



Deep Learning for Vision & Language

Feature-space Optimization: Adversarial Examples, GANs, Style Transfer



RICE UNIVERSITY

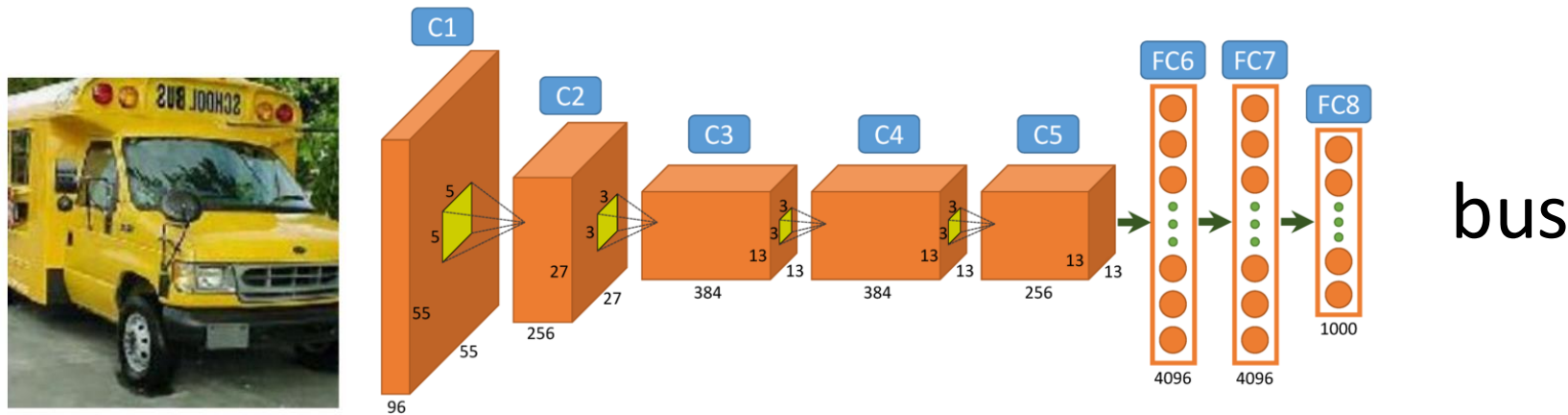


What we have been doing: Optimize weights in the network to predict bus (correct class).

I

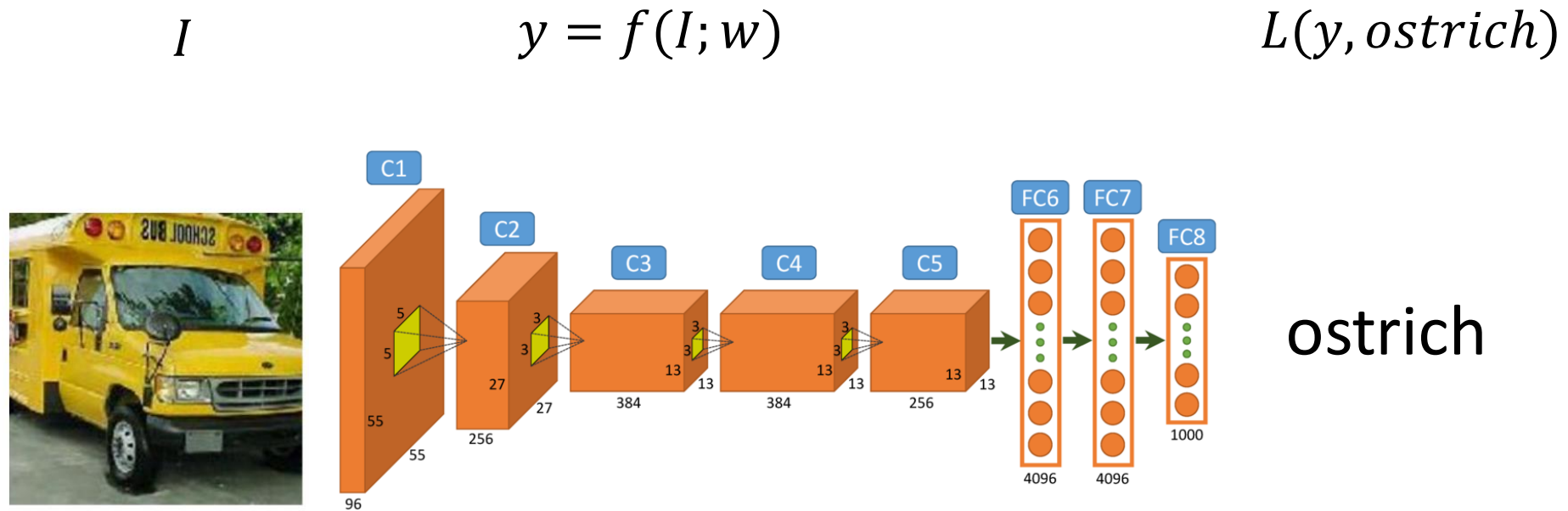
$$y = f(I; w)$$

$$L(y, bus)$$



$$w = w - \lambda \frac{\partial L}{\partial w}$$

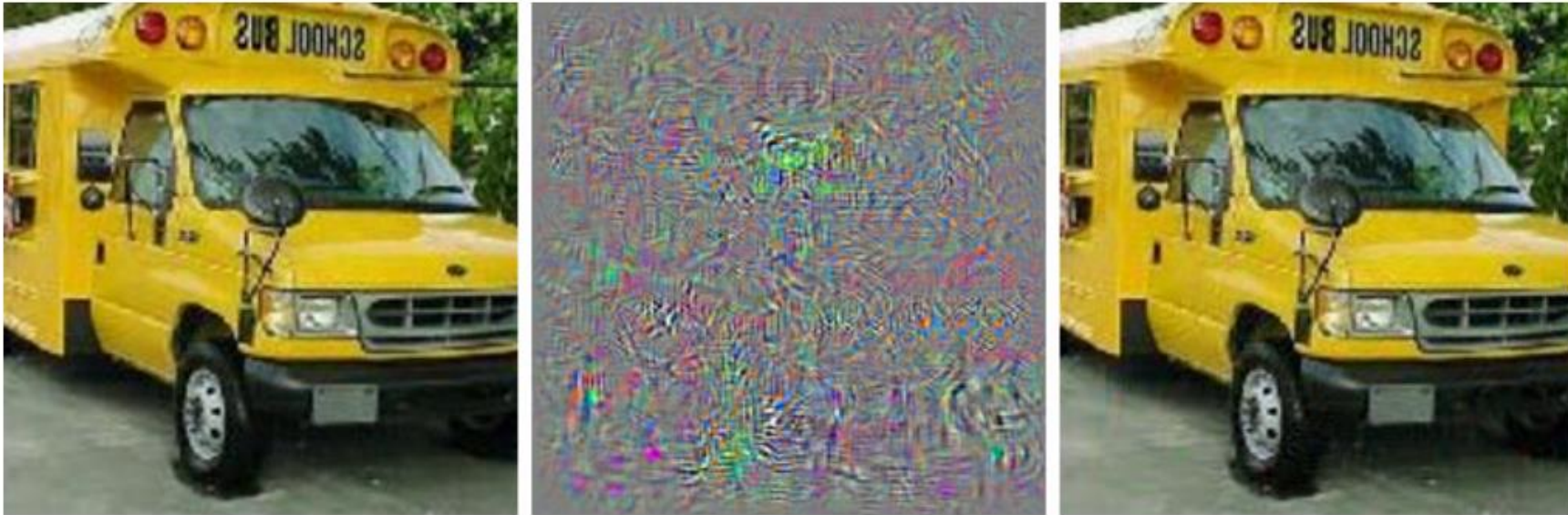
New Idea: Create Adversarial Inputs by optimizing the input image to predict ostrich (wrong class).



