A Neural Algorithm of Artistic Style

Milan Čugurović

DEPARTMENT OF COMPUTER SCIENCE, FACULTY OF MATHEMATICS, UNIVERSITY OF BELGRADE

MACHINE LEARNING GROUP AT FACULTY OF MATHEMATICS, UNIVERSITY OF BELGRADE

How to choose a title?

- **A Neural Algorithm of Artistic Style**

  Leon A. Gatys, Alexander S. Ecker, Matthias Bethge

  Journal of Vision, *August 2015*

- **Perceptual losses for real-time style transfer and super-resolution**

  J Johnson, A Alahi, L Fei-Fei

  European Conference on Computer Vision, October 2016

Being up to date

A neural algorithm of artistic style

Authors	Leon A Gatys, Alexander S Ecker, Matthias Bethge
Publication date	2015/8/26
Journal	arXiv preprint arXiv:1508.06576
Total citations	Cited by 731

Scholar articles A neural algorithm of artistic style
LA Gatys, AS Ecker, M Bethge - arXiv preprint arXiv:1508.06576, 2015
Cited by 731 Related articles All 17 versions

Perceptual losses for real-time style transfer and super-resolution

Authors	Justin Johnson, Alexandre Alahi, Li Fei-Fei
Publication date	2016/10/8
Conference	European Conference on Computer Vision
Pages	694-711
Publisher	Springer, Cham
Total citations	Cited by 1186

Scholar articles Perceptual losses for real-time style transfer and super-resolution
J Johnson, A Alahi, L Fei-Fei - European Conference on Computer Vision, 2016
Cited by 1186 Related articles All 5 versions

Before we start…

$$J^{(D)}\left(\theta^{(D)}, \theta^{(G)}\right) = -\frac{1}{2}E_{x\sim p_{data}} logD(x) - \frac{1}{2}E_z \log(1 - D(G(z)))$$

Teaser

Intro

- Fine art, painting
- Human skill to create unique visual experiences through composing a complex interplay between the **content** and **style** of an image

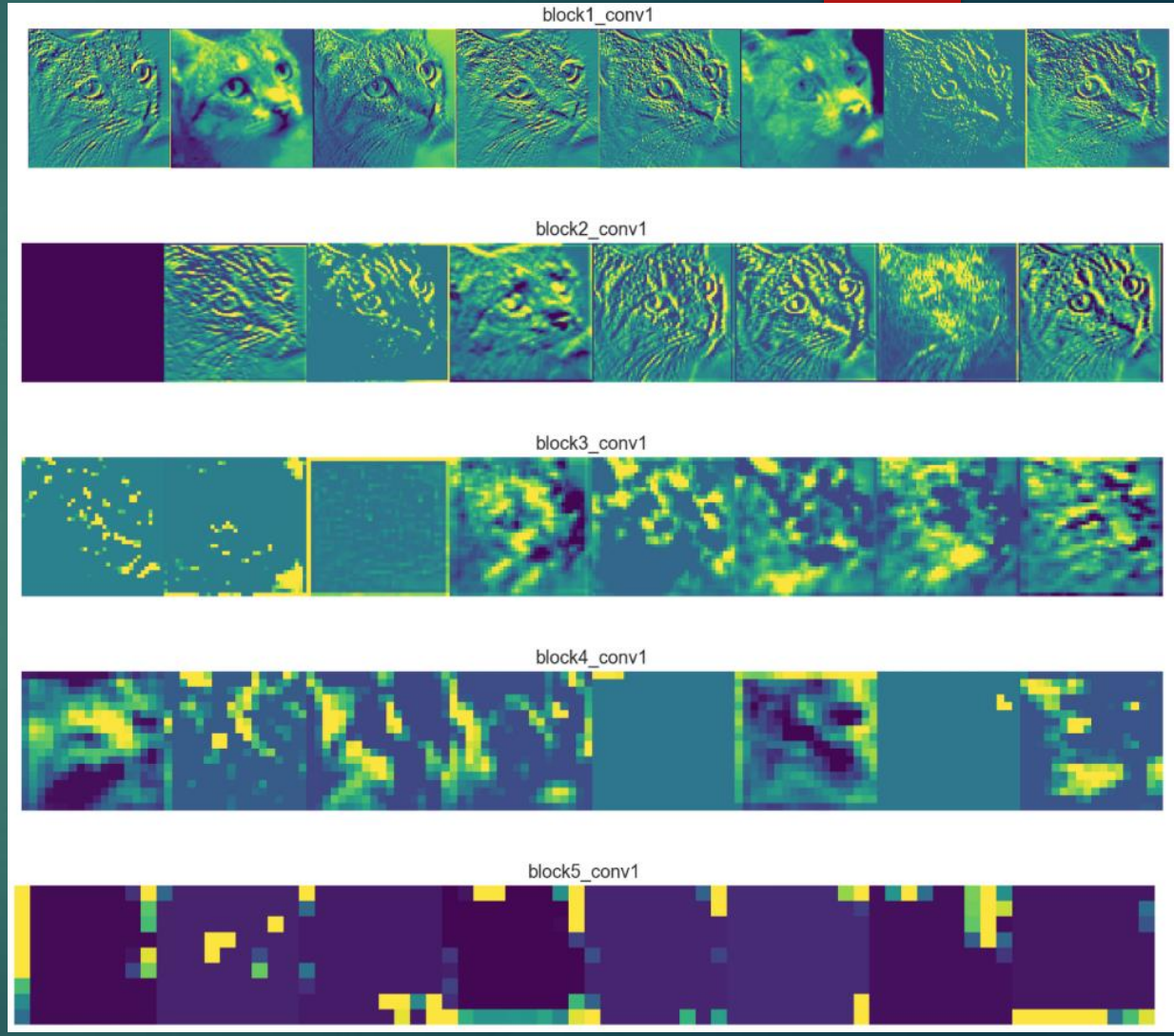
Inspiration

- Algorithm basis of this process is unknown (so far)
- Inspiration: **object detection** and **face recognition** – biologically inspired vision models DNN
- Artificial system based on a Deep Neural Network that *creates artistic images* of high perceptual quality
  - System use neural representations to separate and recombine:
    - Content
    - Style of the image

