Neural Art
by Mark Chang
A Neural Algorithm of Artistic Style

• Author:
  – Leon A. Gatys.
  – Alexander S. Ecker.
  – Matthias Bethge

• Organization:
  – Werner Reichardt Centre for Integrative Neuroscience and Institute of Theoretical Physics, University of Tubingen, Germany.
  – Bernstein Center for Computational Neuroscience, Tubingen, Germany.
The Mechanism of Painting

Scene

Artist

Style

Brain

ArtWork

Computer

Neural Networks
Overview

• Visual Perception
• Computer Vision
• Neural Art
• Demo
Visual Perception

- Neuron
- Visual Pathway
- Misconception
Neuron

- Neuron
- Dendrite
- Cell Body
- Axon

Action Potential

- Voltage
- Time

Threshold
Visual Pathway

- Visual Area V1
- Visual Area V4
- Inferior Temporal Gyrus (IT)

Retina
Visual Pathway

Visual Area V1

Visual Area V4

Inferior Temporal Gyrus (IT)

Receptive Fields
Misconception
Computer Vision

- Neural Networks
- Convolutional Neural Networks
- VGG 19
Neural Networks

\[ n_{out} = \frac{1}{1 + e^{-n_{in}}} \]

\[ n_{in} = w_1 x_1 + w_2 x_2 + w_b \]

\[ n_{out} = \begin{cases} 
  n_{in} & \text{if } n_{in} > 0 \\
  0 & \text{otherwise} 
\end{cases} \]
Neural Networks

Input Layer

Hidden Layer

Output Layer

$W_{12,y}$ $W_{12,x}$ $W_{11,y}$ $W_{11,x}$ $W_{11,b}$ $W_{12,b}$

$W_{21,11}$ $W_{22,11}$ $W_{21,12}$ $W_{22,12}$ $W_{21,b}$ $W_{22,b}$

$x$ $y$ $n_{11}$ $n_{12}$ $n_{21}$ $n_{22}$ $z_1$ $z_2$
Convolutional Neural Networks

- Convolutional Layer

  - weights
  - weights
  - shared weight
  - height
  - depth
  - width
Convolutional Neural Networks

- **Stride**
  - Stride = 1
  - Stride = 2

- **Padding**
  - Padding = 0
  - Padding = 1
Convolutional Neural Networks

- Pooling Layer

- no padding
- no overlap
- no weights
- depth = 1

- Maximum Pooling

- Average Pooling
Convolutional Neural Networks

- Input Layer
- Convolutional Layer
- Pooling Layer
- Convolutional Layer
- Pooling Layer

Receptive Fields
Convolutional Neural Networks

Input Layer

Convolutional Layer with Receptive Fields:

Max-pooling Layer with Width = 3, Height = 3

Filter Responses

Input Image

Filter Responses
VGG 19

- ImageNet Challenge 2014
- 19 (+5) layers
  - 16 Convolutional layers (width=3, height=3)
  - 5 Max-pooling layers (width=2, height=2)
  - 3 Fully-connected layers
VGG 19

depth=64
conv1_1
conv1_2
depth=128
conv2_1
conv2_2
depth=256
conv3_1
conv3_2
conv3_3
conv3_4
depth=512
conv4_1
conv4_2
conv4_3
conv4_4
depth=512
conv5_1
conv5_2
conv5_3
conv5_4
size=4096
FC1
FC2
size=1000
softmax
Neural Art

- Content Generation
- Style Generation
- Artwork Generation
Content Generation

Content

Canvas

Artist

Brain

Neural Stimulation

Minimize the difference

Draw
Content Generation

Content → VGG19 → Filter Responses → \( \text{Width} \times \text{Height} \) → Depth → Minimize the difference → Update the color of the pixels

Canvas → Result
Content Generation

Input Photo: $p$

Layer l’s Filter Responses: $P^l$

Input Canvas: $x$

Layer l’s Filter l Responses: $X^l$

$$L_{content}(p, x, l) = \frac{1}{2} \sum_{i,j} (X^l_{i,j} - P^l_{i,j})^2$$

$$\frac{\partial L_{content}(p, x, l)}{\partial X^l_{i,j}} = X^l_{i,j} - P^l_{i,j}$$
Content Generation