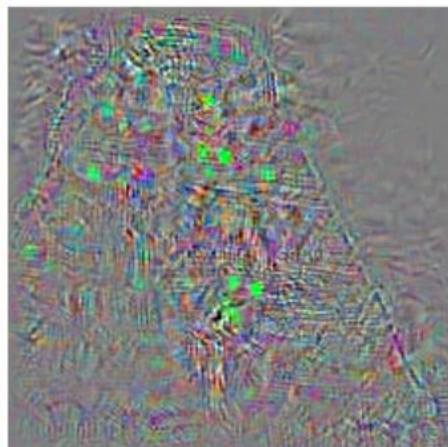
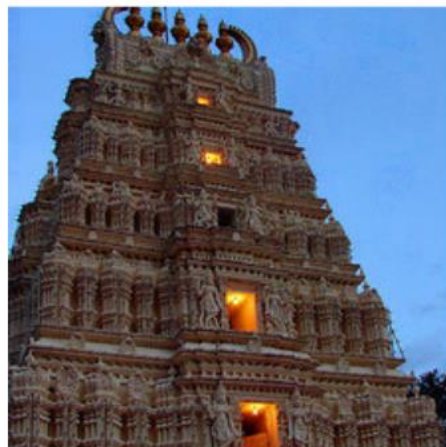
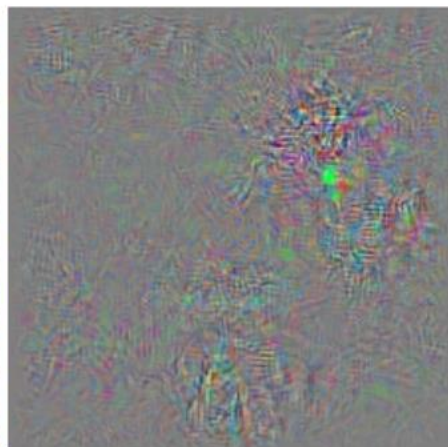
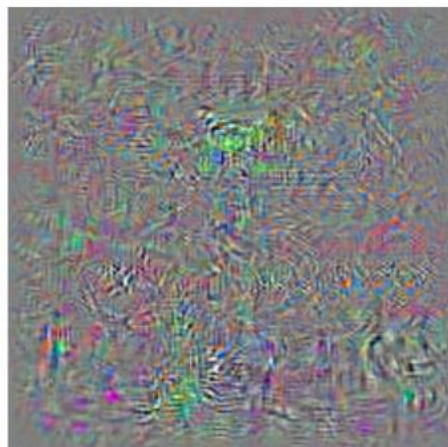
$$I = I - \lambda \frac{\partial L}{\partial I}$$

Work on Adversarial examples by Goodfellow et al. , Szegedy et. al., etc.

Convnets (optimize input to predict ostrich)



Work on Adversarial examples by Goodfellow et al. , Szegedy et. al., etc.



All get
predicted
as ostrich



Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images

Anh Nguyen, Jason Yosinski, Jeff Clune, 2014

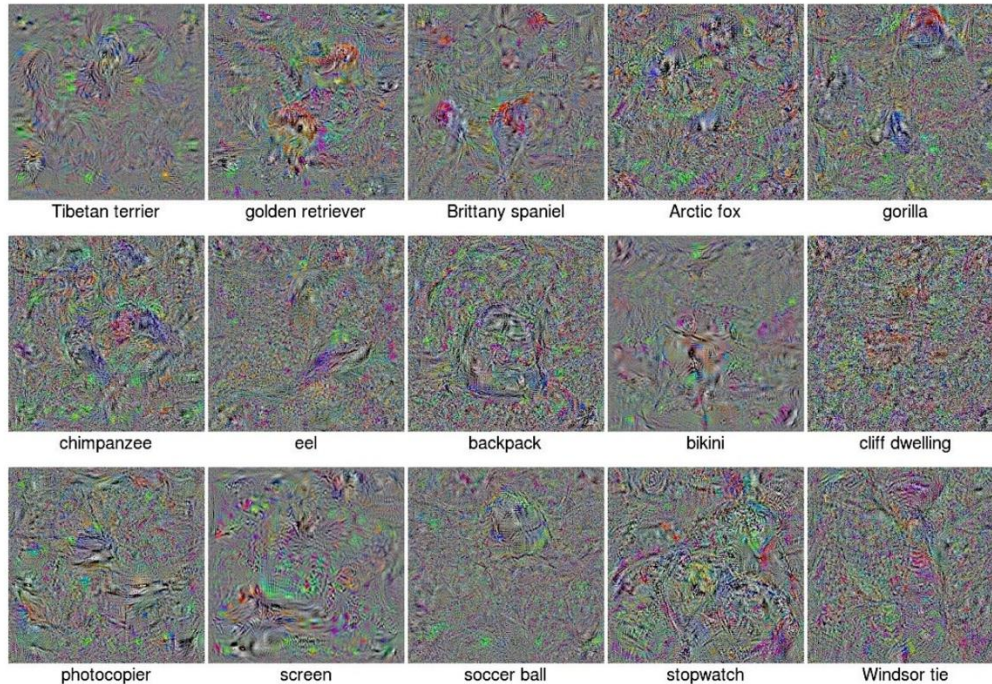
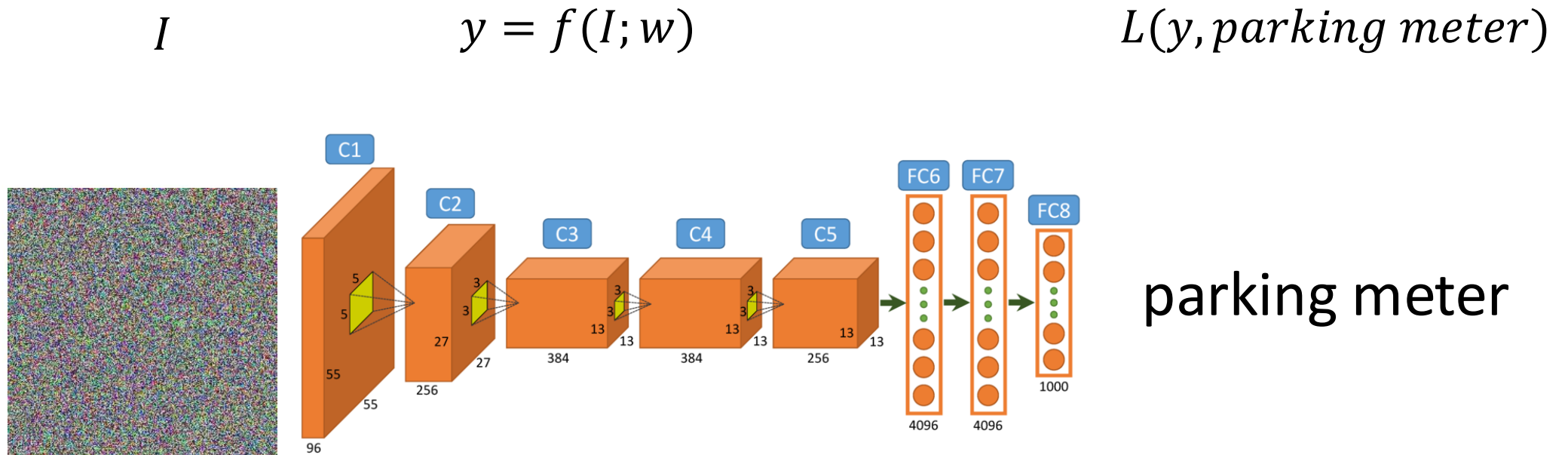


