Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization

Xun Huang Serge Belongie
Department of Computer Science & Cornell Tech, Cornell University

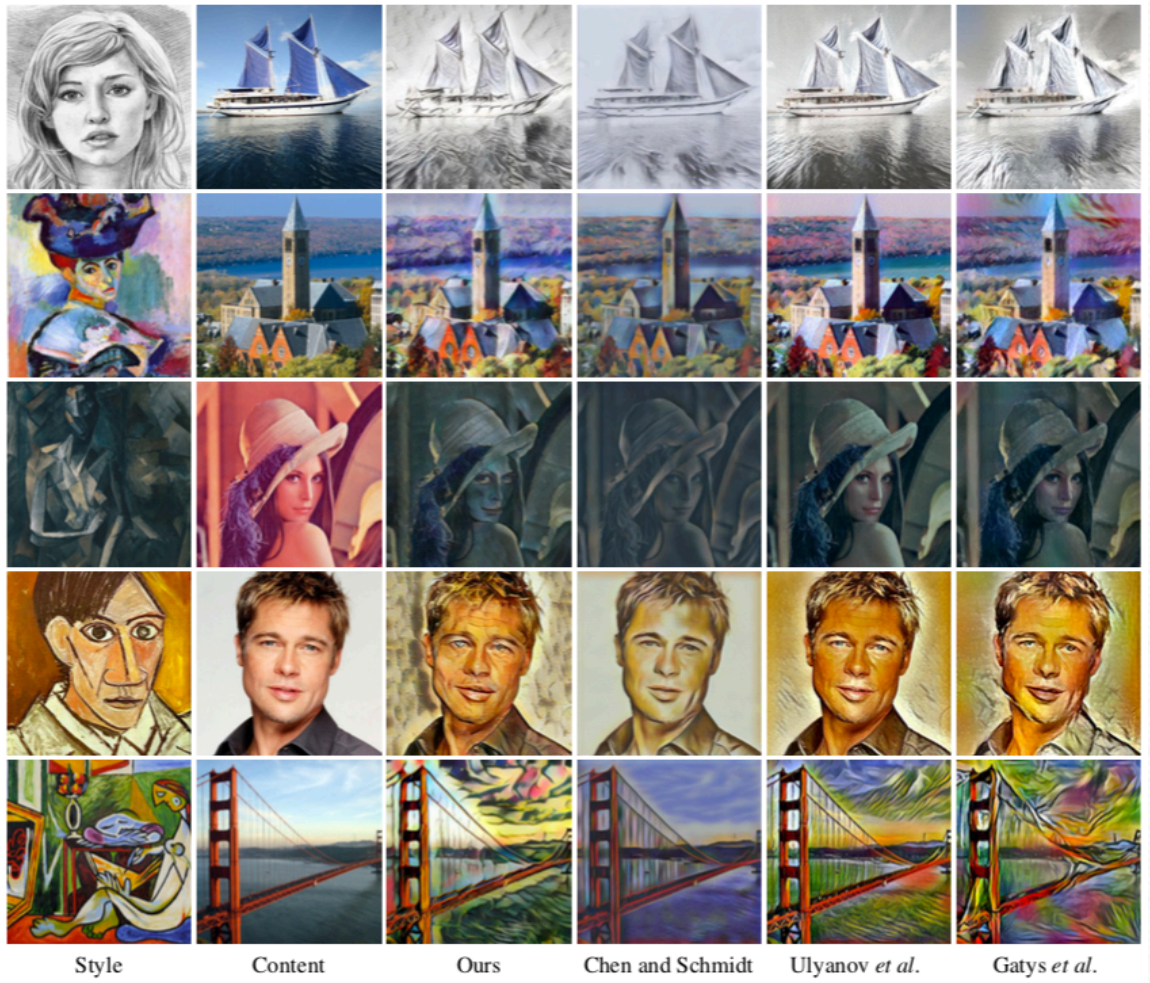
——in ICCV, 2017

Li Na

Example style transfer results.

