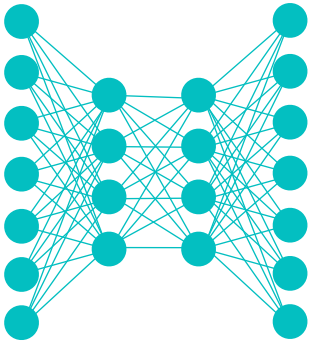
Lecture Notes for

Neural Networks and Machine Learning

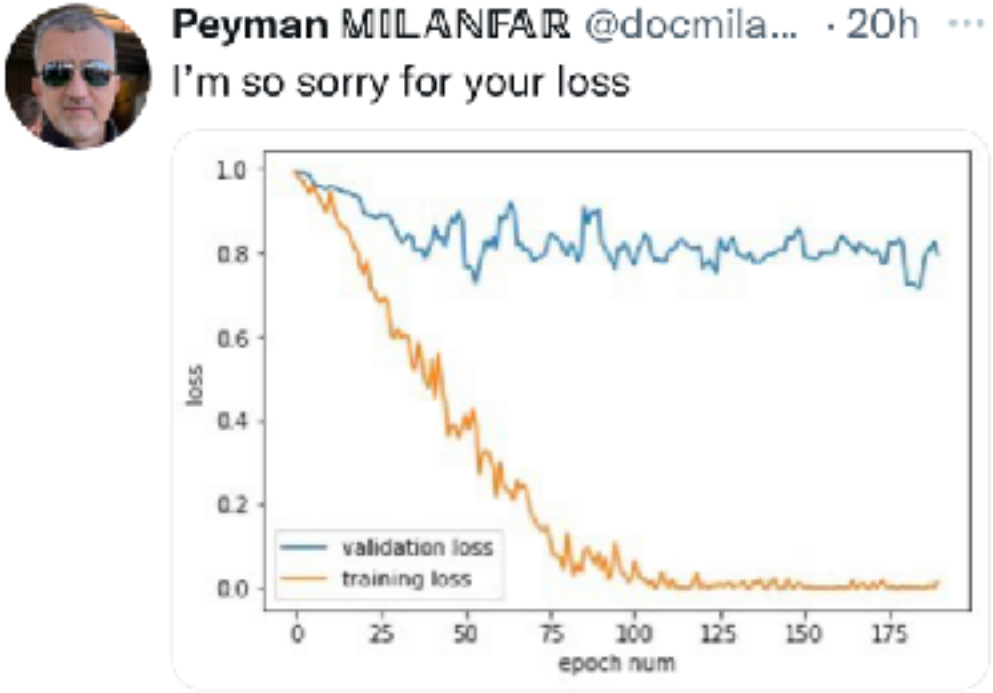
Neural Style Transfer



Logistics and Agenda

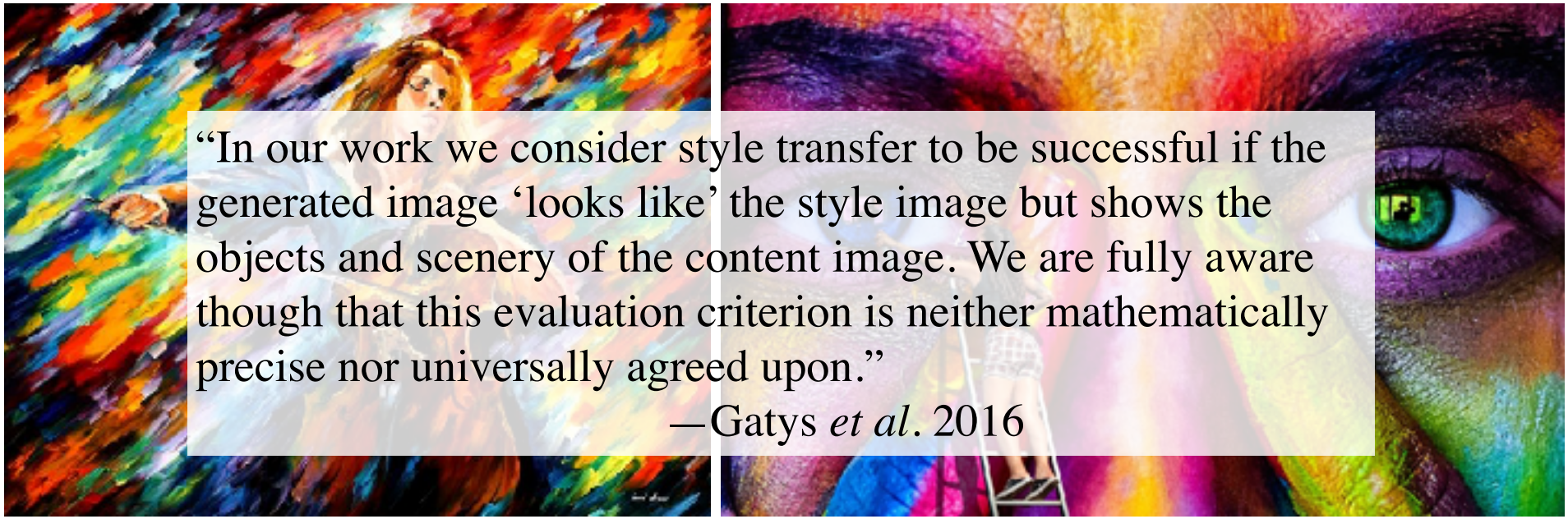
- Logistics
  - Finish final paper discussion
  - Bonus content: style transfer
- Agenda
  - A History of Style Transfer (today)
  - Image Optimization Algorithms (today)
  - Model Optimization Algorithms
  - One Shot Algorithms
  - Evaluating Style Transfer Performance
  - Extensions in Other Domains

Style Transfer: A History



The Premise

- "Style" can be transferred from one domain to another
- While preserving the "content" of an image
- Style and content are in the eye of the beholder
- Can you define style versus content?
- Do you know it when you see it?

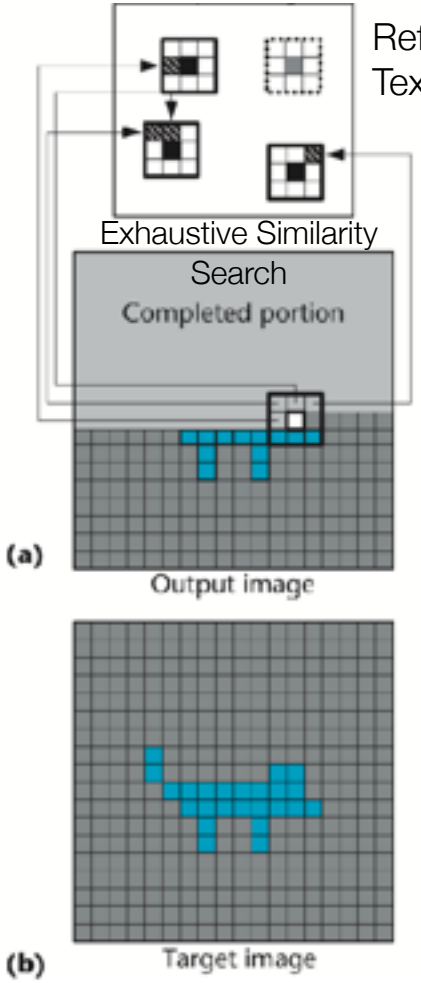


"In our work we consider style transfer to be successful if the generated image 'looks like' the style image but shows the objects and scenery of the content image. We are fully aware though that this evaluation criterion is neither mathematically precise nor universally agreed upon."

