbia

Content

Style

Our Method

Our Fast Approx.

Artistic Style Transfer

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

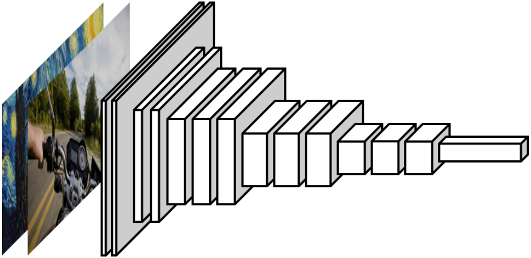
Artistic Style Transfer

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 ⟶ 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)

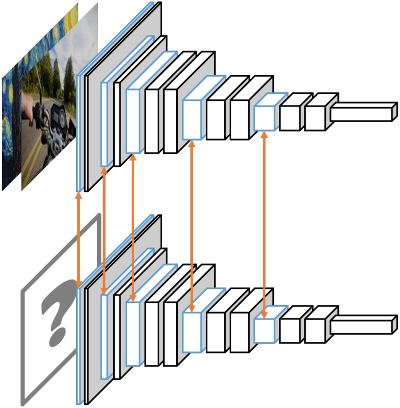
Existing Approaches: Optimization-based

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)

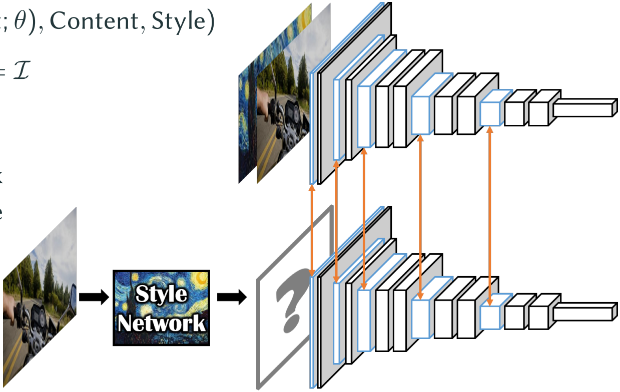
Existing Approaches: Feedforward Style Network

$$\min_{\theta} L(\mathit{StyleNet}(\text{Content}; \theta), \text{Content}, \text{Style})$$

$$\mathit{StyleNet}(\text{Content}; \theta^*) = \mathcal{I}$$

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

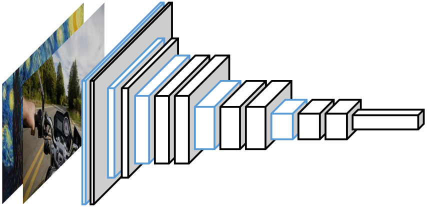
Existing feedforward approaches:

fast but limited

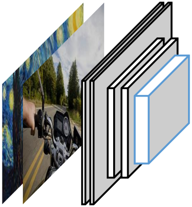
We present an approach that is:

feedforward, fast, and adaptable to any style image

Our Approach

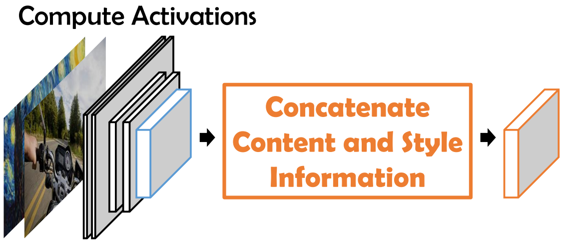


Our Approach



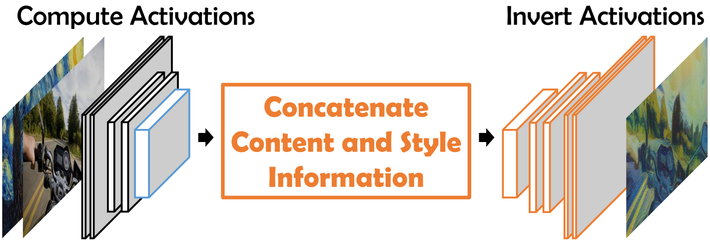
- restrict to the use of just a single layer

Our Approach



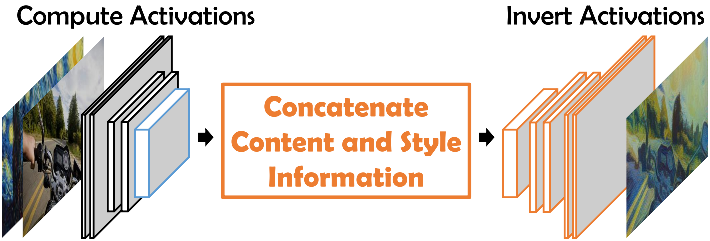
- restrict to the use of just a single layer
- directly construct target activations