Figure 13. Images found by maximizing the softmax output for classes via gradient ascent [11, 26]. Optimization begins at the ImageNet mean (plus small Gaussian noise to break symmetry) and continues until the DNN confidence for the target class reaches 99.99%. Images are shown with the mean subtracted. Adding regularization makes images more recognizable but results in slightly lower confidence scores (see supplementary material).

New Idea: Create Adversarial Inputs by optimizing the input image to predict ostrich (wrong class).



$$I = I - \lambda \frac{\partial L}{\partial I}$$

Work on Adversarial examples by Goodfellow et al. , Szegedy et. al., etc.



parking meter: 0.999679

Total Variation Regularization

A second richer regulariser is *total variation* (TV) $\mathcal{R}_{V^\beta}(\mathbf{x})$, encouraging images to consist of piece-wise constant patches. For continuous functions (or distributions) $f : \mathbb{R}^{H \times W} \supset \Omega \rightarrow \mathbb{R}$, the TV norm is given by:

$$\mathcal{R}_{V^\beta}(f) = \int_{\Omega} \left(\left(\frac{\partial f}{\partial u}(u, v) \right)^2 + \left(\frac{\partial f}{\partial v}(u, v) \right)^2 \right)^{\frac{\beta}{2}} du dv$$

where $\beta = 1$. Here images are discrete ($\mathbf{x} \in \mathbb{R}^{H \times W}$) and the TV norm is replaced by the finite-difference approximation:

$$\mathcal{R}_{V^\beta}(\mathbf{x}) = \sum_{i,j} \left((x_{i,j+1} - x_{ij})^2 + (x_{i+1,j} - x_{ij})^2 \right)^{\frac{\beta}{2}}.$$



Figure 1. **What is encoded by a CNN?** The figure shows five possible reconstructions of the reference image obtained from the 1,000-dimensional code extracted at the penultimate layer of a reference CNN[13] (before the softmax is applied) trained on the ImageNet data. From the viewpoint of the model, all these images are practically equivalent. This image is best viewed in color/screen.

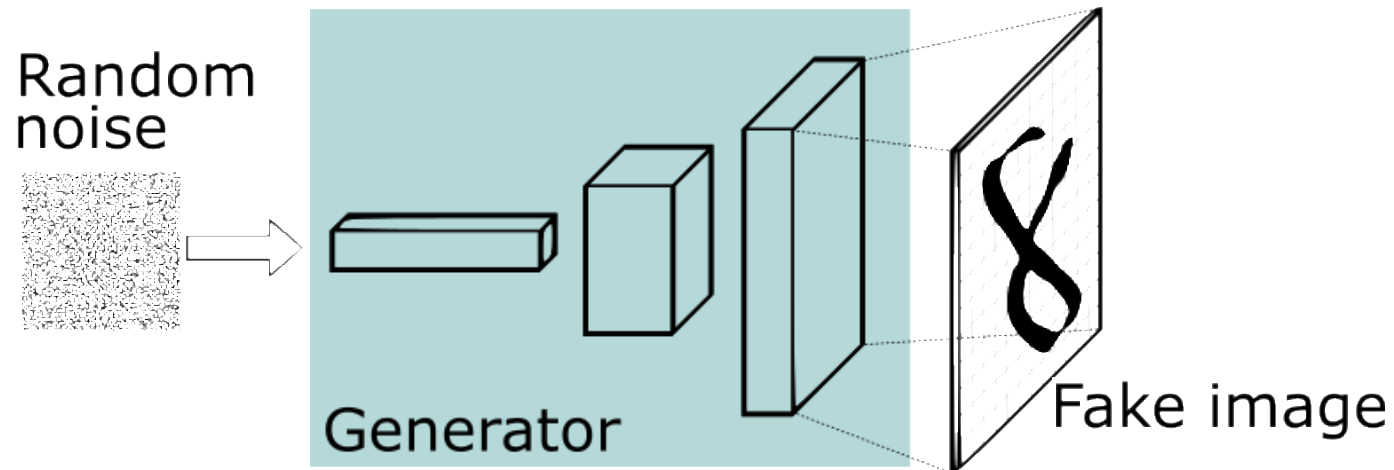
Taking the idea to the extreme: Google's DeepDream