CNN

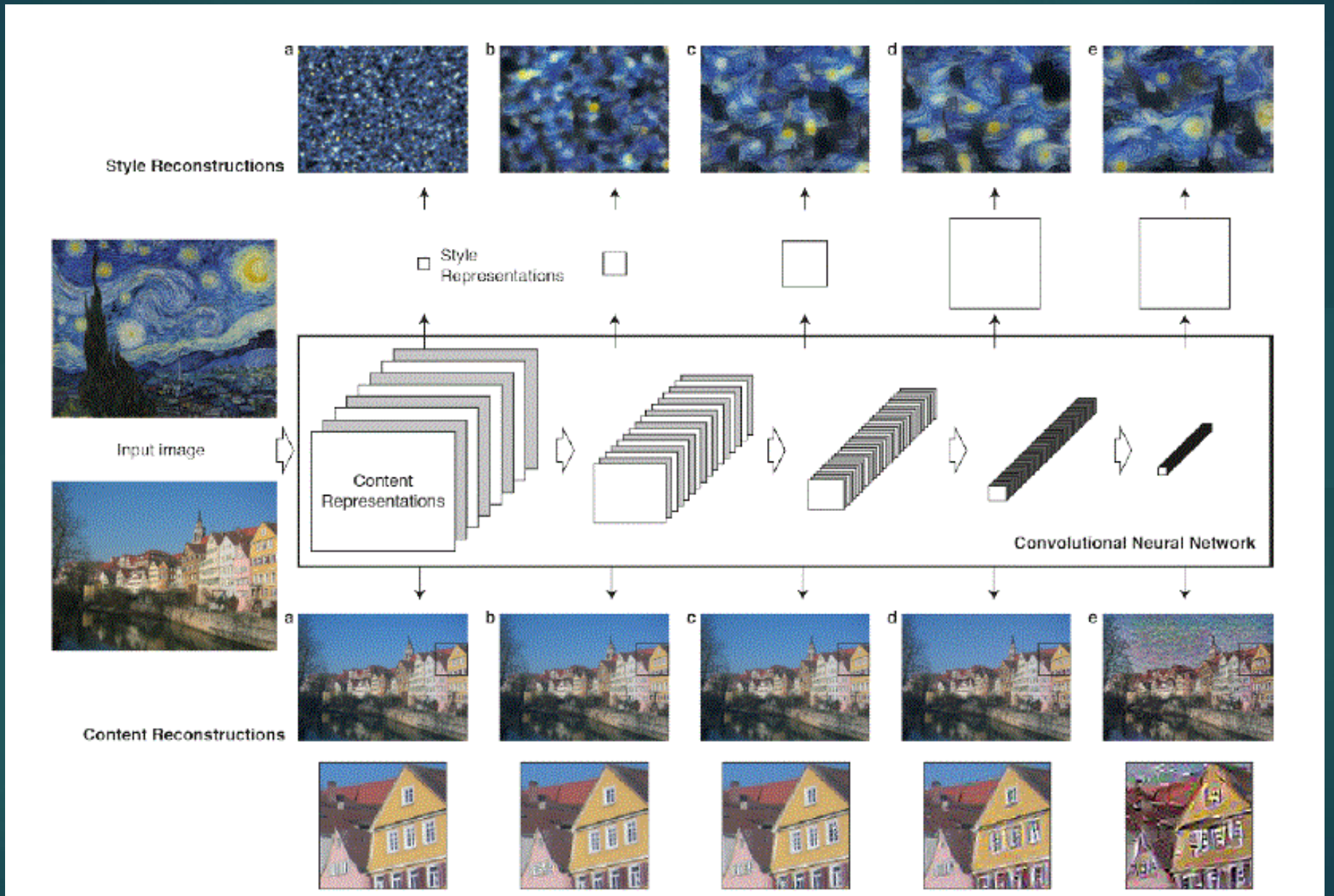
- Convolutional Neural Network - the class of Deep Neural Networks that are most powerful in image processing tasks
- Consist of layers of small computational units (neuron)
- Process visual information hierarchically in a feed-forward manner
- Representation of the image that makes object information increasingly explicit along the processing hierarchy

(Gatys, L. A., Ecker, A. S. & Bethge, M. Texture synthesis and the controlled generation of natural stimuli using convolutional neural networks)



CNN

- Higher layers in the network:
  - capture the high-level **content** in terms of objects and their arrangement in the input image
  - do not constrain the exact pixel values
- feature responses in **higher** layers = **content representation**
- How to catch a **style** of an input image?
  - Use a feature space originally designed to capture texture information
  - This feature space is build on top of the filter responses in each layer
  - Consists of the correlations between the different filter responses over the spatial extent of the feature maps
- Including the feature correlations of multiple layers – capture inputs texture information but not the global arrangements


a
b
c
d
e
Style Reconstructions
Style Representations
Input image
Content Representations
Convolutional Neural Network
a
b
c
d
e
Content Reconstructions

CNN

- Reconstructions from the style features produce texturised versions of the input image that capture its general appearance in terms of *colour* and *localised structures*
- The size and complexity of local image structures from the input image increases along the hierarchy
- Increasing **receptive field sizes** and feature complexity
- We refer to this multi-scale representation as **style representation**

Previous work in this area

BUSINESS INSIDER

TECH | FINANCE | POLITICS | STRATEGY | LIFE | ALL

Spanish police arrest art dealer suspected (by Picasso and Dali

REUTERS Reuters Nov. 30, 2018, 8:50 AM

MADRID (Reuters) - Spanish police have arrested an art dealer

abcNEWS VIDEO LIVE SHOWS

Art Dealer Accused of Selling Millions in Fakes

Jose Carlos Bergantinos Diaz is wanted in the US for fraud.

INSIDER

A museum in France found out that nearly half of its collection was fake.

večernje 65 NOVOSTI ONLINE

NASLOVNA SPORT PLANETA SCENA KULTURA SRBIJA BEOGRAD LIFESTYLE ZABAVA TURIZAM

Politika Društvo Ekonomija Preduzetnik Hronika Dosije Reportaže Republika Srpska Tehnologije A

SAD: Izložba slika čuvenog falsifikatora

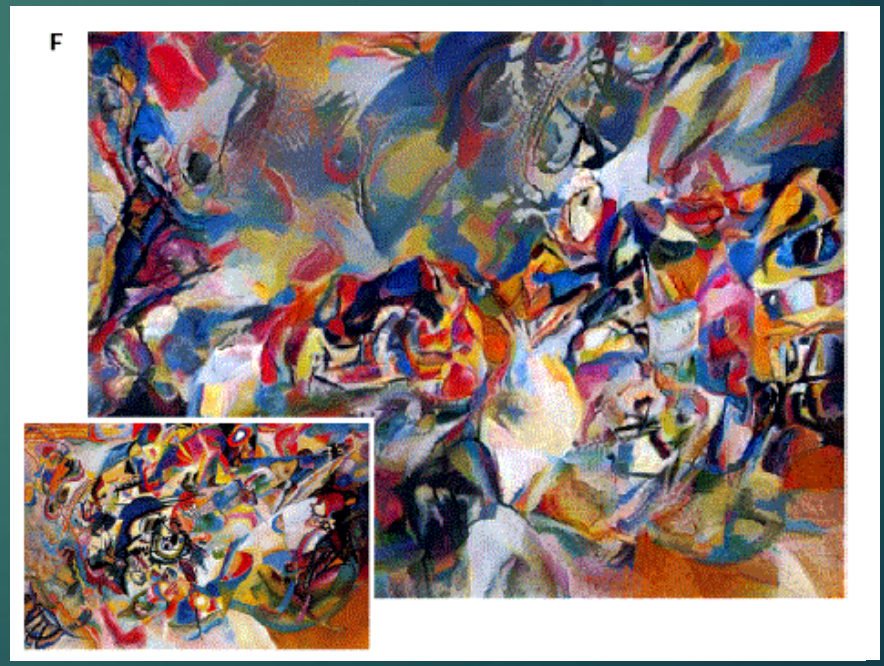
Tanjug | 01. april 2012. 11:58 | Komentara: 0

Na izložbi u Sinsinatiju biće prikazano oko 40 dela koja je Mark A. Landis, uspešni falsifikator umetničkih dela poklonio 15 muzeja, a sam autor obećao je još 60 slika i svoju svešteničku odeću