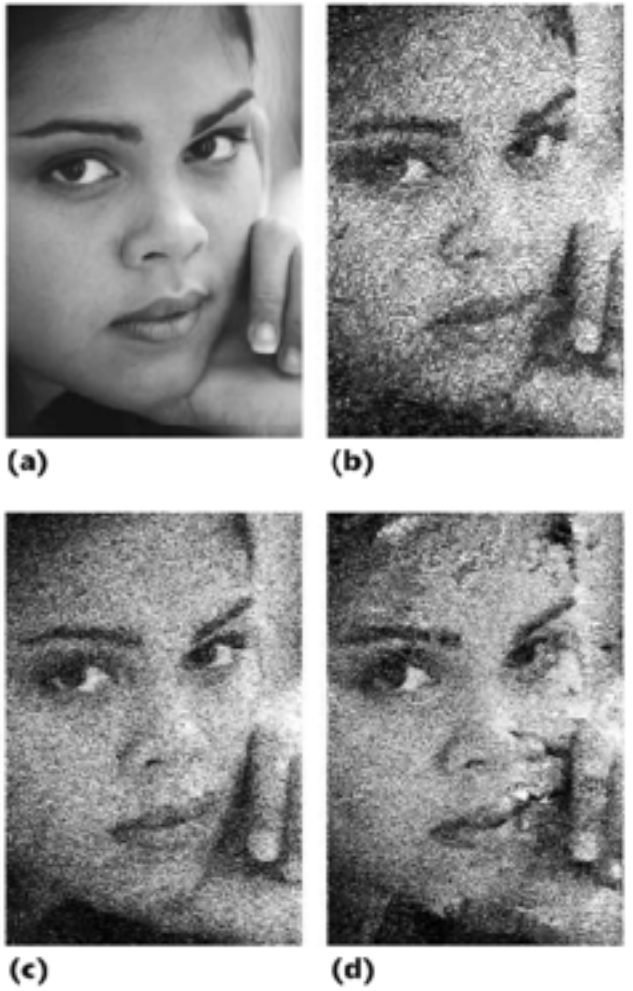
—Gatys *et al*. 2016

Before Neural Networks

- Ashekhmin, Fast Texture Transfer, 2003



1 (a) Each pixel in this L-shaped neighborhood generates a shifted candidate pixel (black) according to its original position in the input image (hatched). A single random candidate (light blue with dashed lines) is added with probability *p*. The candidate whose neighborhood best matches the one in the output image according to an application-specific similarity metric is chosen as the next pixel value. In the original algorithm, the complete neighborhood for matching is composed from two L-shaped halves, with top half coming from the already synthesized part of the output image and (b) the bottom half from the target image.



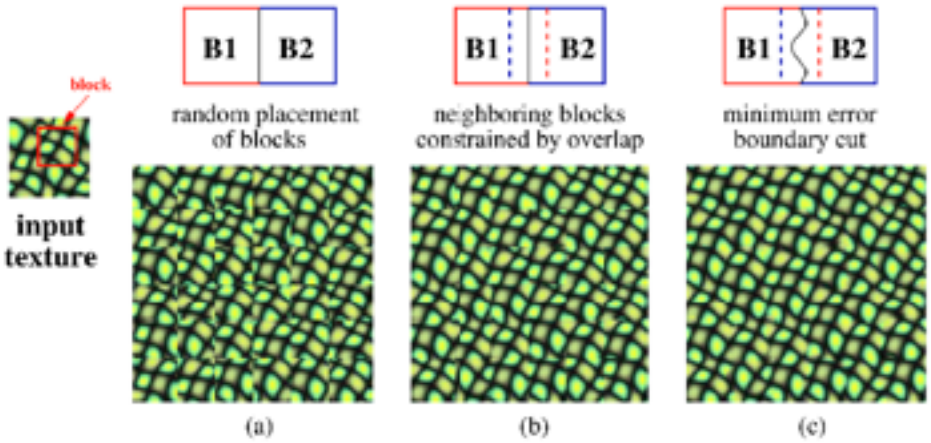
Before Neural Networks

- Efros and Freeman, 2001

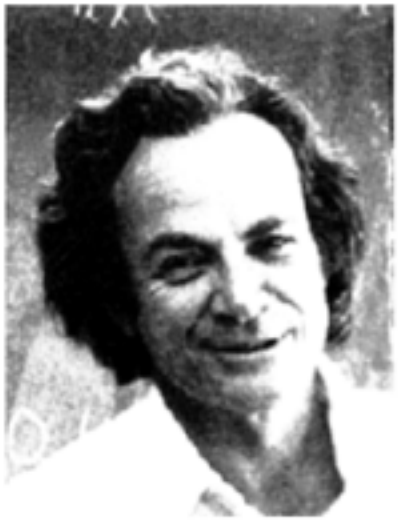
2.2 The Image Quilting Algorithm

The complete quilting algorithm is as follows:

- Go through the image to be synthesized in raster scan order in steps of one block (minus the overlap).
- For every location, search the input texture for a set of blocks that satisfy the overlap constraints (above and left) within some error tolerance. Randomly pick one such block.
- Compute the error surface between the newly chosen block and the old blocks at the overlap region. Find the minimum cost path along this surface and make that the boundary of the new block. Paste the block onto the texture. Repeat.



source texture



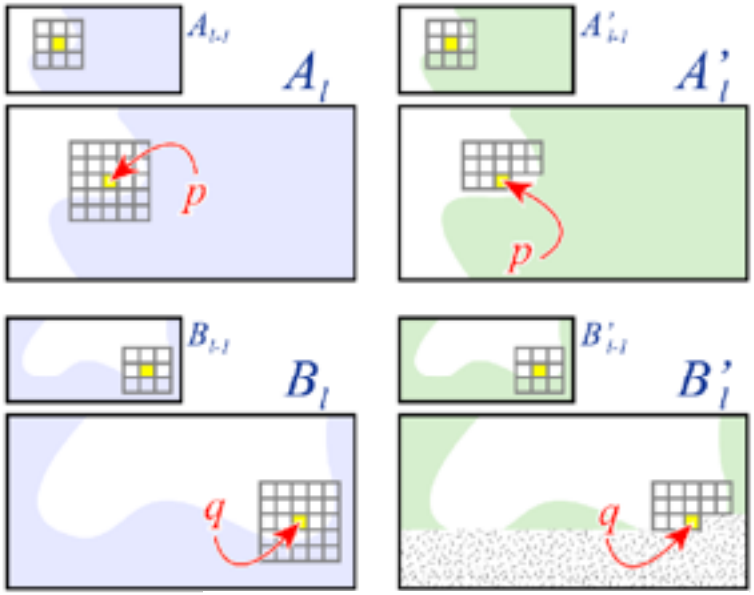
target image

correspondence maps

texture transfer result

Before Neural Networks

- Hertzman *et al.*, 2001
- Image analogies, *A is to A' as B is to "?"*



Input Analogy

Output

Early methods: so many downsides

- Exhaustive patch wise image searches
  - not suitable for any real time processing,
  - took tens of minutes and hours in early 2000's
- Analogy required existing style transfer examples
  - Typically brittle to new types of images
  - …and images without structures in the original analogy
- Research field was dormant for about a decade
- Until, 2016! Convolutional Neural Networks are found to have "Information Distillation Pipeline"