The key point

1. Keep the content same
2. Style transfer



Existing problems

- Gatys et al[1] framework requires a slow iterative optimization process, which limits its practical application.
- Fast approximations with feed-forward neural networks:
  - the speed improvement comes at a cost;
  - the network is usually tied to a fixed set of styles and cannot adapt to arbitrary new styles.

- [1]. Gatys L A, Ecker A S, Bethge M. Image style transfer using convolutional neural networks (CVPR), 2016

The innovation points

- A simple yet effective approach that for the first time enables arbitrary style transfer in real-time.
- A novel adaptive instance normalization (AdaIN) layer that aligns the mean and variance of the content features with those of the style features.

Background

- Batch Normalization
- Given an input batch $x \in \mathbb{R}^{N \times C \times H \times W}$
- BN normalizes the mean and standard deviation for each individual feature channel

$$\mathbf{BN}(x) = \gamma \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \beta$$

$$\mu_c(x) = \frac{1}{NHW} \sum_{n=1}^{N} \sum_{h=1}^{H} \sum_{w=1}^{W} x_{nchw}$$

$$\sigma_c(x) = \sqrt{\frac{1}{NHW} \sum_{n=1}^{N} \sum_{h=1}^{H} \sum_{w=1}^{W} (x_{nchw} - \mu_c(x))^2 + \epsilon}$$

Background

- Instance Normalization

$$\mathbf{IN}(x) = \gamma\left(\frac{x - \mu(x)}{\sigma(x)}\right) + \beta$$

$$\mu_{nc}(x) = \frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W} x_{nchw}$$

$$\sigma_{nc}(x) = \sqrt{\frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W}(x_{nchw} - \mu_{nc}(x))^2 + \epsilon}$$

Background

- Conditional Instance Normalization

$$\mathbf{CIN}(x; s) = \gamma^s \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \beta^s$$

- The network can generate images in completely different styles by using the same convolutional parameters but different affine parameters in IN layers.
- Compared with a network without normalization layers, a network with CIN layers requires 2*feature maps*styles parameters.
- Since the number of additional parameters scales linearly with the number of styles, it is challenging to extend their method to model a large number of styles.

Method

- Adaptive Instance Normalization

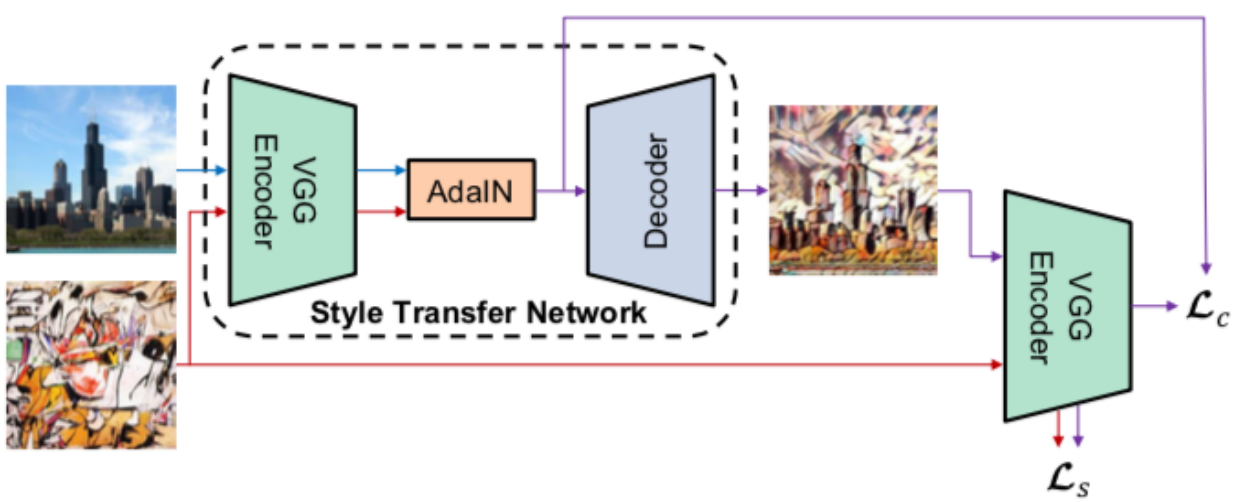
$$\mathbf{AdaIN}(x, y) = \sigma(y)\left(\frac{x - \mu(x)}{\sigma(x)}\right) + \mu(y)$$

- in which we simply scale the normalized content input with σ(y), and shift it with μ(y). Similar to IN, these statistics are computed across spatial locations.