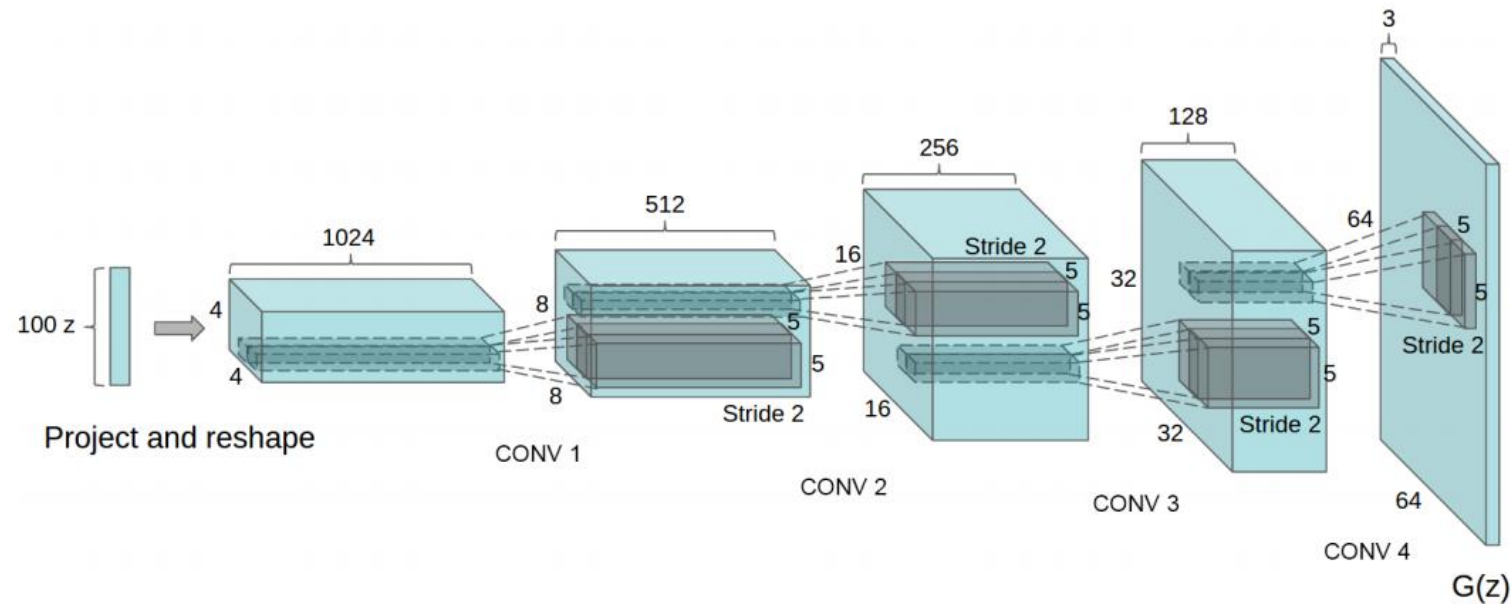
<https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>
Generate your own in Pytorch: https://github.com/XavierLinNow/deepdream_pytorch

Generative Adversarial Networks (GAN)

[Goodfellow et al 2014]



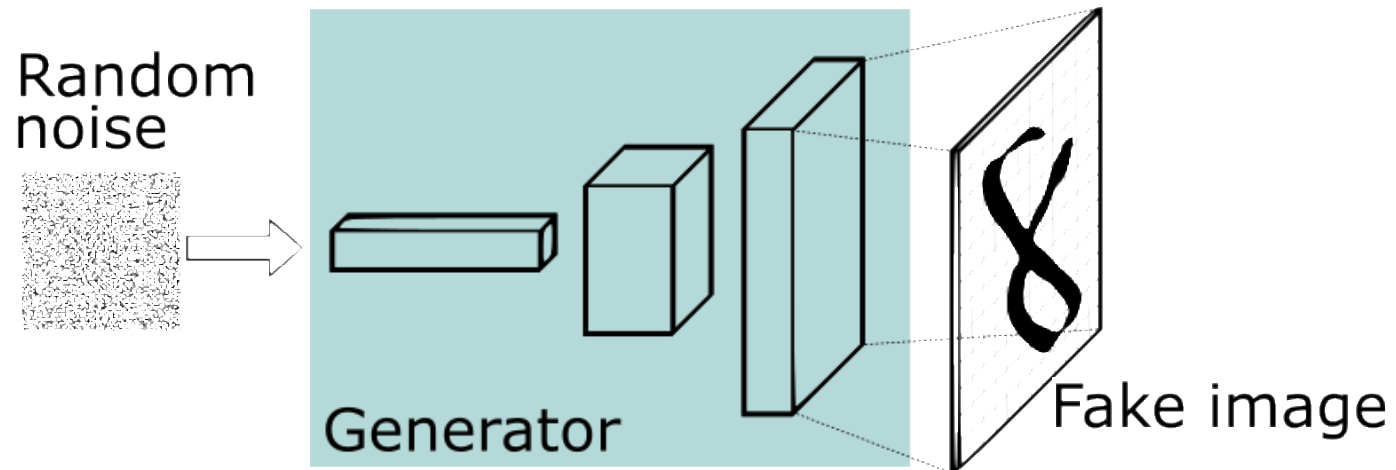
Generative Network (closer look)