Neural Methods

- 2016 early: Gatys, Ecker, Bethge, Original Paper, Use CNNs instead of patch wise searches to separate style and content **Usually Best Results**
- 2016 mid: Johnson, Alahi, and Fei-Fei. Define loss through neural network as “perceptual” **Usually Fastest Results**
- 2016 late: Li and Wand: GANs for translating style, slightly better results than perceptual loss
- 2017: Lots of small improvement papers based on loss function and normalization tricks
- 2017 late: Li, Fang, Yang, Wang, Lu, and Yang,
  One shot style transfer: no training methods for transferring infinite styles
  Best Quality/Time Tradeoff,
  Generally Impressive that it Even Works!
- 2018 mid: Li, Liu, Li, Yang, Kautz: Photo realistic one shot transfer

Image Optimization Based Style Transfer

In the Beginning, there was Gatys…

- How to define content and style with CNNs?
- Use Pre-trained network like VGG.
- Content:

$$\mathscr{L}_c(I_c, I_{new}) = \sum_{l \in L_c} \lambda_l \cdot \|A_c^{(l)} - A_{new}^{(l)}\|^2$$

Content Layers

Vectorized Activations
(Conv Layer Outputs)

- Style:

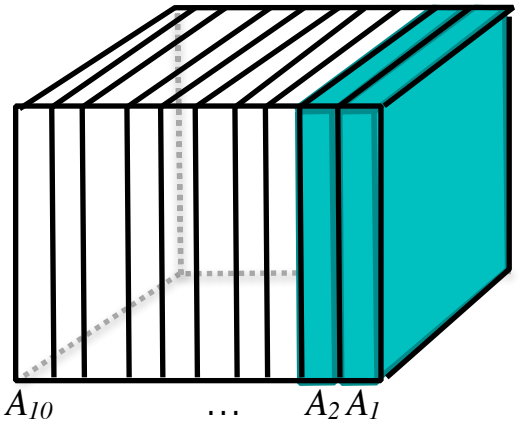
$$\mathscr{L}_s(I_s, I_{new}) = \sum_{l \in L_s} \beta_l \cdot \|G_s^{(l)} - G_{new}^{(l)}\|^2$$

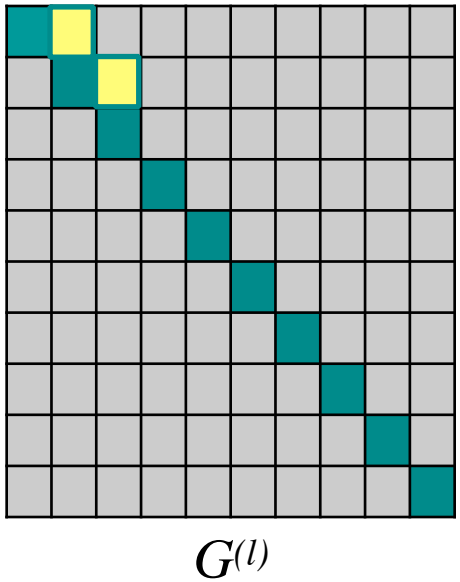
Grammian of Each
$(l)^{th}$ Activation Tensor

$$G_{i,j}^{(l)} = \sum A_i^{(l)} \cdot A_j^{(l)}$$

Into the Grammian… Inner Product

- Grammian is a square matrix, defining covariance among filter activations:





So this is the non-normalized "covariance" among the different filters, where spatial information is aggregated away.

We reduce "style" to the correlated responses of filters in a specific layer.

$$G_{i,j}^{(l)} = \sum A_i^{(l)} \cdot A_j^{(l)}$$

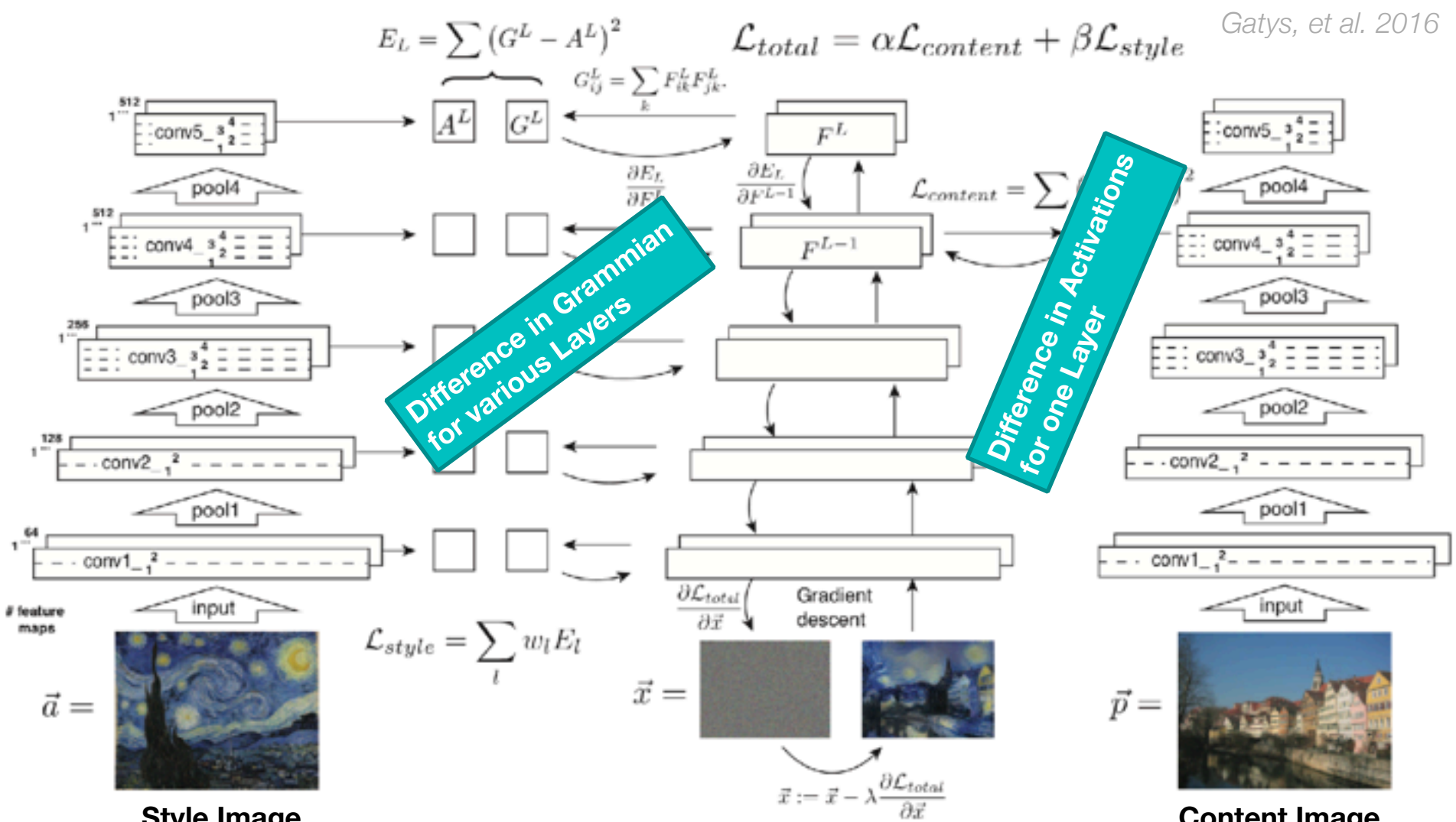
```
# Easy to Implement
Av =  A.reshape( (channels, rows*cols) )
G  =  Av @ Av.T
```