- Backward Propagation

Input Canvas: \( x \)

VGG19

Layer \( l \)’s Filter \( l \) Responses: \( X^l \)

\[
\frac{\partial L_{\text{content}}}{\partial x} = \frac{\partial L_{\text{content}}}{\partial X^l} \frac{\partial X^l}{x}
\]

Update Canvas

\( x \leftarrow x - \eta \frac{\partial L_{\text{content}}}{\partial x} \)

Learning Rate
Content Generation
Content Generation

VGG19

conv1_2  conv2_2  conv3_4  conv4_4  conv5_1  conv5_2
Style Generation

- "Style" is position-independent
Style Generation

Artwork

VGG19

Filter Responses

Gram Matrix

Width*Height

Position-dependent

Depth

Position-independent

Depth
Style Generation

Layer l’s Filter Responses

Width*Height

Depth

 Gram Matrix

\[ G_{i,j}^l = F_i^l \cdot F_j^l \]

Depth

\[ G_{4,1}^l = F_4^l \cdot F_1^l \]

= 1 \times 1 + 0 \times 0.5 + 0 \times 0 + ... \\
= 1
Style Generation

Style

Canvas

VGG19

Filter Responses

Gram Matrix

Minimize the difference

Update the color of the pixels

Result
Style Generation

Input Artwork: $\mathbf{a}$

Layer l’s Gram Matrix

Input Canvas: $\mathbf{x}$

Layer l’s Gram Matrix

Layer l’s Filter Responses

Layer l’s

Filter Responses

$L_{style}(\mathbf{a}, \mathbf{x}, l) = \frac{1}{2} \sum_{i,j} (X_{i,j}^l - A_{i,j}^l)^2$

$\frac{\partial L_{style}(\mathbf{a}, \mathbf{x}, l)}{\partial F_{i,j}^l} = ((F^l)^T(X^l - A^l))_{j,i}$
Style Generation
Style Generation

VGG19

Conv1_1
Conv2_1
Conv3_1
Conv4_1
Conv5_1
Artwork Generation

$L_{total} = \alpha L_{content} + \beta L_{style}$

$x \leftarrow x - \eta \frac{\partial L_{total}}{\partial x}$
Artwork Generation

\[ L_{\text{content}}(p, x) \]

\[ L_{\text{total}} = \alpha L_{\text{content}} + \beta L_{\text{style}} \]
Artwork Generation
Demo

- Content v.s. Style
- Different Initial State
- Different VGG Layers
- Sketch & Watercolor
- Painting & Poem
Content v.s. Style

\[ \alpha \]

\[ \beta \]

0.15

0.05

0.02

0.007
Different Initial State

- noise
- $0.9 \times \text{noise} + 0.1 \times \text{photo}$
- photo
Different VGG Layers

\[ \frac{\alpha}{\beta} = 0.002 \]
Sketch & Watercolor
Further Reading

- A Neural Algorithm of Artistic Style
- Texture Synthesis Using Convolutional Neural Networks
- Convolutional Neural Network
- Neural Network Back Propagation
- Computational Poetry
  - http://www.slideshare.net/ckmarkohchang/computational-poetry
Code

- Python Tensorflow
  - https://github.com/ckmarkoh/neuralart_tensorflow

- Python Theano
  - https://github.com/woonketwong/artify

- Python Theano (ipython notebook)
  - https://github.com/Lasagne/Recipes/blob/master/examples/styletransfer/Art%20Style%20Transfer.ipynb

- Python deegppy
  - https://github.com/andersbll/neural_artistic_style
Image URL


Acknowledgement

- NTU imlab
About the Speaker

Mark Chang

- Email: ckmarkoh at gmail dot com
- Blog: http://cpmarkchang.logdown.com
- Github: https://github.com/ckmarkoh
- Facebook: https://www.facebook.com/ckmarkoh.chang
- Slideshare: http://www.slideshare.net/ckmarkohchang