Препоручите 0 Подели Tweet

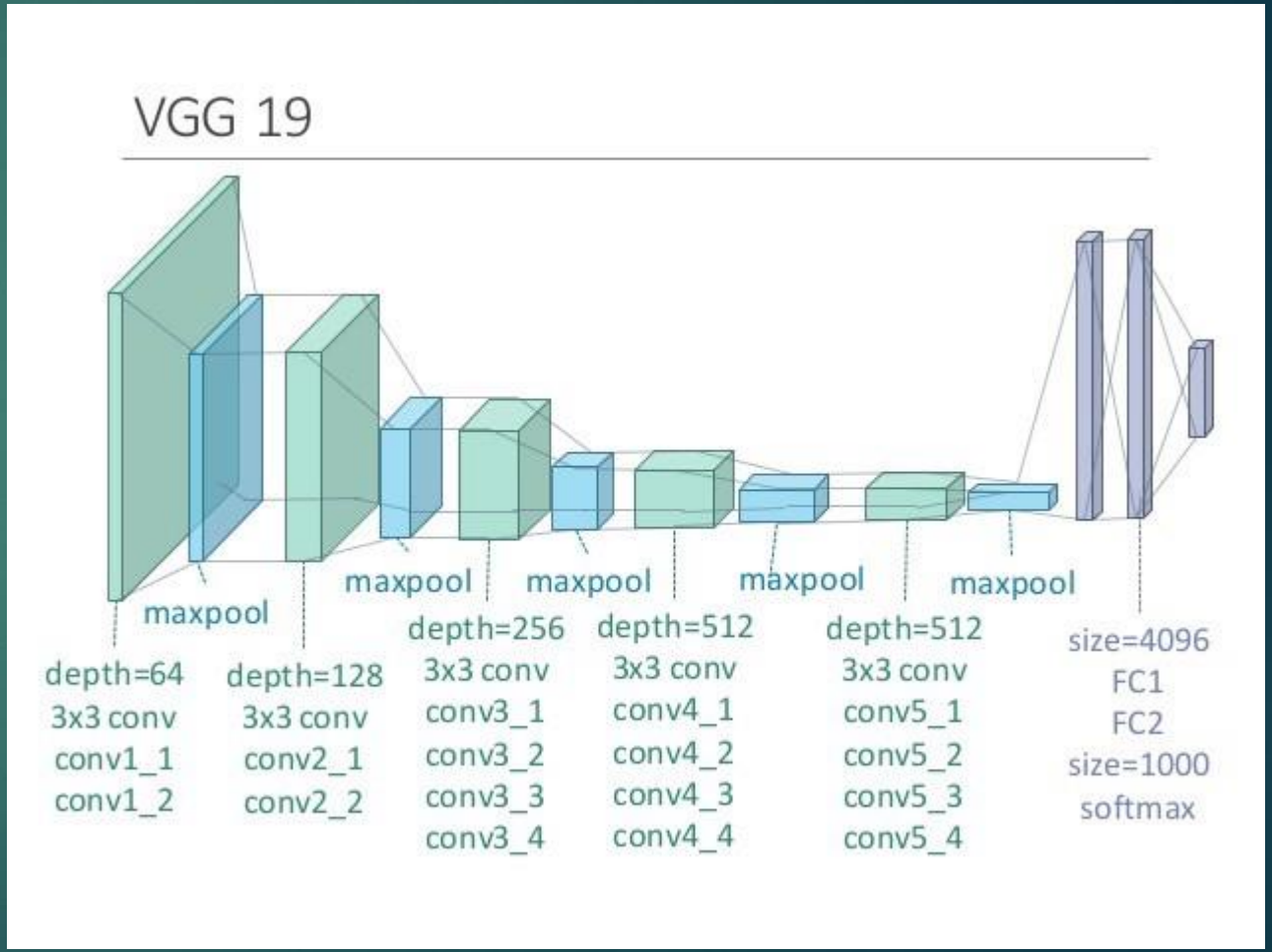
Key finding of this paper

- **The representations of content and style in the CNN are separable**
- We can manipulate both representations independently

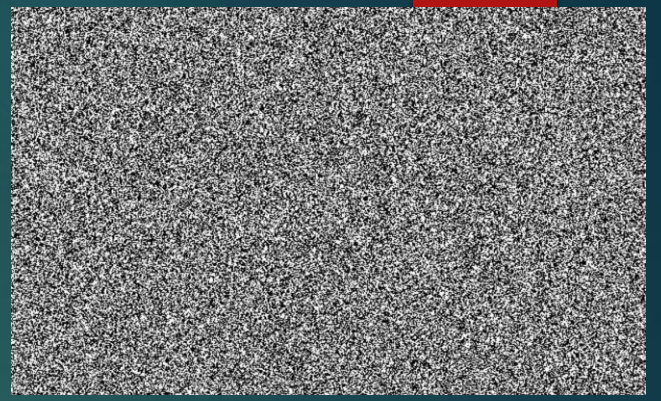


Methods

- Base network: VGG19 Network
- Feature space consists of 19 layers of VGG network:
  - 16 convolutional layer
  - 5 pooling layer
  - Don't use any of the fully connected layers
- max-pooling/average pooling



Methods



- Input image $\vec{x}$ is encoded in each layer of the CNN by the filter responses to that image
- A layer with N_l distinct filters has N_l feature maps each of size M_l
- The responses in a layer l can be stored in a matrix $F^l \in R^{N_l \times M_l}$
  - ${F_{ij}}^l$ is the activation of the i^{th} filter at position j in layer l
- To visualise the image information that is encoded at different layers of the hierarchy - perform **gradient descent** on **a white noise image** to find another image that matches the feature responses of the original image

Methods

- $\vec{p}$ original image (P^lappropriate feature representation in layer l)
- $\vec{x}$ generated image (F^lappropriate feature representation in layer l)
- The squared-error loss between the two feature representations:

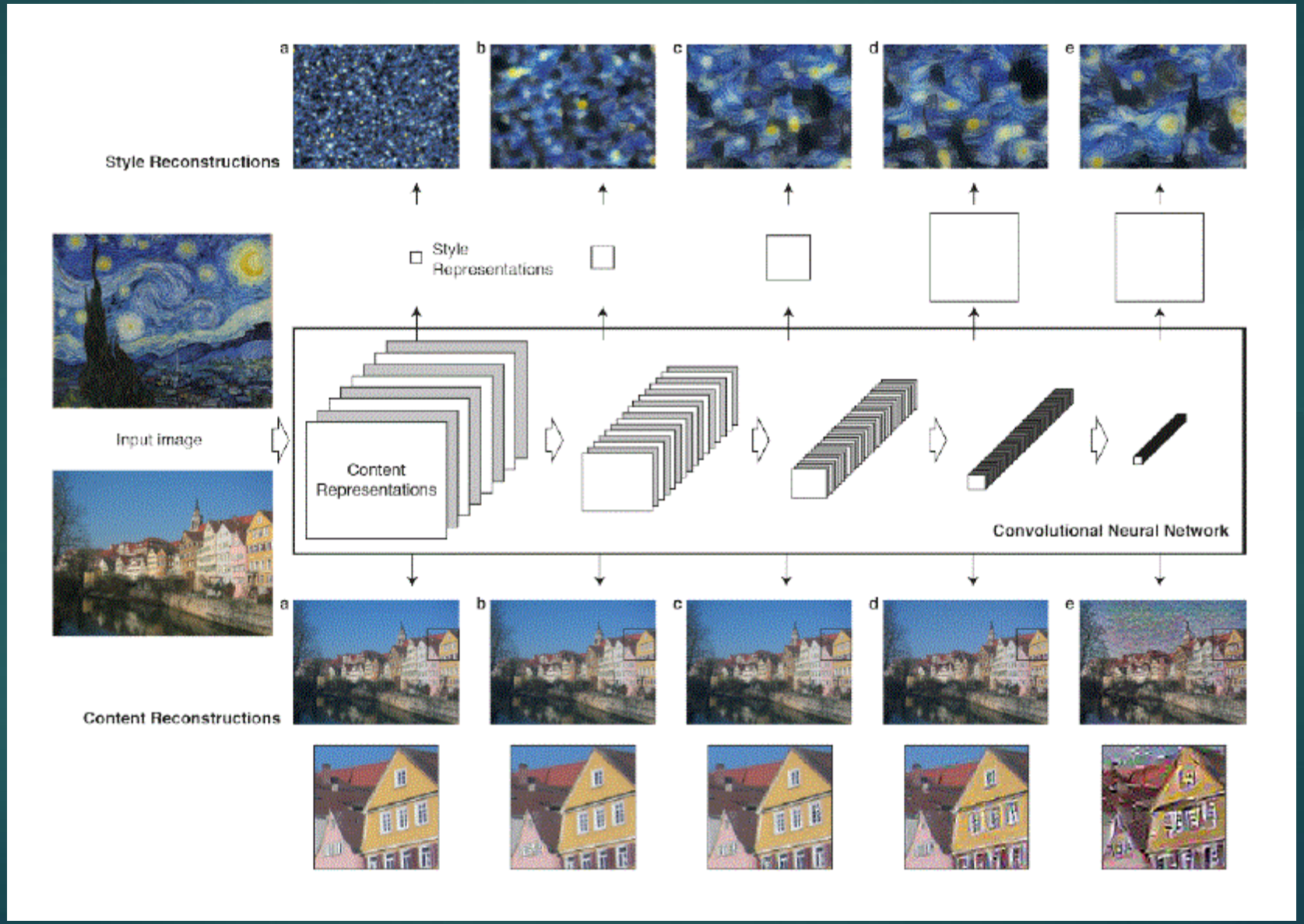
$$L_{content}(\vec{p}, \vec{x}, l) = \frac{1}{2}\sum_{i,j}({F_{ij}}^l - {P_{ij}}^l)^2$$