Is that really style?

- No.
- But, the vision system and brain are complex.
- The vision system does classify texture through correlated responses of cortical cells… So we are approximating correlation between neurons in the vision system…
  - …at best, this is an argument about inspiration, not explanatory
  - but it is independent of the content, or at least not completely dependent on the content…
- Or, in the words of Gatys when presenting the paper:
  "The Gram matrix encodes second order statistics of a set of filters. It sort of mushes up all the features at a given layer, tossing out spatial information in favor of a measure of how correlated the different features [are]."

Gatys's Procedure

Gatys, et al. 2016



Style Image into VGG

Iteratively Optimized Image

Content Image into VGG

The Loss Functions

$$\mathscr{L}_c(I_c, I_{new}) = \sum_{l \in L_c} \lambda_l \cdot \|A_c^{(l)} - A_{new}^{(l)}\|^2$$

Content Loss

$$\mathscr{L}_s(I_s, I_{new}) = \sum_{l \in L_s} \beta_l \cdot \|G_s^{(l)} - G_{new}^{(l)}\|^2$$

Style Loss

$$\mathscr{L}_{tv}(I_{new}) = \sum_{i,j} \|I_{i,j} - I_{i,j+1}\|^2 + \|I_{i,j} - I_{i+1,j}\|^2$$

Total Variation

Total Loss $$\mathscr{L}(I_c, I_s, I_{new}) = \alpha \cdot \mathscr{L}_c(I_c, I_{new}) + \beta \cdot \mathscr{L}_s(I_s, I_{new}) + \mathscr{L}_{tv}(I_{new})$$

- Hyperparameters:
  - alpha/beta ratio
  - I_{in} initialization method: I_c, I_s, *White Noise*
  - Layers to use in VGG