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



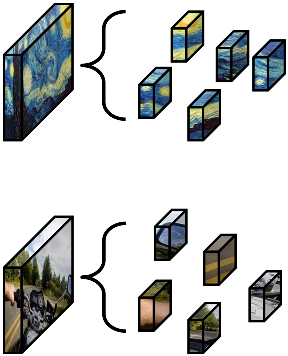
- restrict to the use of just a single layer
- directly construct target activations
- an inverse network that is not style-dependent

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)

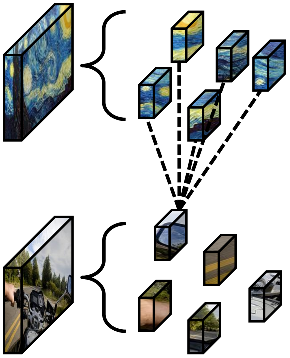
For every content patch, swap it with the best matching style patch.



Patch-based Replacement (Style Swap)

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

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

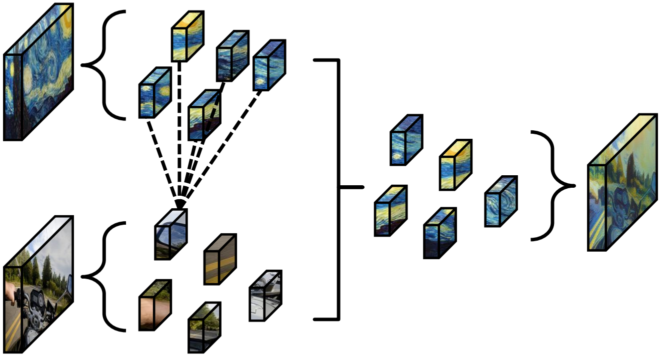


*(Li and Wand (2016) uses this measure, but inside an optimization procedure.)

Patch-based Replacement (Style Swap)

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

$$\mathrm{BestMatch}(c) = \underset{s \in S}{\arg\max} \frac{\langle c, s \rangle}{||c|| \cdot ||s||}$$

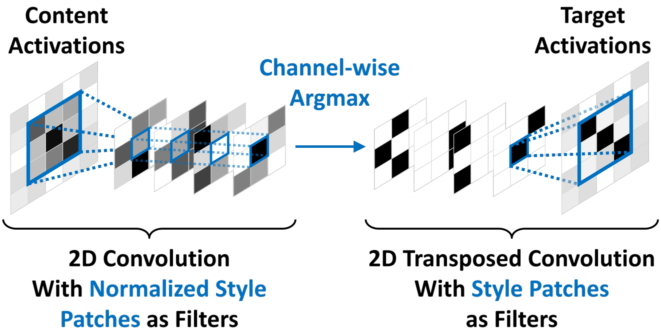


*(Li and Wand (2016) uses this measure, but inside an optimization procedure.)

Patch-based Replacement (Style Swap)

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

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



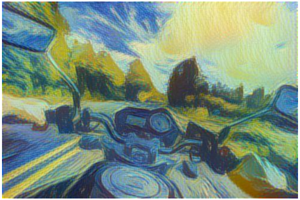
Other names for the transposed convolution: fractionally-strided convolution, backward convolution, upconvolution, or “deconvolution”.

Properties of Style Swap: RGB vs. Activations

Content

Style

Style Swap RGB

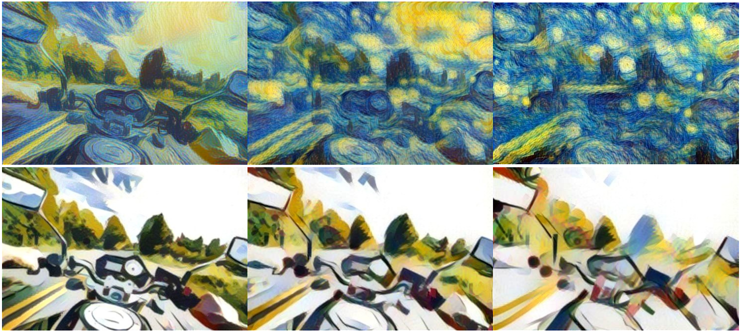


Style Swap Activations

Properties of Style Swap: Intuitive Tuning Parameter

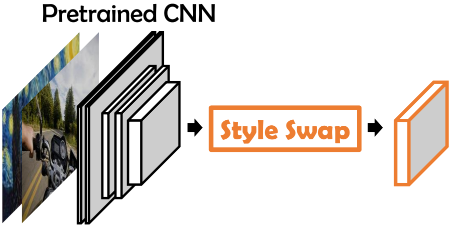
Control level of abstraction using a single discrete parameter:

patch size.