- The derivative of this loss with respect to the activations in layer l:

$$\frac{\delta L_{content}(\vec{p}, \vec{x}, l)}{\delta {F_{ij}}^l} = \begin{cases} {F_{ij}}^l - {P_{ij}}^l, {F_{ij}}^l > 0 \\ 0 \qquad\qquad , {F_{ij}}^l < 0 \end{cases}$$

- From which the gradient with respect to the image $\vec{x}$ can be computed using standard error **back-propagation**
- We can change, initially random image $\vec{x}$ until it generates the same response in a certain layer of the CNN as the original image $\vec{p}$

Methods


a
b
c
d
e
Style Reconstructions
Style Representations
Input image
Content Representations
Convolutional Neural Network
Content Reconstructions

Methods

- Style representation:
  - On top of the CNN responses in each layer of the network
  - Computers the correlations between the different filter responses
    - (the expectation is taken over the spatial extend of the input image)
- Feature correlations is Gram matrix $G^l \in R^{N_l \times N_l}$:

$$G_{ij}{}^{l} = \sum_{k} F_{ik}{}^{l} F_{jk}{}^{l}$$

Inner product between the *vectorised* feature map *i* and *j* in layer *l*

- **Similar idea:** use gradient descent from a white noise image to find another image that matches the style representation of the original image

Methods

- Minimising the mean-squared distance between the Gram matrix from the original image and the Gram matrix of the image to be generated
- $\vec{a}$ the original image (A^l style representation in layer l)
- $\vec{x}$ image that is generated(G^l style representation in layer l)
- Loss is:

$$L_{style}(\vec{a}, \vec{x}) = \sum_l w_l E_l$$

$$E_l = \frac{1}{4{N_l}^2 {M_l}^2} \sum_{i,j} ({G_{ij}}^l - {A_{ij}}^l)^2$$

w_l weights

Methods

- The derivative of this loss with respect to the activations in layer l

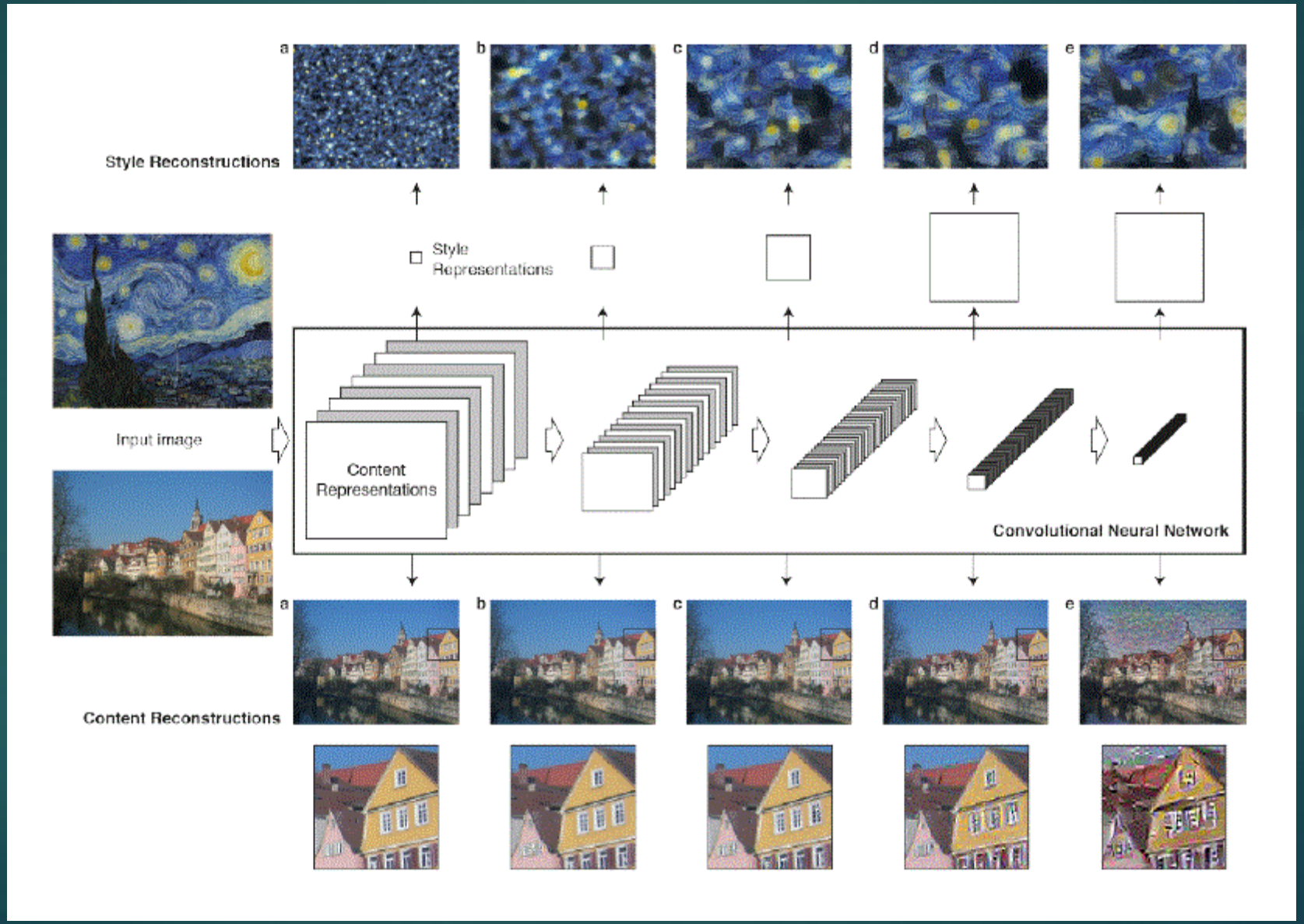
$$\frac{\delta L_{style}(\vec{a},\vec{x})}{\delta {F_{ij}}^{l}} = ?$$

- Linearity of derivation

$$\frac{\delta E_l}{\delta {F_{ij}}^{l}} = \begin{cases} \frac{1}{{N_l}^2 {M_l}^2}\left(\left(F^l\right)^T\left(G^l - A^l\right)\right)_{ji}, {F_{ij}}^{l} > 0 \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad , {F_{ij}}^{l} < 0 \end{cases}$$

- The gradients of El with respect to the activations in lower layers of the network can be readily computed using standard error **back-propagation**