$$I_{new} \leftarrow I_{new} + \eta \nabla \mathscr{L}(I_c, I_s, I_{new})$$

Update Equation

Alpha Beta Ratio

10^{-4}

10^{-3}

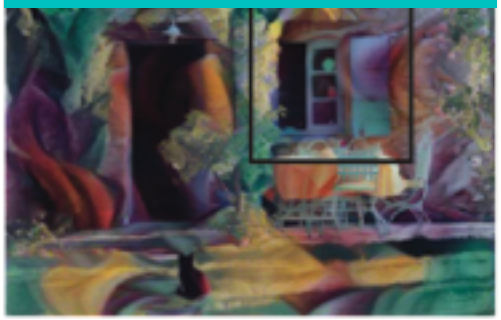
10^{-2}

10^{-1}

Gatys, et al. 2016

Layer Selection and Initialization

Content Loss: Convolutional Layer 2

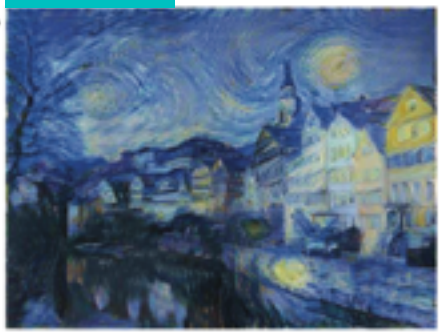


Content Loss: Convolutional Layer 4

Gatys, et al. 2016

Init Content

A

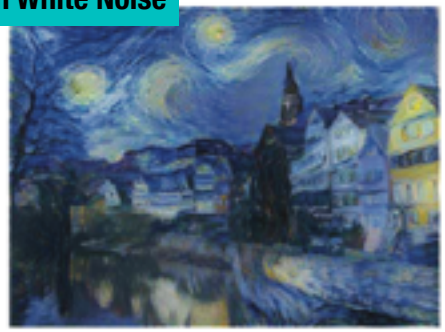


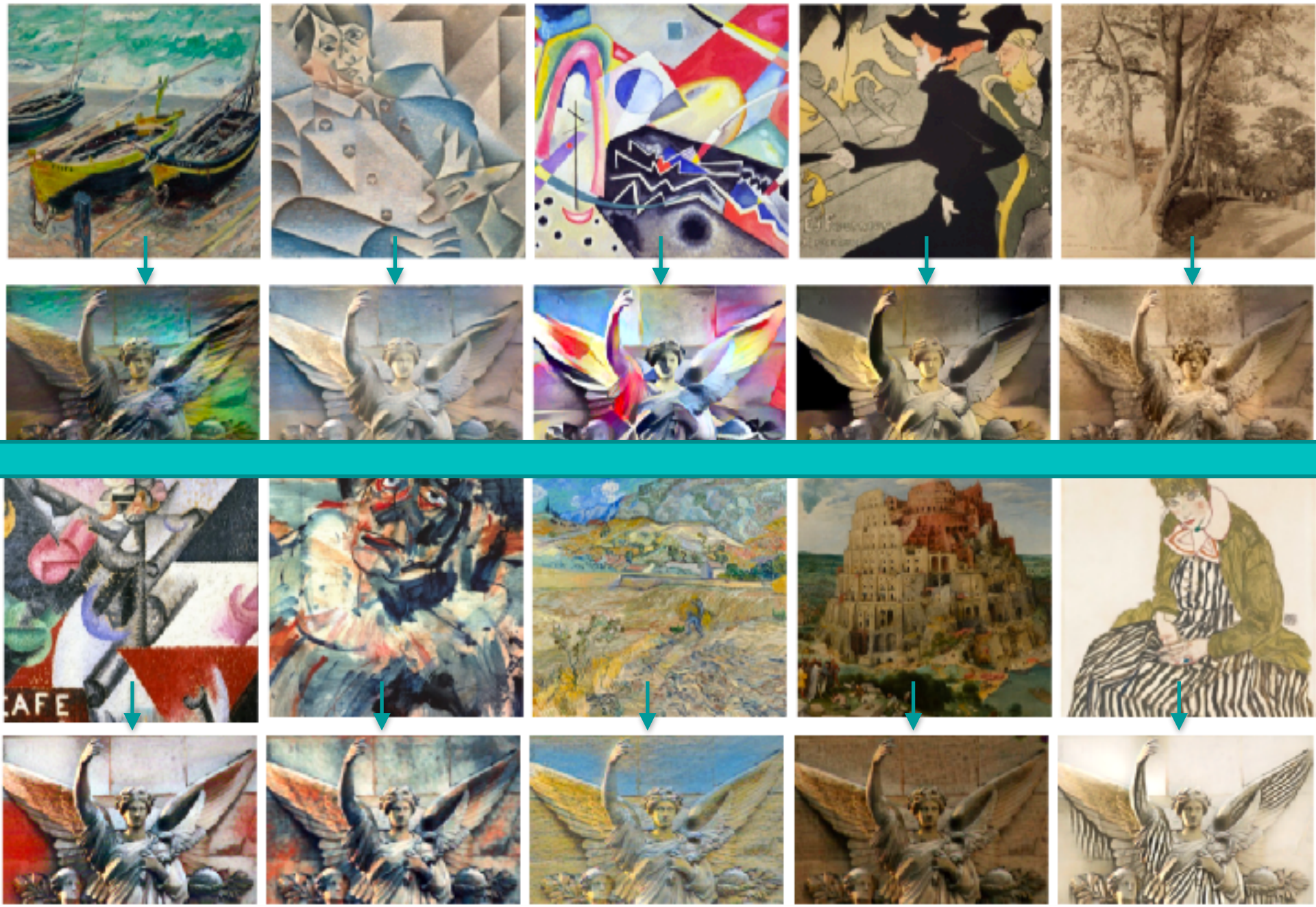
Init Style

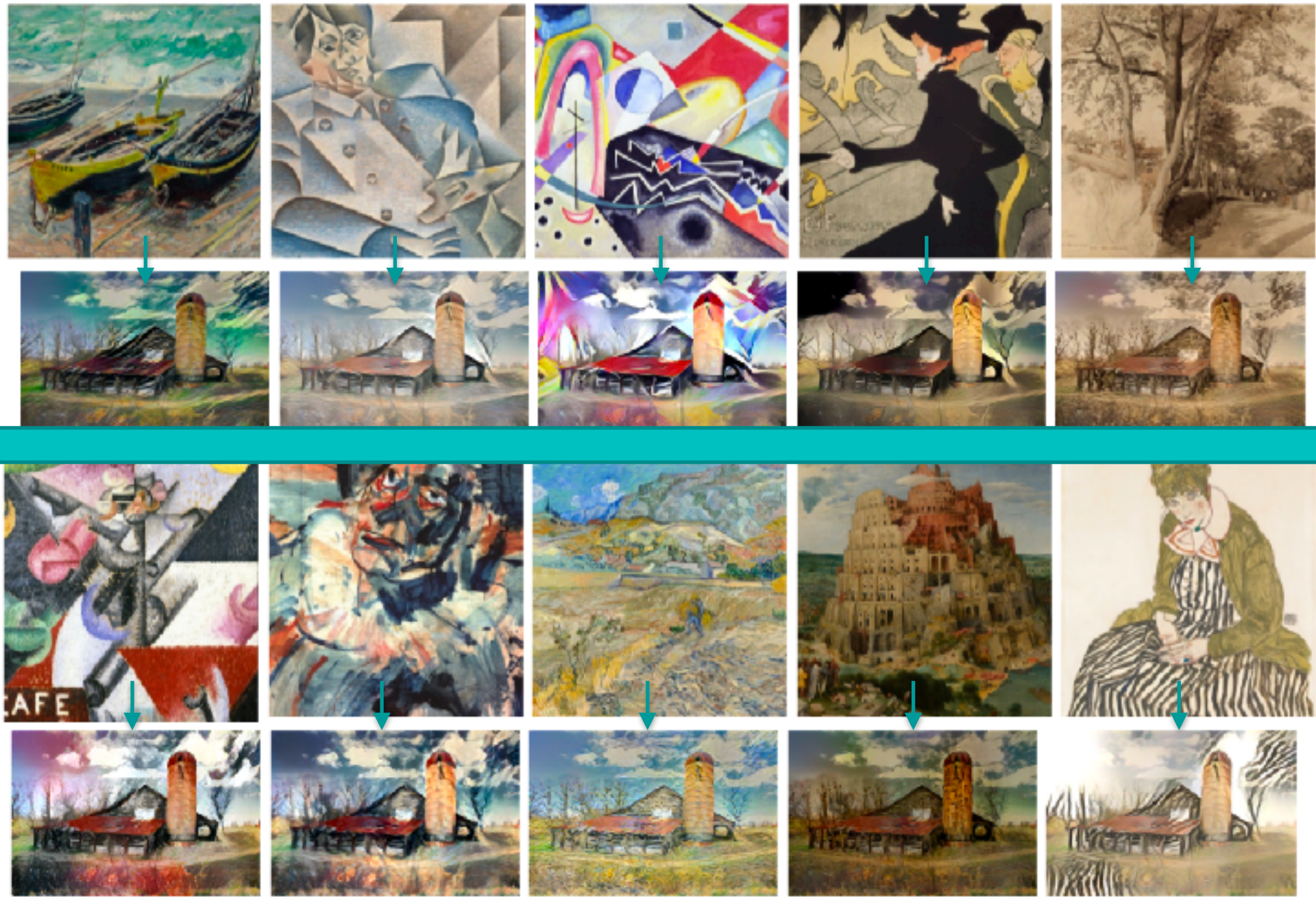
B

C

Init Random White Noise




AFE



Specifics of Implementation

- Uses basic tensorflow operations
  - GradientTape
  - Einsum
  - Loading from VGG
- Normalizes Style loss
  - sum / size^2 channels^2

Demo by Google

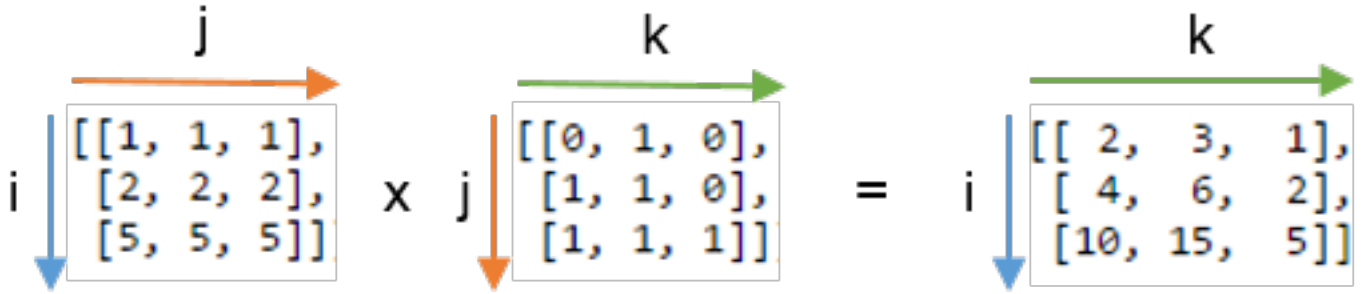
$$G_{i,j}^{(l)} = \sum A_i^{(l)} \cdot A_j^{(l)}$$