Radford et. al. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. ICLR 2016

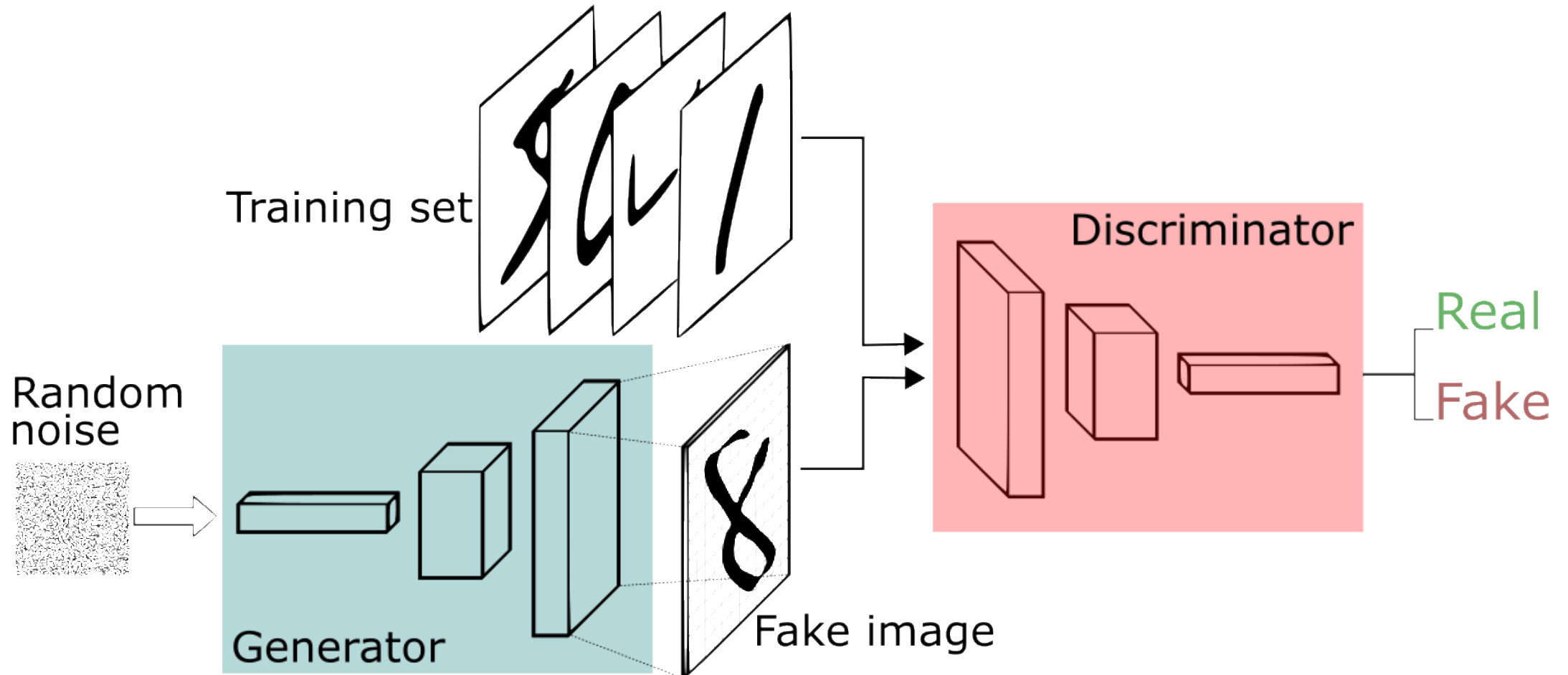
Generative Adversarial Networks (GAN)

[Goodfellow et al.]



Generative Adversarial Networks (GAN)

[Goodfellow et al.]



Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k , is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Sample minibatch of m examples $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\mathbf{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m \left[\log D(\mathbf{x}^{(i)}) + \log \left(1 - D(G(\mathbf{z}^{(i)})) \right) \right].$$

end for

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log \left(1 - D(G(\mathbf{z}^{(i)})) \right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k , is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Sample minibatch of m examples $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\mathbf{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m \left[\log D(\mathbf{x}^{(i)}) + \log \left(1 - D(G(\mathbf{z}^{(i)})) \right) \right].$$

end for

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log \left(1 - D(G(\mathbf{z}^{(i)})) \right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Update
Discriminator
D

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k , is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Sample minibatch of m examples $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\mathbf{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m \left[\log D(\mathbf{x}^{(i)}) + \log \left(1 - D(G(\mathbf{z}^{(i)})) \right) \right].$$

end for

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log \left(1 - D(G(\mathbf{z}^{(i)})) \right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Update
Generator
G

Until
Desirable
Results are
Achieved?

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k , is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Sample minibatch of m examples $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\mathbf{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m \left[\log D(\mathbf{x}^{(i)}) + \log \left(1 - D(G(\mathbf{z}^{(i)})) \right) \right].$$

end for

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Update the generator by descending its stochastic gradient:

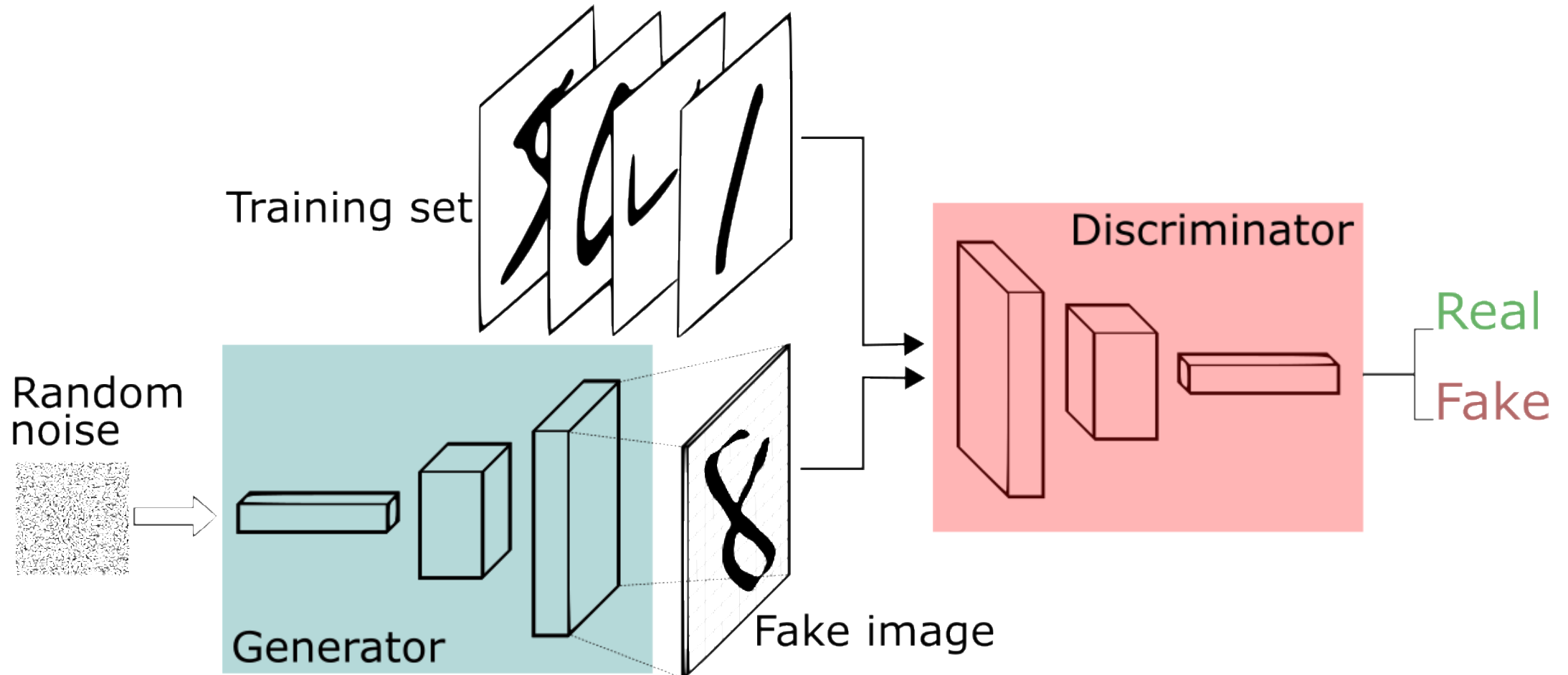
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log \left(1 - D(G(\mathbf{z}^{(i)})) \right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Generative Adversarial Networks (GAN)