Methods

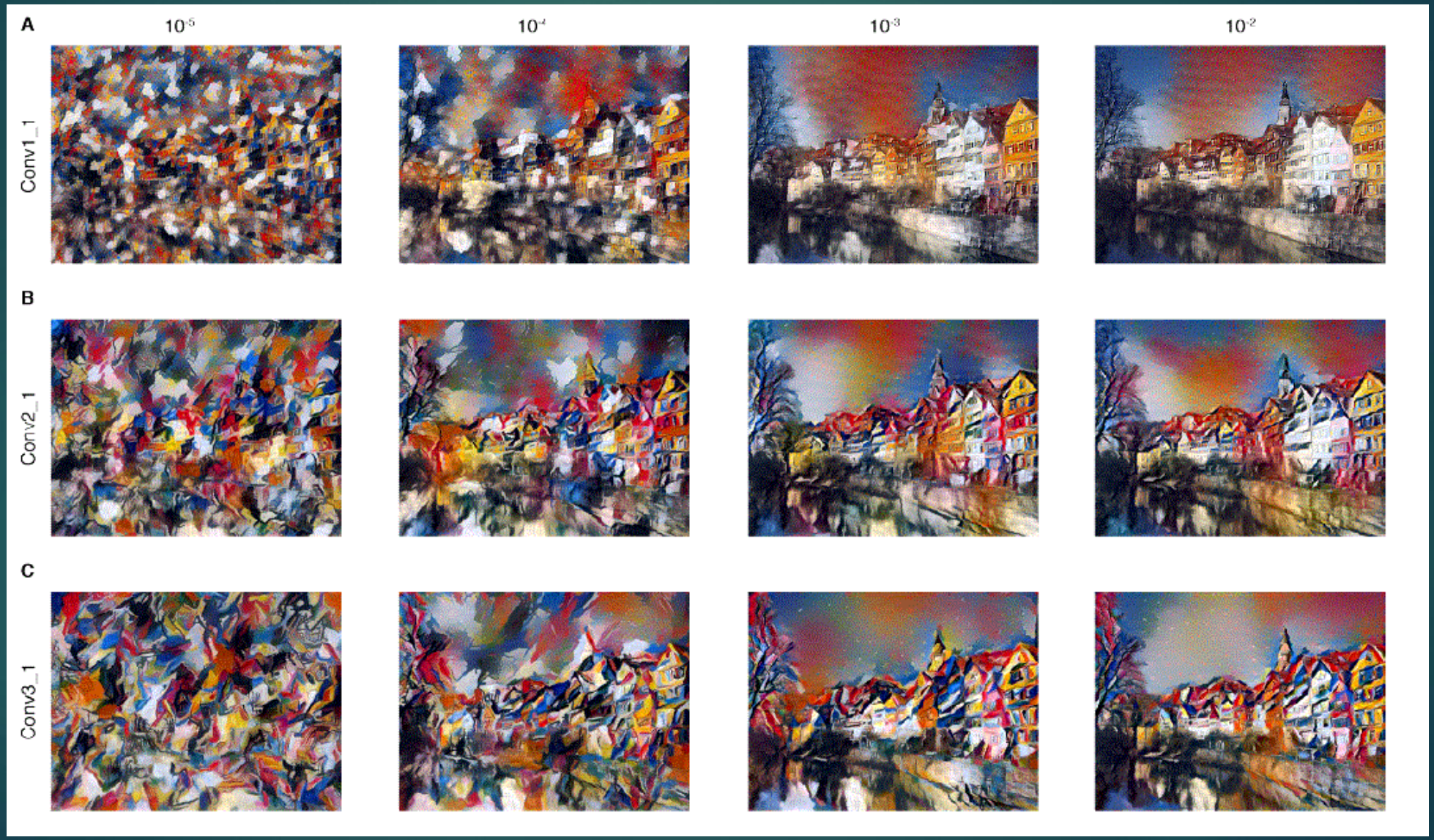

a
b
c
d
e
Style Reconstructions
Style Representations
Input image
Content Representations
Convolutional Neural Network
Content Reconstructions

Methods

- Mix the **content** of a photograph with the **style** of a painting
- *Jointly minimise* the distance of a white noise image from:
  - the content representation of the photograph in one layer of the network
  - the style representation of the painting in a number of layers of the CNN
- $\vec{p}$ photograph
- $\vec{a}$ artwork

$$L_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha L_{content}(\vec{p}, \vec{x}) + \beta L_{style}(\vec{a}, \vec{x})$$

Wassily Kandinsky, *Composition VII*



New approach

- Gatys use **per-pixel loss** to train CNN
  - does not capture perceptual differences between output and ground-truth images
- Johnson introduce **perceptual loss** function
  - Based not on differences *between pixels* but instead on differences *between high-level image feature representations* extracted from pretrained convolutional neural networks
  - Train feed forward **transformation network** for image recognition task
  - Train it using perceptual loss function that depend on high level features from a pretrained (for image classification) **loss network** (more robust, real time in test)

  The loss network remains fixed during the training process

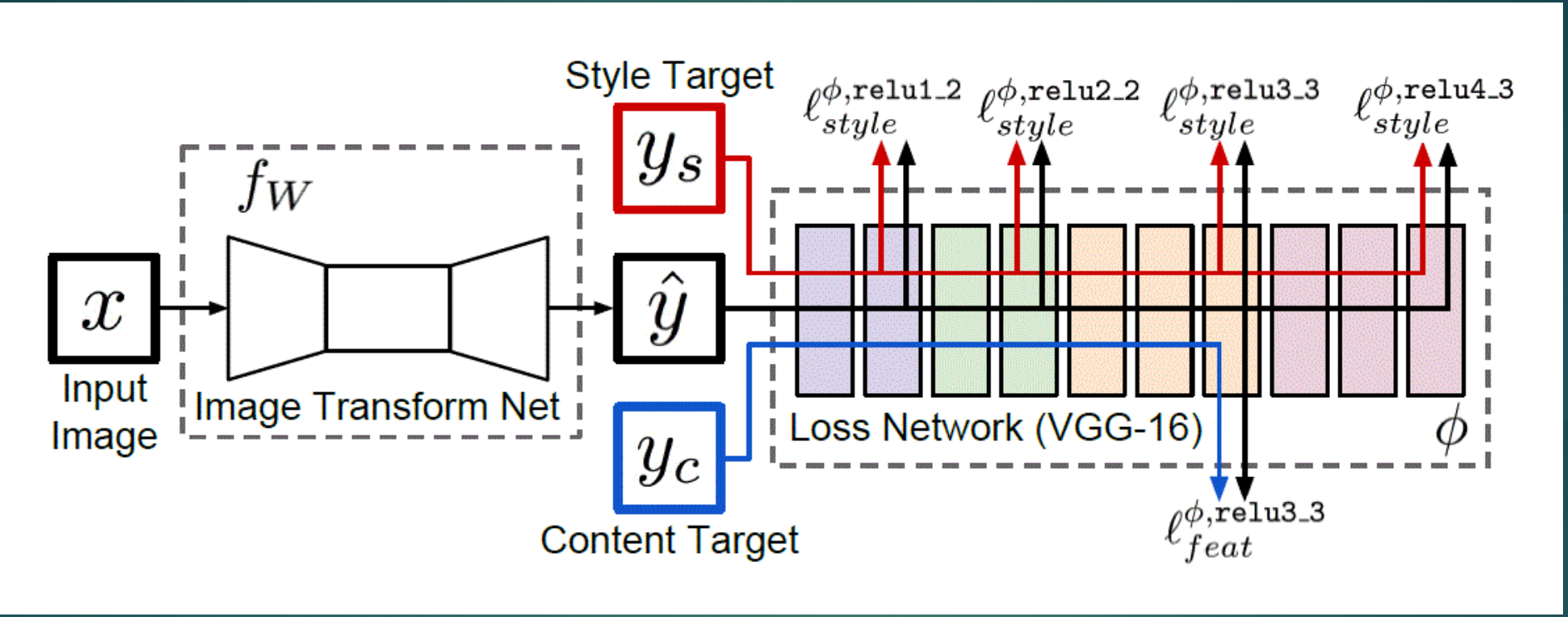
Method

- Image transformation network $\boldsymbol{f_W}$
  - Deep residual CNN parameterized by weights W
  - $\hat{y} = f_W(x)$, where x is input image, $\hat{y}$ output image
- Loss network $\boldsymbol{\phi}$ (used to define several loss function $l_1, \ldots, l_k$
  - Loss function computes a scalar value $l_i(\hat{y}, y_i)$
  - measuring the difference between the output image $\hat{y}$ and a target image y_i