Network Architecture

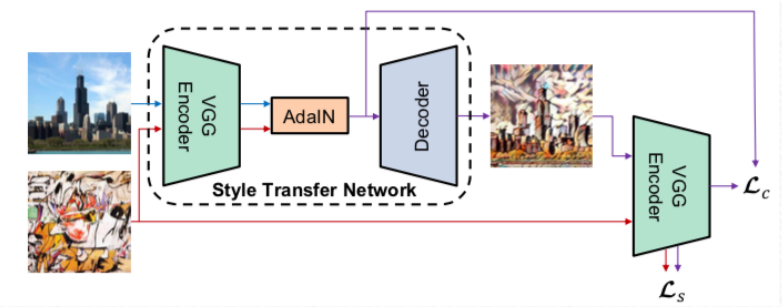
$$t = \text{AdaIN}(f(c), f(s))$$

$$T(c, s) = g(t)$$



- The decoder mostly mirrors the encoder.
- IN normalizes each sample to a single style while BN normalizes a batch of samples to be centered around a single style. Both are undesirable when we want the decoder to generate images in vastly different styles. Thus, we do *not* use normalization layers in the decoder.

Loss



$$\mathcal{L} = \mathcal{L}_c + \lambda \mathcal{L}_s$$

$$\mathcal{L}_c = \|f(g(t)) - t\|_2$$

$$\mathcal{L}_s = \sum_{i=1}^{L} \|\mu(\phi_i(g(t))) - \mu(\phi_i(s))\|_2 \quad + \quad \sum_{i=1}^{L} \|\sigma(\phi_i(g(t))) - \sigma(\phi_i(s))\|_2$$