size squared

$$\sum_{l \in L_s} \beta_l \cdot \| G_s^{(l)} - G_{new}^{(l)} \|^2$$

channels squared

```
np.einsum('ij,jk->ik', A, B)
```



https://ajcr.net/Basic-guide-to-einsum/

Image Optimization Based Style Transfer

Gatys, et. al

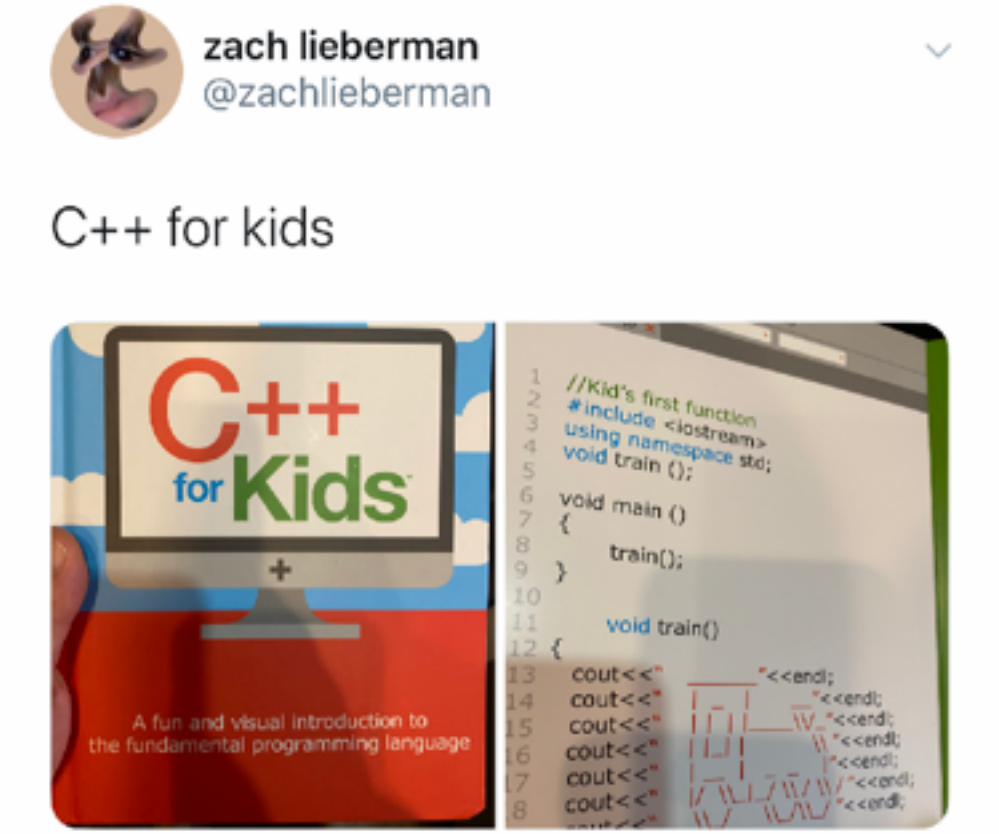
Our master class repository:
05a GatysStyleTransfer.ipynb

Alternative Implementation:

Follow Along: https://github.com/fchollet/deep-learning-with-python-notebooks/blob/master/8.3-neural-style-transfer.ipynb

Demo by Francois Chollet

Model Optimization Based Style Transfer



Paper Presentation

Perceptual Losses for Real-Time Style Transfer and Super-Resolution

Justin Johnson, Alexandre Alahi, Li Fei-Fei
{jcjohns, alahi, feifeili}@cs.stanford.edu

Department of Computer Science, Stanford University

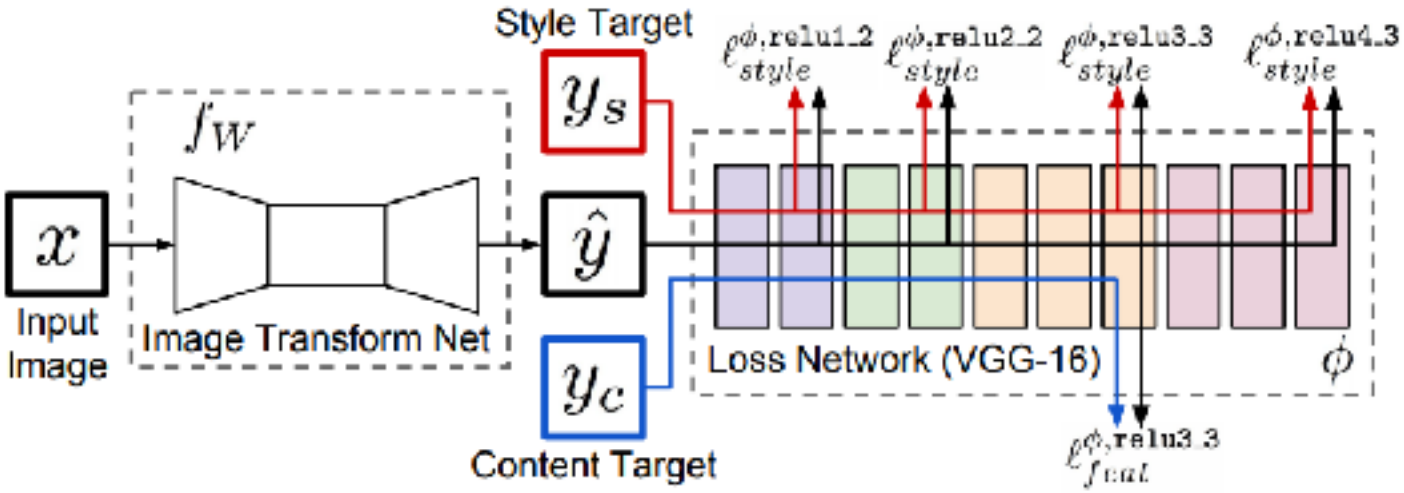
Abstract. We consider image transformation problems, where an input image is transformed into an output image. Recent methods for such

One other thought!