Method

- Key insight: CNN pretrained for image classification task have *already learned* to encode the perceptual and semantic information we would like to measure in our loss functions
- Our deep convolutional transformation network is thus trained using **loss functions** that are **also deep convolutional networks**
- The loss network ϕ is used to define a *feature reconstruction loss* $\boldsymbol{l^{\phi}_{feat}}$ and a *style reconstruction loss* $\boldsymbol{l^{\phi}_{style}}$ that measure differences in **content** and **style** between images

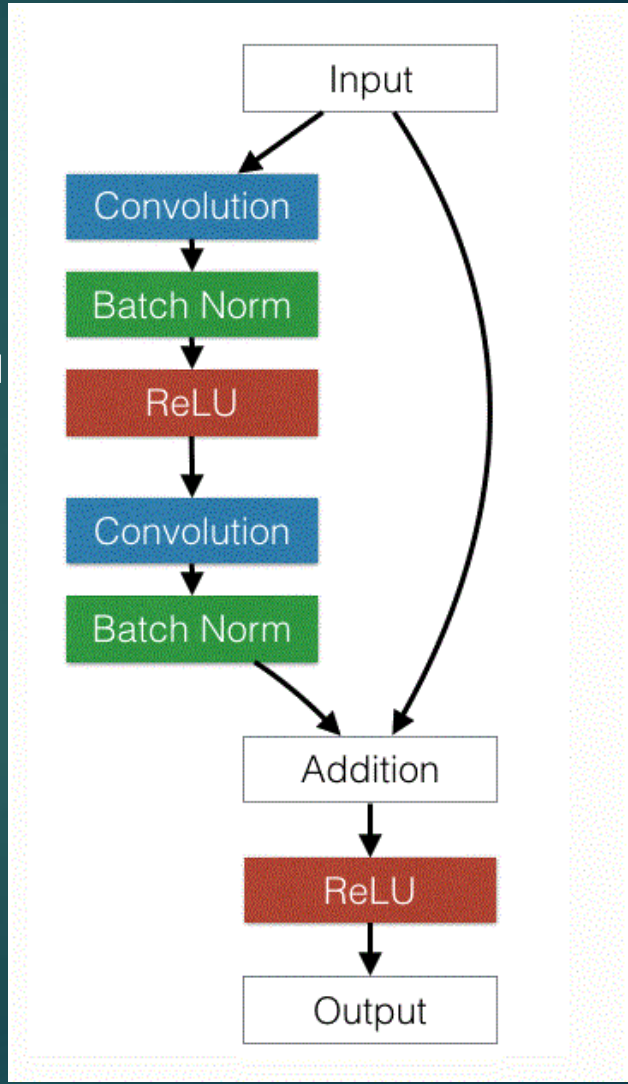
Method



$$W^* = argmin_W \mathbb{E}_{x,\{y_i\}} \left[\sum_i \lambda_i l_i(f_W(x), y_i) \right]$$

Image transformation network

- pooling layers / strided and fractionally strided convolutions
- five residual blocks using ResNet architecture
- Use of batch normalization
- ReLU nonlinearities (except output layer, which use scaled ta
- 3x3 and 9x9 kernels
- Input and Output is RGB images 3x256x256
- Downsampling and Upsampling:
  - Two stride-2 convolutions to downsample
  - Five residual block
  - Two stride-$\frac{1}{2}$ convolutions to upsample
  - Benefits: computational, receptive field size



Perceptual Loss Functions

- Two perceptual loss functions that measure high-level **perceptual** and **semantic** differences between images
- Make use of a loss network ϕ
  - 16-layer VGG network pretrained on ImageNet, Karpathy
- These perceptual loss functions are themselves deep CNNs

Feature Reconstruction Loss

- Gatys:
  - encouraging the pixels of the output image $\hat{y} = f_W(x)$ to exactly match the pixels of the target image y
- Johnson:
  - encourage them to have similar feature representations as computed by the loss network ϕ
- $\phi_j(x)$ the activations of the j-th layer of the network ϕ when processing the image x
  - If j is a convolutional layer, then $\phi_j(x)$ will be feature map of shape $C_j \times H_j \times W_j$

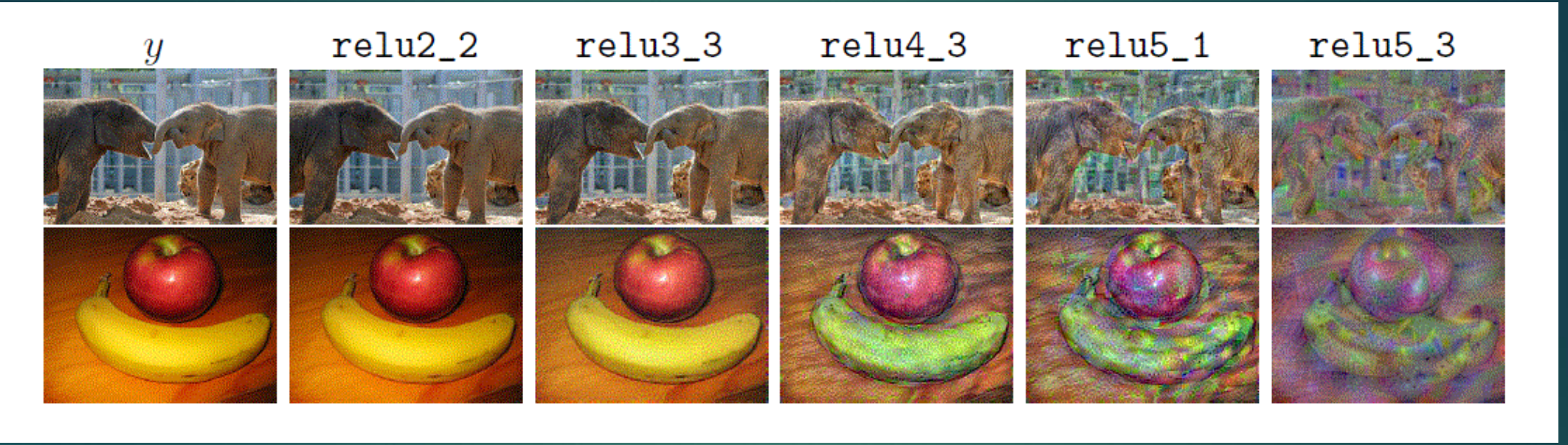
Feature Reconstruction Loss – part II

- It's (squared, normalized) Euclidean distance between feature representations:

$$l^{\phi,j}{}_{feat}(\hat{y}, y) = \frac{1}{C_j H_j W_j} \left\| \phi_j(\hat{y}) - \phi_j(y) \right\|_2^2$$

- **Minimizing** the feature reconstruction loss for early layers tends to produce images that are visually indistinguishable from y
- **Reconstruct** from higher layers, image content and overall spatial structure are preserved but color, texture, and exact shape are not

Feature Reconstruction Loss – part III



- Reconstruct from higher layers:
  - image **content** and **overall spatial structure** are preserved
  - **color**, *texture*, and **exact shape** are not

Style Reconstruction Loss

- Penalizes the output image $\hat{y}$ when it deviates in content from the target y *(till now)*
- Wish to penalize differences in style: **colors**, **textures**, common **patterns**, etc.
- Again, $\phi_j(x)$ the activations of the j-th layer of the network ϕ
- Define the Gram matrix ${G_j}^{\phi}$ to be $C_j \times C_j$ matrix:

$$ {G_j}^{\phi}(x)_{c,c'} = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \phi_j(x)_{h,w,c} \phi_j(x)_{h,w,c'} $$