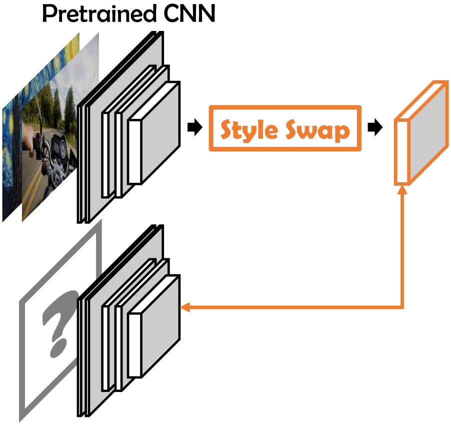


3 × 3 Patches 7 × 7 Patches 12 × 12 Patches

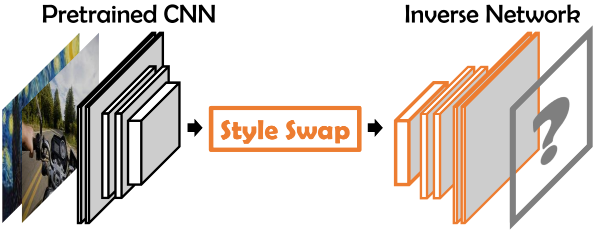
Inverting Activations: Option I



Inverting Activations: Option I

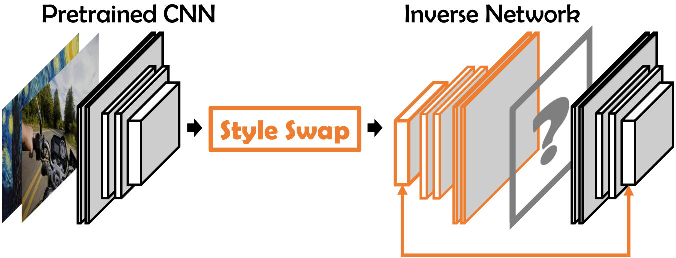


Inverting Activations: Option II



[1] Microsoft COCO
[2] Painter by Numbers (public; hosted on kaggle.com)

Inverting Activations: Option II



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)

Computation Time

Comparison with existing methods that can handle arbitary style images:

Method	N. Iters.	Time/Iter. (s)	Total (s)
Gatys et al.	500	0.1004	50.20
Li and Wand	200	0.6293	125.86
Style Swap (Optim)	100	0.0466	4.66
Style Swap (InvNet)	1	1.2483	1.25

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

Computation Time

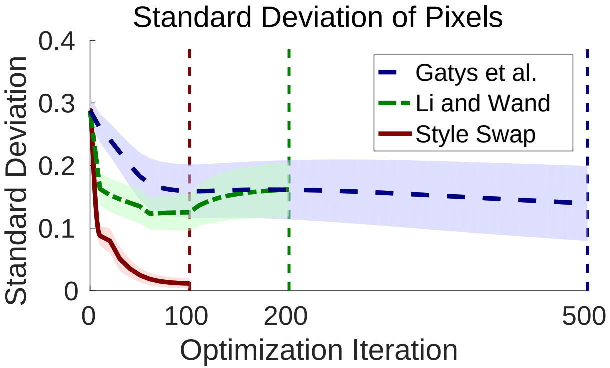
Comparison with existing methods that can handle arbitary style images:

Method	N. Iters.	Time/Iter. (s)	Total (s)
Gatys et al.	500	0.1004	50.20
Li and Wand	200	0.6293	125.86
Style Swap (Optim)	100	0.0466	4.66
Style Swap (InvNet)	1	1.2483	1.25

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.

Consistency: Few Local Optima

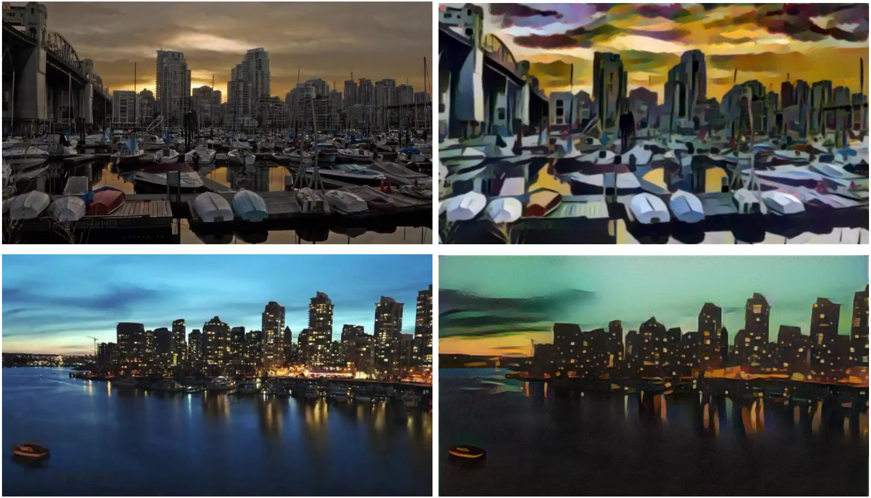


Empirically, we observe:

- 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.

Summary

- 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