- I hate the term **perceptual loss**!
- It's marketing! Joy!
- …but not good science verbiage
- …like the term global warming
  - it can introduce subjective opinion, misunderstanding through misleading labeling
- There is nothing perceptual here, its a neural network
- A better description of the loss:
  - Convolutional Gram Loss?
  - Information Distillation Covariance?
  - Weighted Grammian Norm (WGN Loss)?
- Just don't say "perceptual" in this class (so you don't fail)

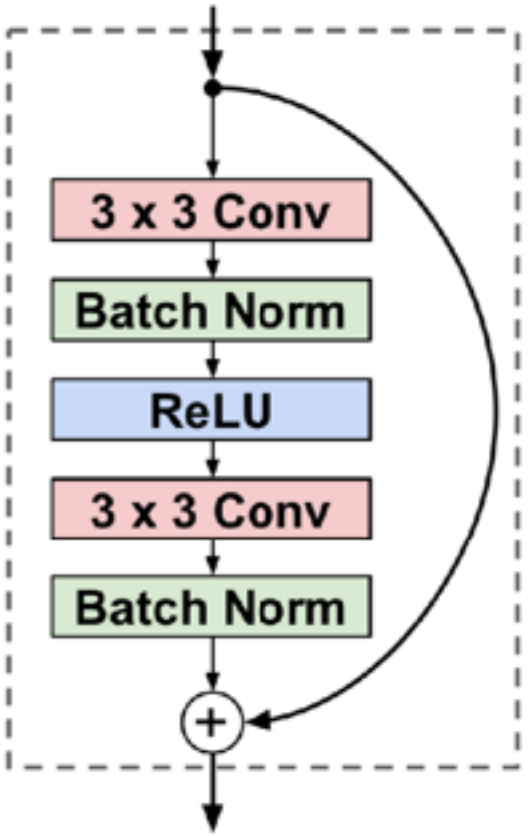
Johnson Paper Recap (if needed)

- **Basic Idea**: replace image optimization with single pass, fully convolutional network to perform the transformation
- Loss functions stay identical to Gatys
  - "Content" through activations difference
  - "Style" through Grammian differences
- Instances are normalized along the channel input
- No bias in filters



Specifics of the Architecture

Layer	Activation size
Input	$3 \times 256 \times 256$
$32 \times 9 \times 9$ conv, stride 1	$32 \times 256 \times 256$
$64 \times 3 \times 3$ conv, stride 2	$64 \times 128 \times 128$
$128 \times 3 \times 3$ conv, stride 2	$128 \times 64 \times 64$
Residual block, 128 filters	$128 \times 64 \times 64$
Residual block, 128 filters	$128 \times 64 \times 64$
Residual block, 128 filters	$128 \times 64 \times 64$
Residual block, 128 filters	$128 \times 64 \times 64$
Residual block, 128 filters	$128 \times 64 \times 64$
$64 \times 3 \times 3$ conv, stride 1/2	$64 \times 128 \times 128$
$32 \times 3 \times 3$ conv, stride 1/2	$32 \times 256 \times 256$
$3 \times 9 \times 9$ conv, stride 1	$3 \times 256 \times 256$



Methods	Time(s)			Styles/Model
	256×256	512×512	1024×1024	
Gatys et al. [10]	14.32	51.19	200.3	∞
Johnson et al. [47]	0.014	0.045	0.166	1

Johnson Paper Extensions

- Multi-style transfer:

Set of Style Images

For each Style Image

$$\mathscr{L}_s(I_{s0}, \ldots, I_{sN}, I_{new}) = \sum_{i=0}^{N} \beta_i \sum_{l \in L_s} \| G_{si}^{(l)} - G_{new}^{(l)} \|^2$$

(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

Final projects from: http://cs231n.stanford.edu/reports/2017/pdfs/428.pdf

Fast Style Paper Extensions

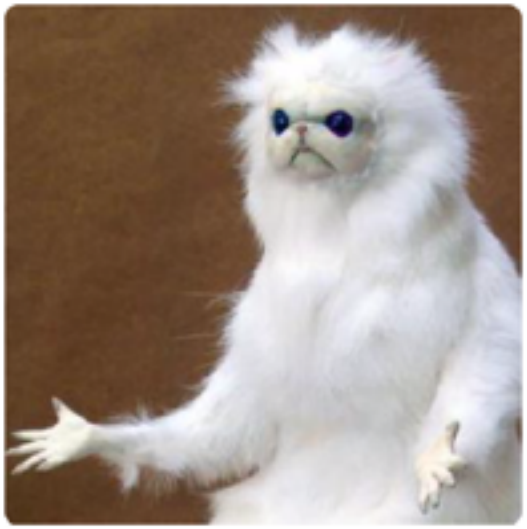
- Color Preservation:
  - Apply style to luminance only
  - Reapply color channels from original image (blurred)
  - HS{$V_{transform}$}

Final projects from: http://cs231n.stanford.edu/reports/2017/pdfs/428.pdf

Model Based Optimization Style Transfer

Johnson, *et. al* Fast Style Transfer

Jake Carlson



Justin Ledford

Luke Wood

`Follow Along: LectureNotesMaster/05 LectureStyleTransfer.ipynb`
`Training:` https://github.com/8000net/StyleTransfer/blob/master/Style_Transfer_Training.ipynb

Optional Demo

One Shot Transfer

The Problem of Training

- One network is only capable of one style transformation
  - Great for trained filters in an app
  - Does not work for generic transfer of one style to another
- How to define a transformation on a content and style image that transfers style into the content?
  - …without any "per style" training
  - …but still using the "grammian style losses"
  - …and allows for infinite customization?
- We need to first cover **whitening** and **coloring** transformations

Aside: Whitening and Coloring

- Signal processing terms applied typically to spectra of signals
- **Whitening** is the process of removing correlation in a signal
  - For a matrix, the whitening transform:
    - makes covariance nearly diagonal (identity)
    - using only linear operations
- **Coloring** is the process of adding the observed correlation back into a signal
  - For a matrix, the coloring transform
    - makes the covariance of signal A as close as possible to signal B (the target)
    - with only linear operations to A

Aside: Whitening with SVD

- Recall that Singular Value Decomposition (SVD) decomposes a matrix into three elements
  - $A = U\Sigma V^T$
  - U is eig-vec of AA^T and V is eig-vec of A^TA
  - where U and V are orthogonal matrices such that
    $$UU^T=I \quad \text{and} \quad VV^T=I$$
  - Σ is a diagonal matrix of the singular values
- the matrix $UV^T=A_w$ is a whitened version of the signal A such that
  $$A_w\, A_w{}^T=I$$
- since the Gram of a layer activation is $A\, A^T=G$, the whitened signal has $A_w\, A_w{}^T = I = G$
  - which would have "no style" according to Gatys

Coloring with SVD

- Suppose we have two activations
  - $A_c = U_c \Sigma_c V_c^T$ and $G_c = A_c A_c^T$
  - $A_s = U_s \Sigma_s V_s^T$ and $G_s = A_s A_s^T$
- We can transfer the Gram matrix of A_s to A_c using coloring
  - To color the matrix:

$$A_{new} = U_s \Sigma_s V_c^T$$

$$A_{new} \, A_{new}^T = G_s$$

  - But A_{new} is more similar to A_c than A_s
- That is exactly what we want for style transfer!
- *However, there are some computational problems with this method of coloring, so in practice, we use the SVD of the Grammian to perform coloring.*