Style Reconstruction Loss – part II

- Interpret $\phi_j(x)$ as giving C_j dimensional features for each point on $H_j \times W_j$ grid
- ${G_j}^{\phi}(x)$ is proportional to the uncentered covariance of the C_j dimensional features (treating each grid location as an independent sample)
- It thus captures information about *which features tend to activate together*
- The Gram matrix can be computed efficiently

Style Reconstruction Loss – part III

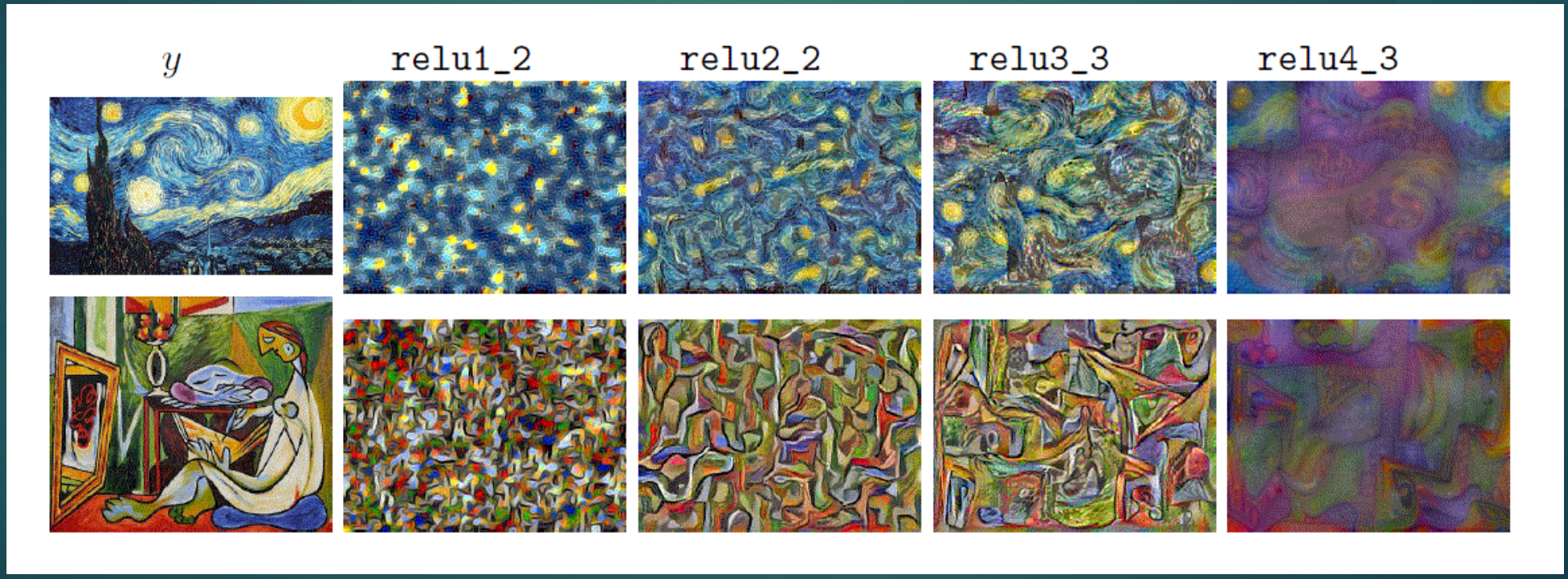
- The squared Frobenius norm of the difference between the Gram matrices of the output and target images:

$$l^{\phi,j}{}_{style}(\hat{y}, y) = \left\| G_j{}^{\phi}(\hat{y}) - G_j{}^{\phi}(y) \right\|_F{}^2$$

- Well-defined even when $\hat{y}$ and y have different sizes
- Generating an image $\hat{y}$ that minimizes the style reconstruction loss
  - preserves **stylistic features** from the target image
  - does not preserve its **spatial structure**
- Reconstructing from higher layers transfers *larger-scale structure* from the target image

Style Reconstruction Loss – part IIII

- To perform style reconstruction from a set of layers J rather than a single layer j we define ${l^{\phi,j}}_{style}(\hat{y}, y)$ to be the sum of losses for each layer $j \in J$