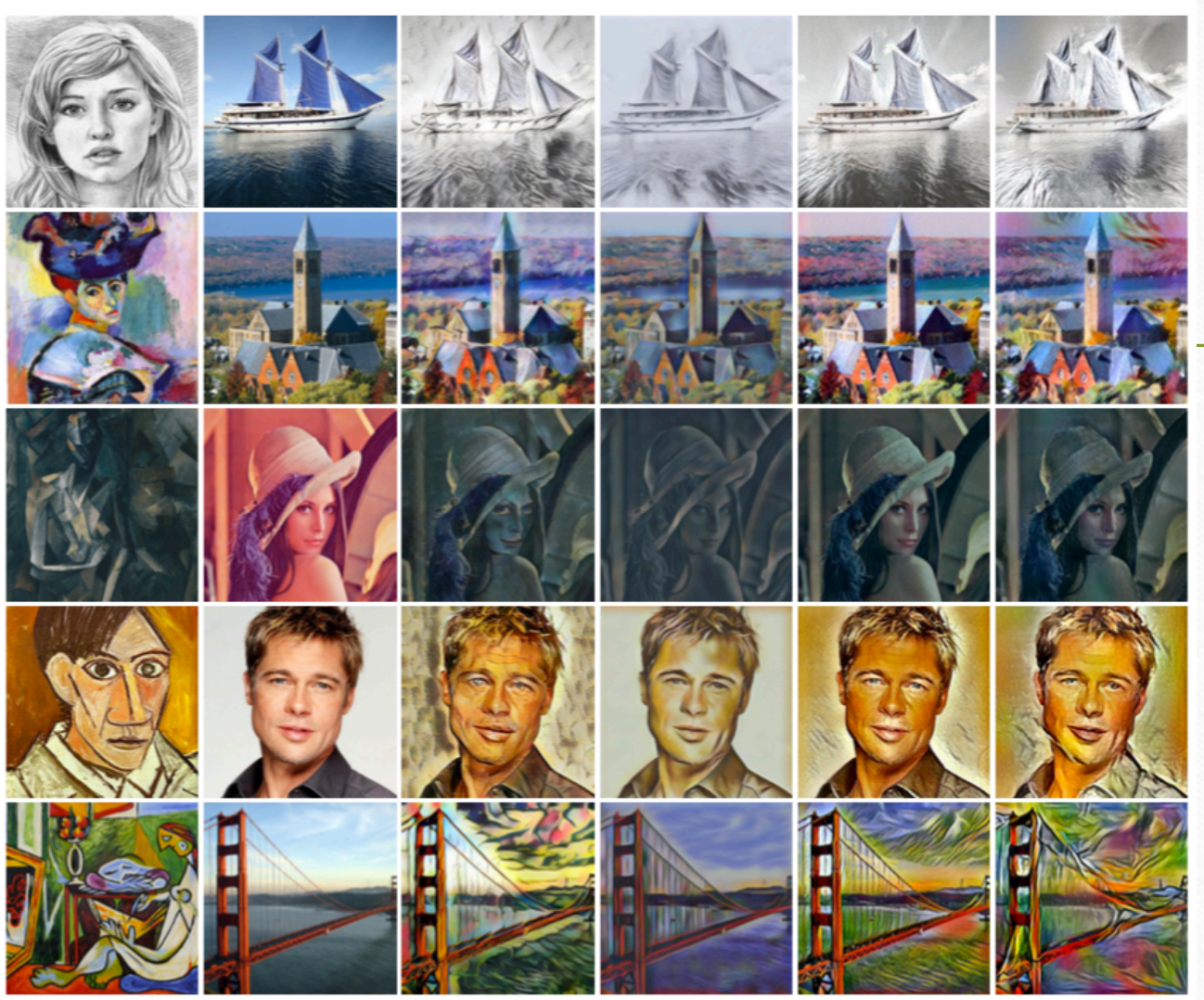
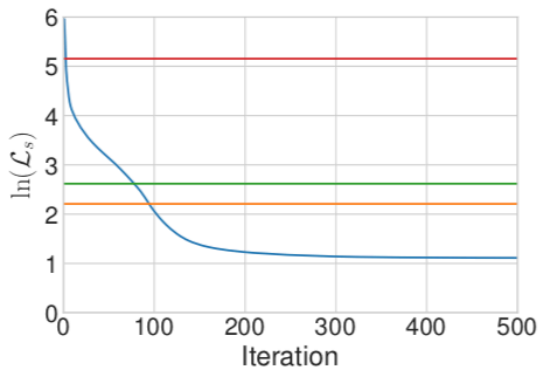


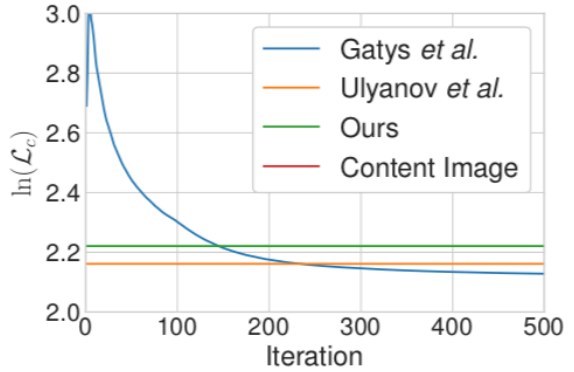
Figure 4. Example style transfer results. All the tested content and style images are never observed by our network during training.

Quantitative evaluations.

Method	Time (256px)	Time (512px)	# Styles
Gatys *et al.*	14.17 (14.19)	46.75 (46.79)	∞
Chen and Schmidt	0.171 (0.407)	3.214 (4.144)	∞
Ulyanov *et al.*	**0.011** (N/A)	**0.038** (N/A)	1
Dumoulin *et al.*	**0.011** (N/A)	**0.038** (N/A)	32
Ours	**0.018** (0.027)	**0.065** (0.098)	∞



(a) Style Loss



(b) Content Loss

Q&A