Whitening and Coloring

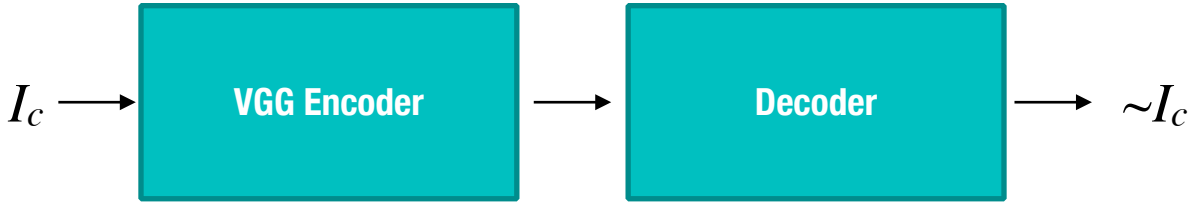
Computational Problems with coloring and the Eigen Decomposition

Demo will introduce concept of coloring with PCA

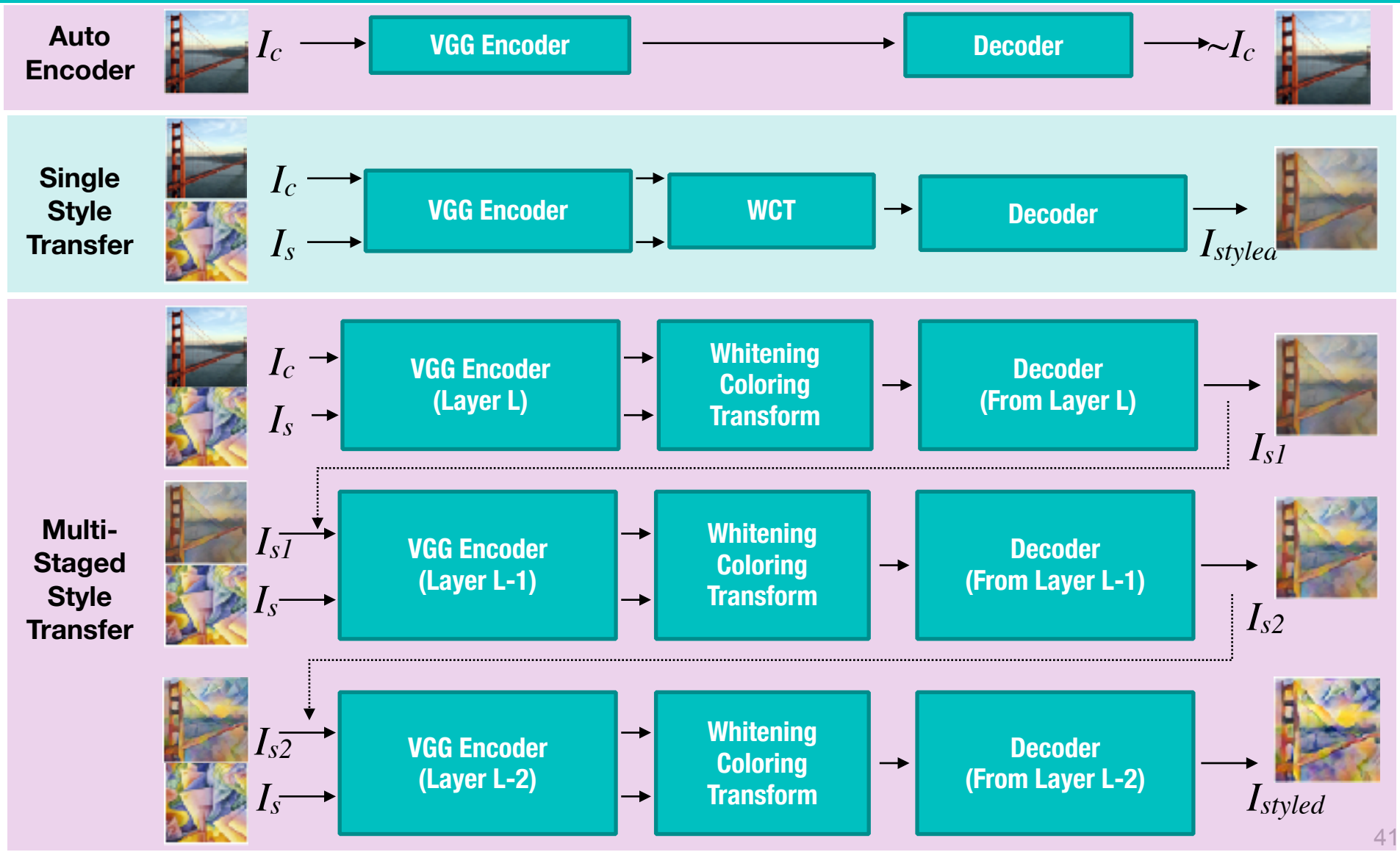
```
Follow Along:
05b WhiteningColoringExample.ipynb
```


How to use this for style transfer?

- If we can learn to **reconstruct an image from VGG**…
  - …we can **whiten content activations**
  - …then **color with desired style Grammian**
- The resulting reconstruction should have largely the same content, but style from the colored activations
- …One transformation network for any style!

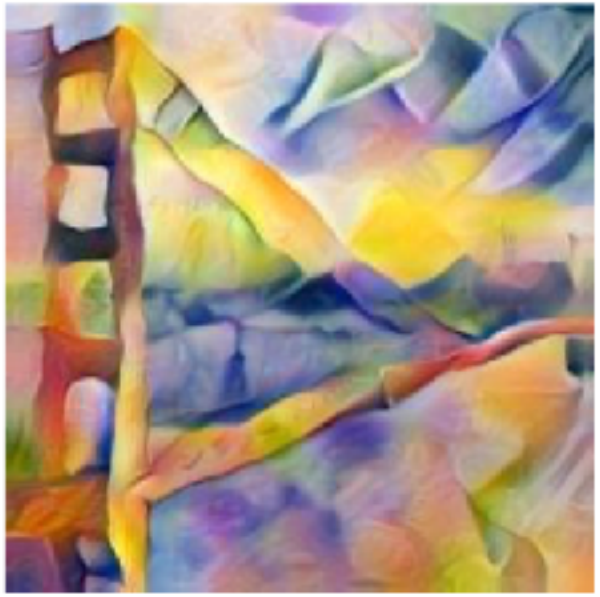


Multi-Staged WCT



Why not go the other way?

- Start at earlier layers and apply WCT as we progress through the network
- Paper does not have good explanation, but results are subjectively poorer:



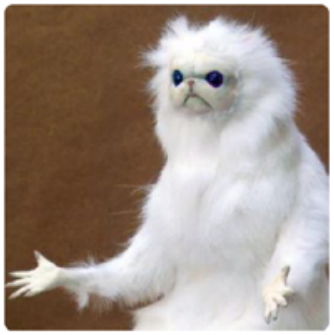
L > L-1 > L-2 > L-3

L-3 > L-2 > L-1 > L

Removing Style? Only Whitening

One Shot Style Transfer

Li, *et. al* Universal Style Transfer



justinledford

Justin Ledford

Yihao Wang

Follow Along: https://github.com/8000net/universal-style-transfer-keras

Or in the master repository:
05c UniversalStyleTransfer.ipynb

Lecture Notes for

Neural Networks and Machine Learning

Style Transfer: Model Opt.

Next Time:
Photo Realistic WCT

Reading: None