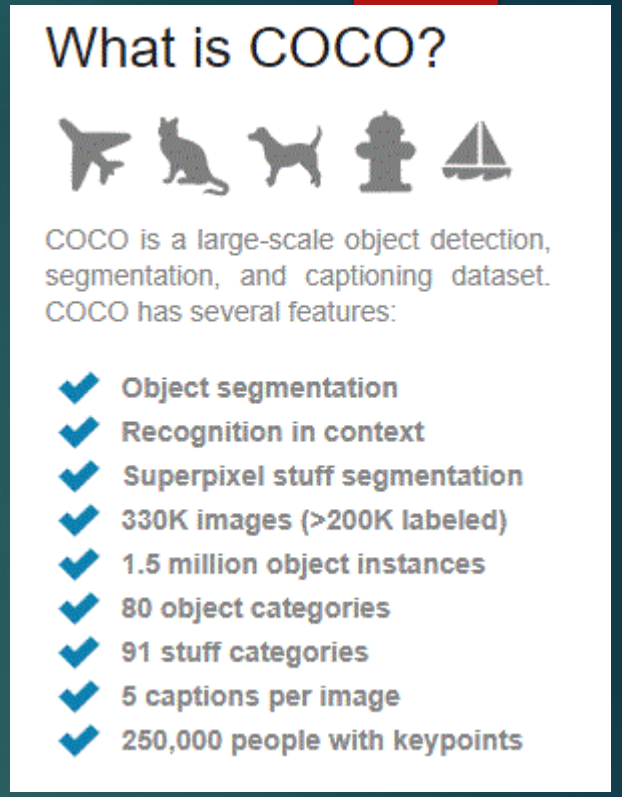


Additional Simple Loss Functions

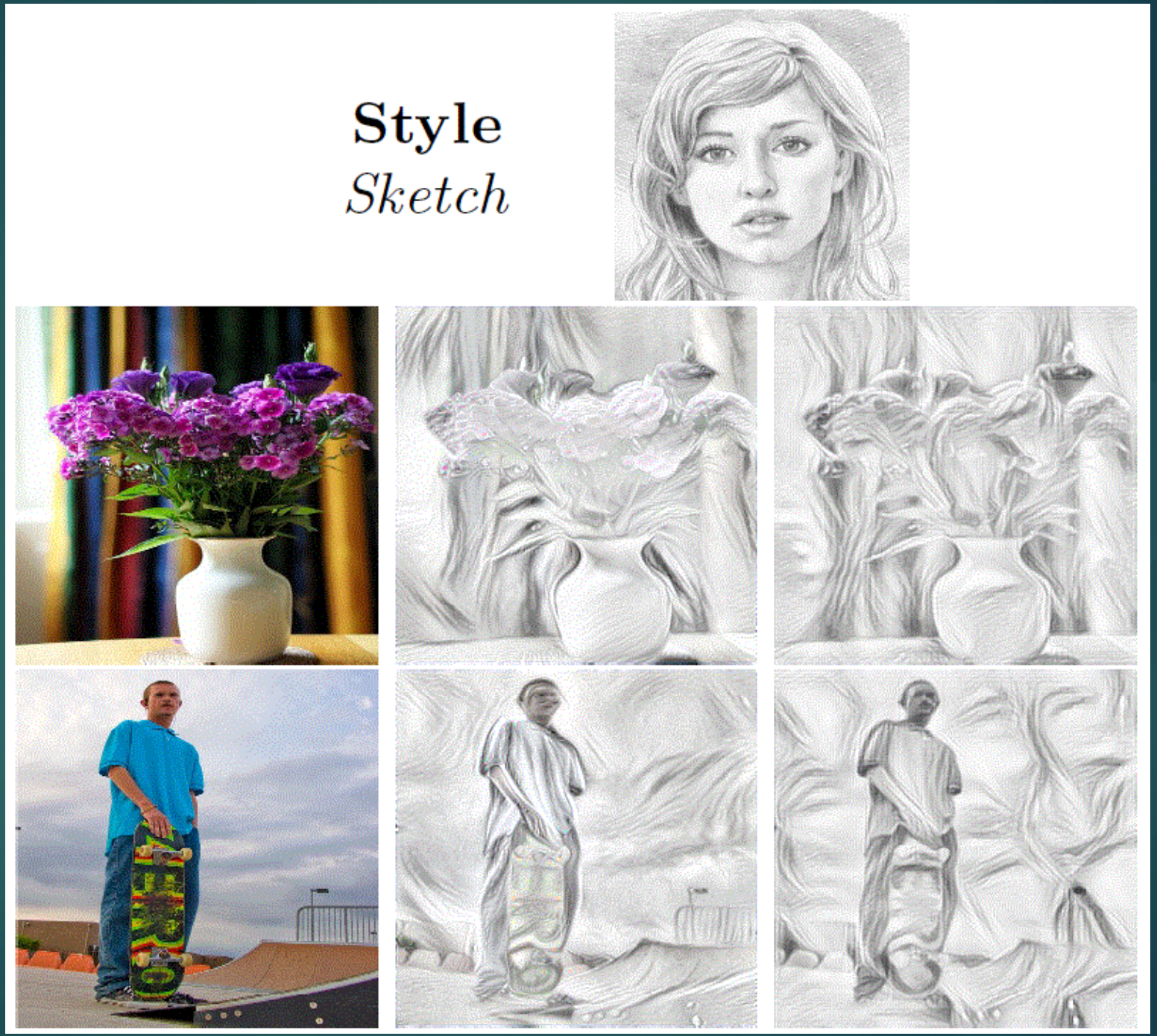
- In addition to the perceptual losses defined above, we also define two simple loss functions that depend only on *low-level pixel information*:
- **Pixel loss**:
  - The (normalized) Euclidean distance between the output image $\hat{y}$ and the target y
  - $l_{pixel}(\hat{y}, y) = \|\hat{y} - y\|_2^{\,2}/CHW$ (both images have shape $C \times H \times W$)
- **Total Variation Regularization**:
  - Used to encourage spatial smoothness in the output image $\hat{y}$
  - make use of total variation regularizer $l_{TV}(\hat{y})$

Experiments

- Train style transfer network on MS-COCO dataset
- 80k train images, 256x256
- Batch size 4, 40k iterations
- Adam, lr=$1x10^{-3}$
- Do not use weight decay or dropout
- Compute feature reconstruction loss at layer *relu3_3*
- Compute style reconstruction loss at layers *relu1_2, relu2_2, relu3_3, relu4_3*
- Torch, cuDNN
- 4 hours training on a single GTX Titan X GPU



Results:



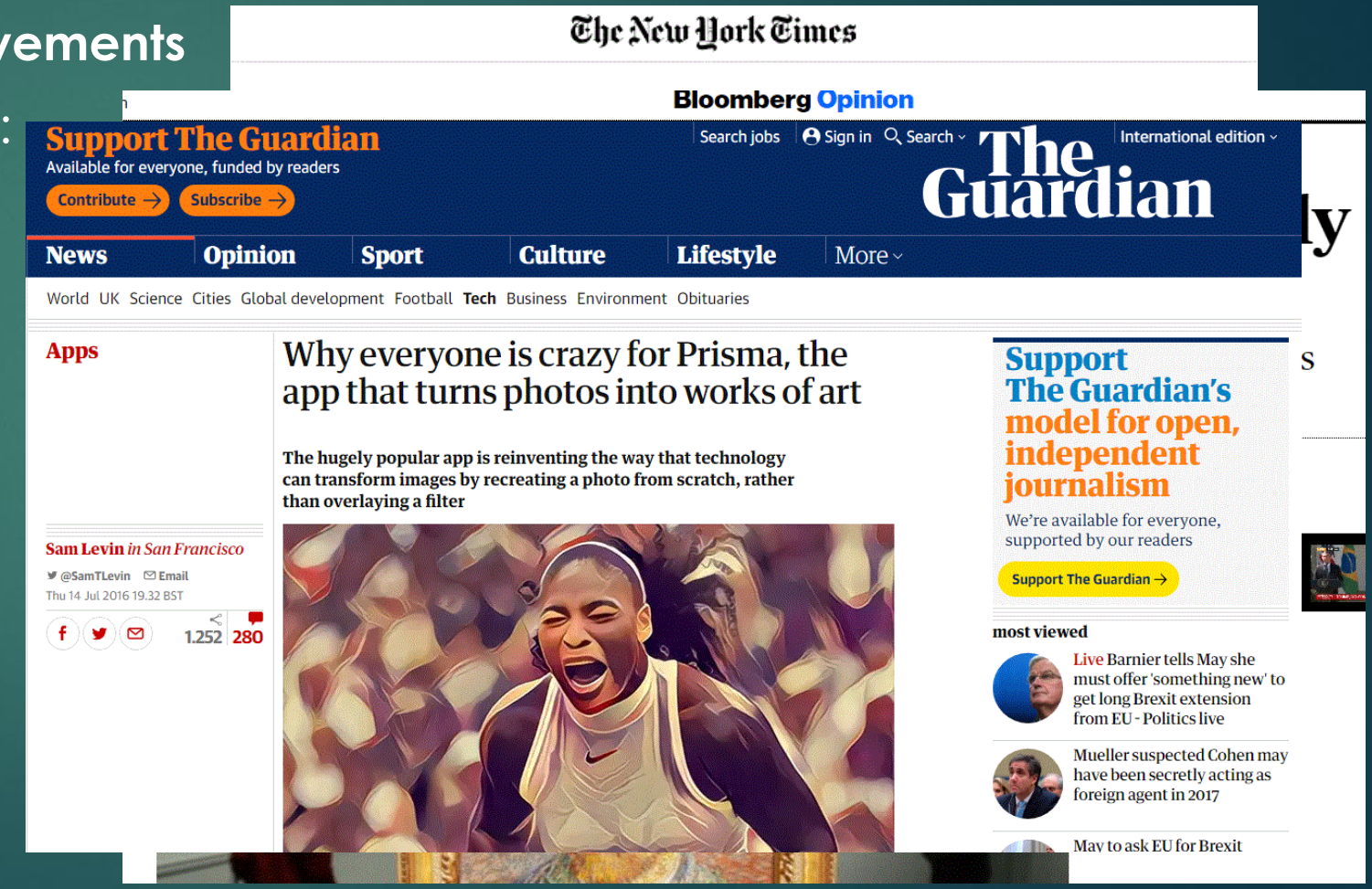
But first, let me take a selfie!

Johnson code

```
File Edit Options Buffers Tools C++ Help
git config --global http.proxy http://proxyuser:proxypwd@147.91.1.42:8080
# in a terminal, run the commands WITHOUT sudo
git clone https://github.com/torch/distro.git ~/torch --recursive
cd ~/torch; bash install-deps;
sudo apt-get install cmake  # install cmake
./install.sh

# source ~/.bashrc

cd Desktop
git clone https://github.com/jcjohnson/fast-neural-style.git ./JCJOHNSON

luarocks install torch
luarocks install nn
luarocks install image
luarocks install lua-cjson

bash models/download_style_transfer_models.sh  # Here I found an error
th fast_neural_style.lua -model models/eccv16/starry_night.t7 -input_image images/content/chicago.jpg -output_image out.png

# export http_proxy=http://147.91.1.42:8080
# export https_proxy=https://147.91.1.42:8080
# fix image library: https://github.com/jcjohnson/neural-style/issues/140
```

Further work (some ideas)

- Use on all paintings of one author (period of painting)
- Image belonging to **Art Movements**
- Mobile App (Already Done):



Questions?

Thank You