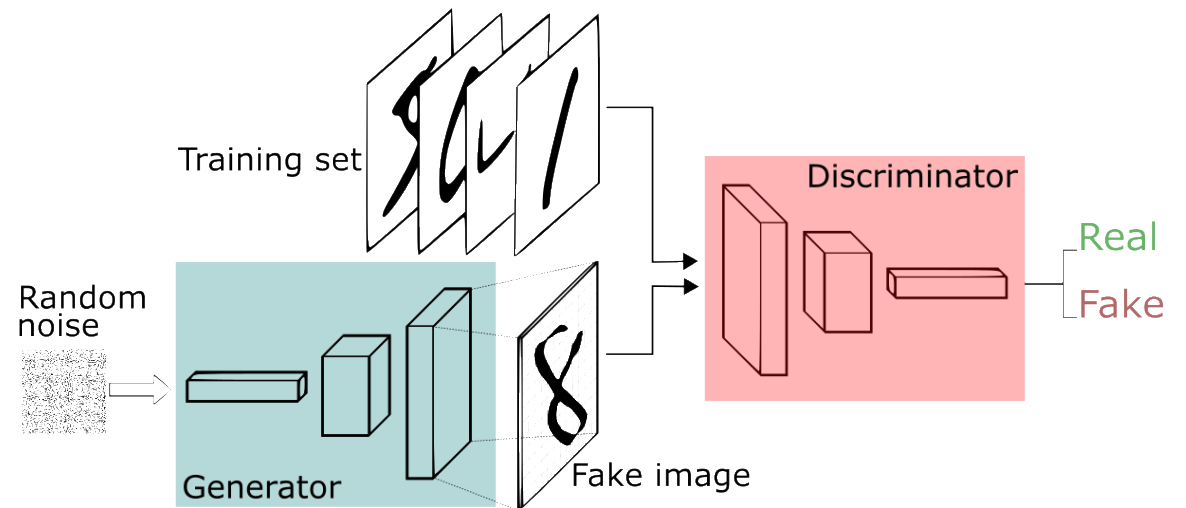
[Goodfellow et al.]



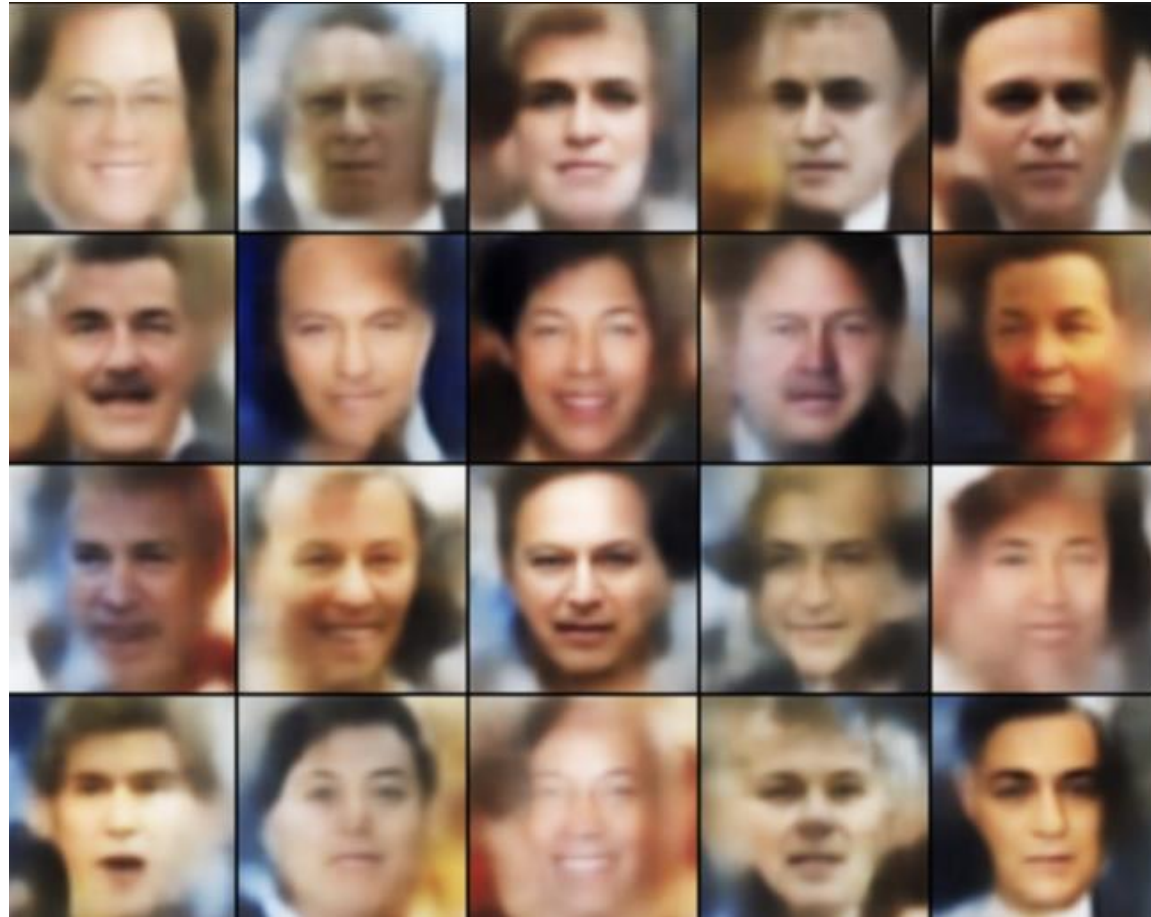
Generative Adversarial Networks (GAN)

[Goodfellow et al.]

- GANs are hard to train, loss for the discriminator and generator might fluctuate.
- There are many choices for loss, and other auxiliary signals.
- Training of these models is even less well understood than for other deep models.



Basic GAN Results (Example implementation is provided in Pytorch's examples)



NVidia's progressive GANs ICLR 2018



Google's BigGAN



Google's BigGAN

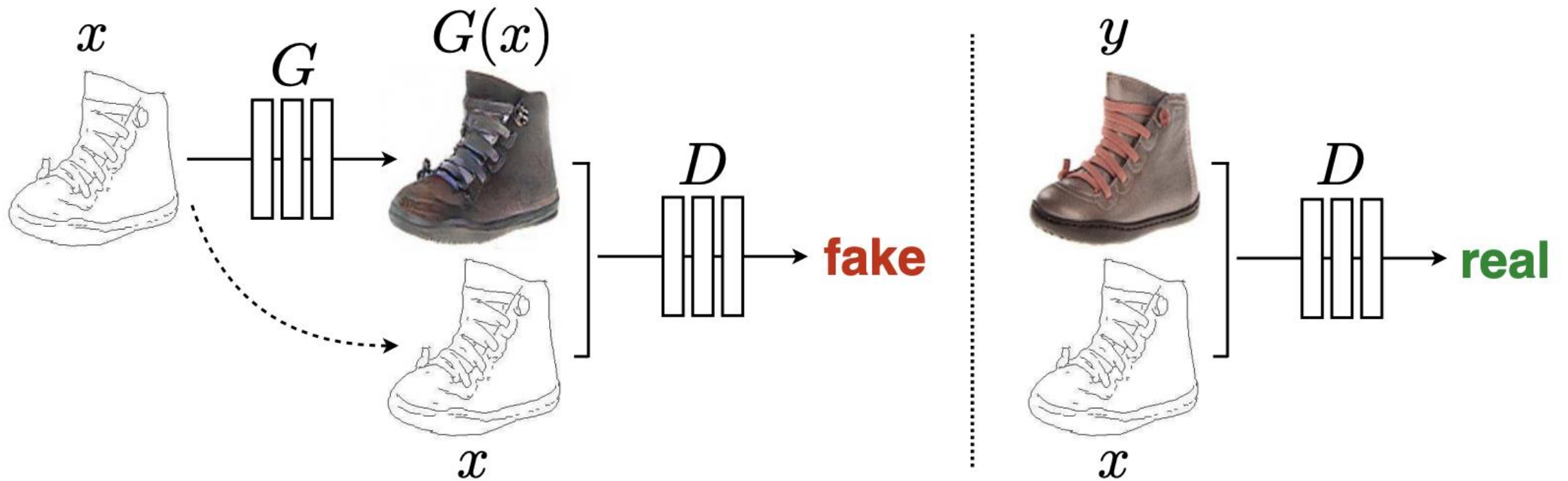
Teddy Bear



Microphone



Conditional GANs: Input is not just Noise



Conditional GANs: Also Hard to Train

L1

L1+cGAN

Encoder-decoder



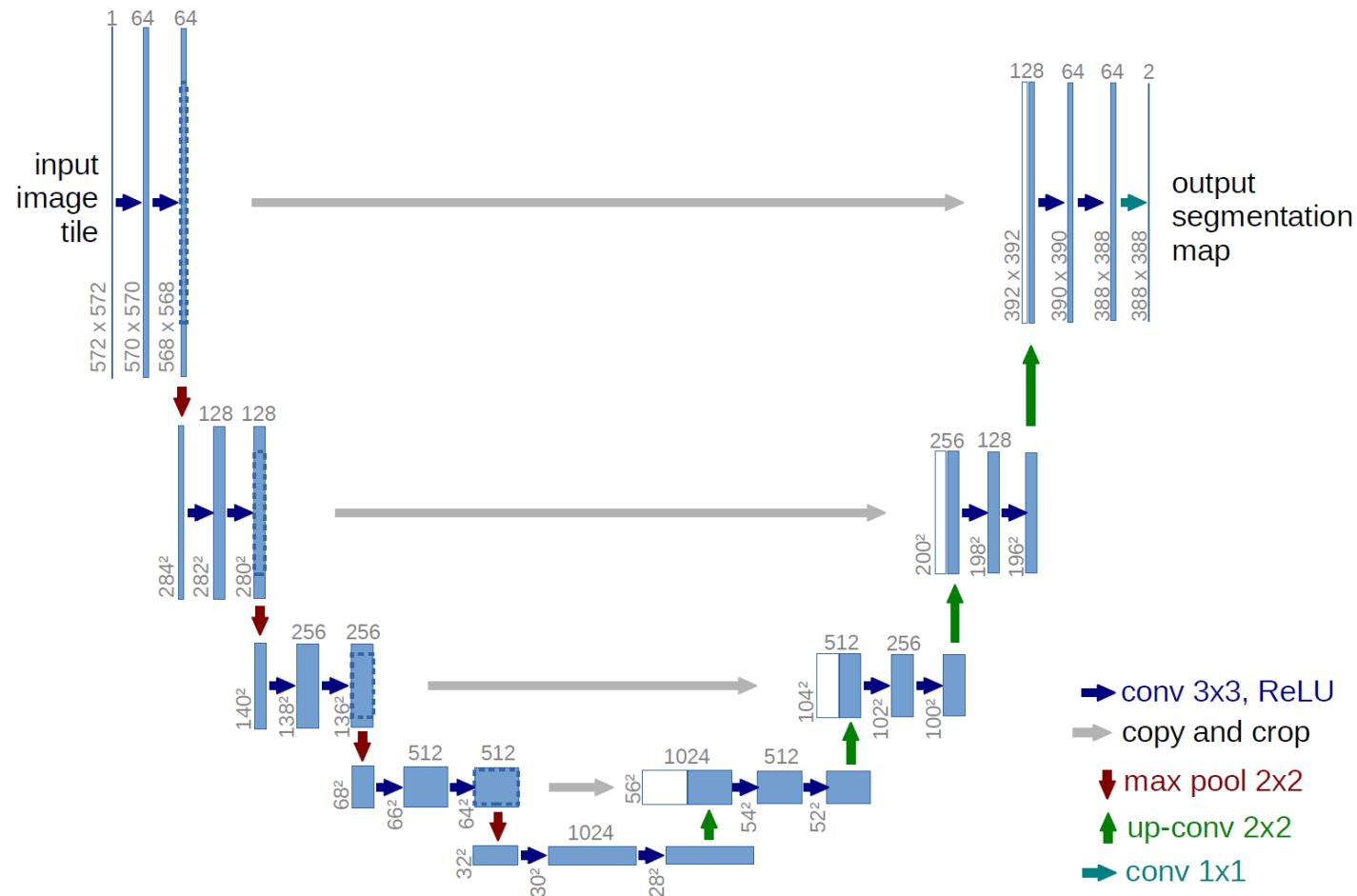
U-Net



Result they
obtained with
a regular Fully
Convolutional
Network

Result they
obtained with a
U-Net network
(with skip-
connections)

Conditional GANs: Also Hard to Train



Conditional GANs / Text-conditioned

AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks

Tao Xu^{*1}, Pengchuan Zhang², Qiuyuan Huang²,
Han Zhang³, Zhe Gan⁴, Xiaolei Huang¹, Xiaodong He²

¹Lehigh University ²Microsoft Research ³Rutgers University ⁴Duke University
{tax313, xih206}@lehigh.edu, {penzhan, qihua, xiaohe}@microsoft.com
han.zhang@cs.rutgers.edu, zhe.gan@duke.edu

Conditional GANs / Text-conditioned

Generative Adversarial Text to Image Synthesis

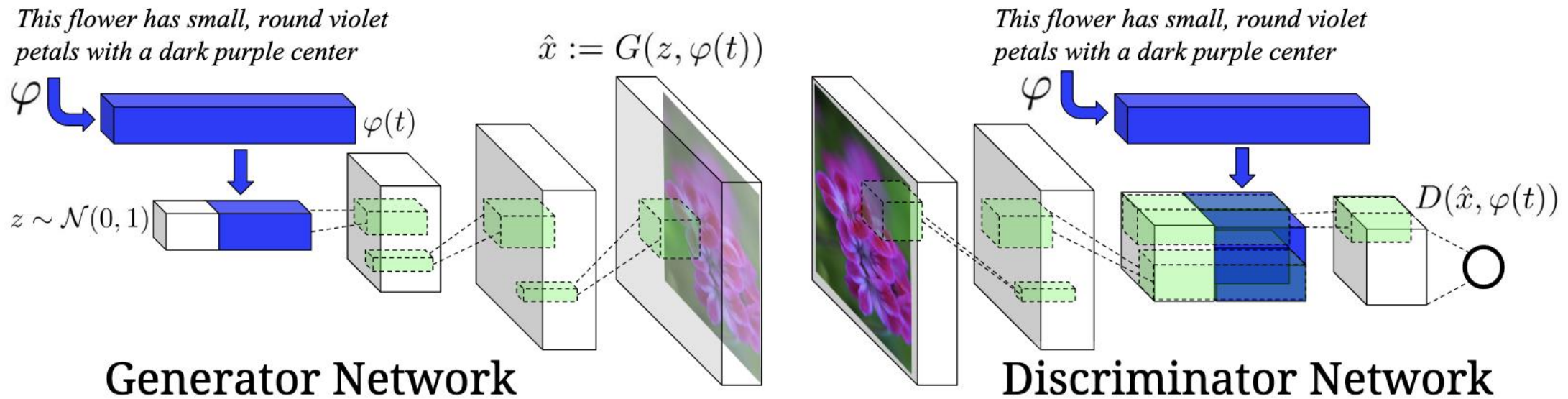
Scott Reed, Zeynep Akata, Xincheng Yan, Lajanugen Logeswaran
Bernt Schiele, Honglak Lee

REEDSCOT¹, AKATA², XCYAN¹, LLAJAN¹
SCHIELE², HONGLAK¹

¹ University of Michigan, Ann Arbor, MI, USA (UMICH.EDU)

² Max Planck Institute for Informatics, Saarbrücken, Germany (MPI-INF.MPG.DE)

Conditional GANs / Text-conditioned

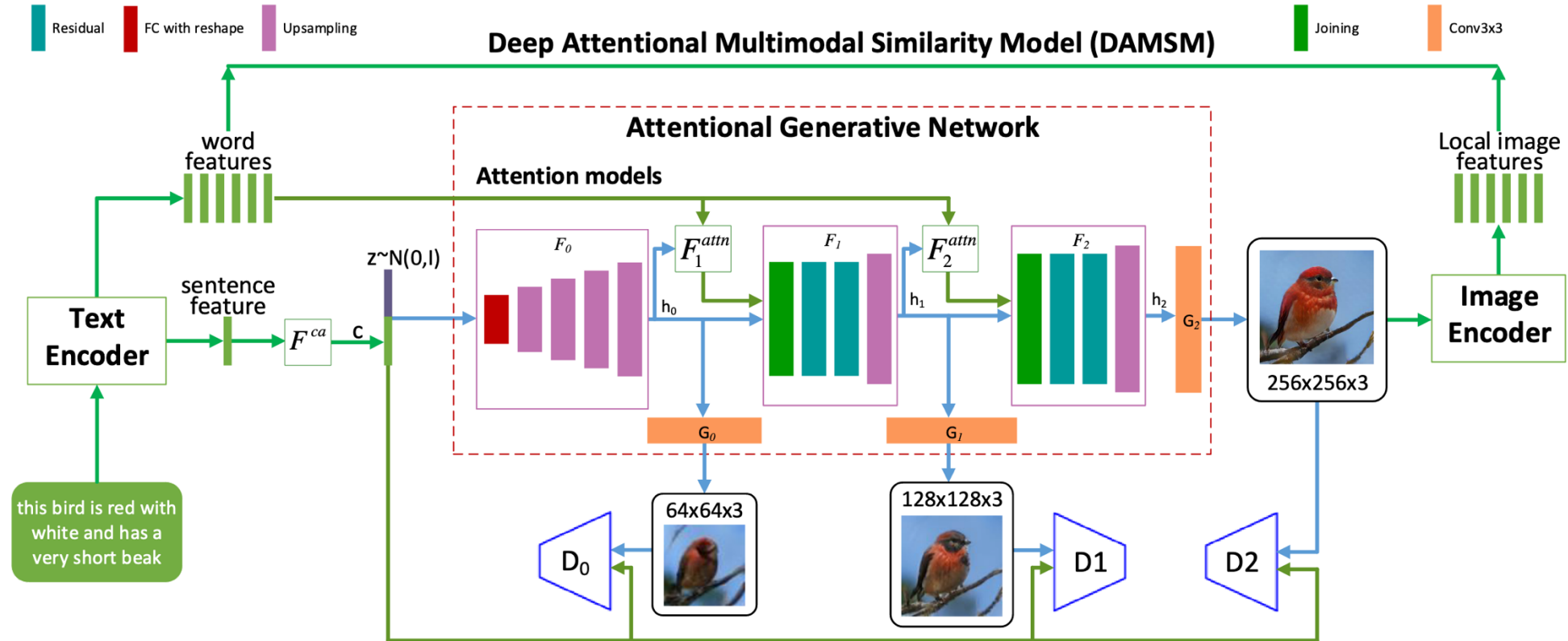


Conditional GANs / Text-conditioned

this small bird has a pink
breast and crown, and black
primaries and secondaries.



Conditional GANs / Text-conditioned



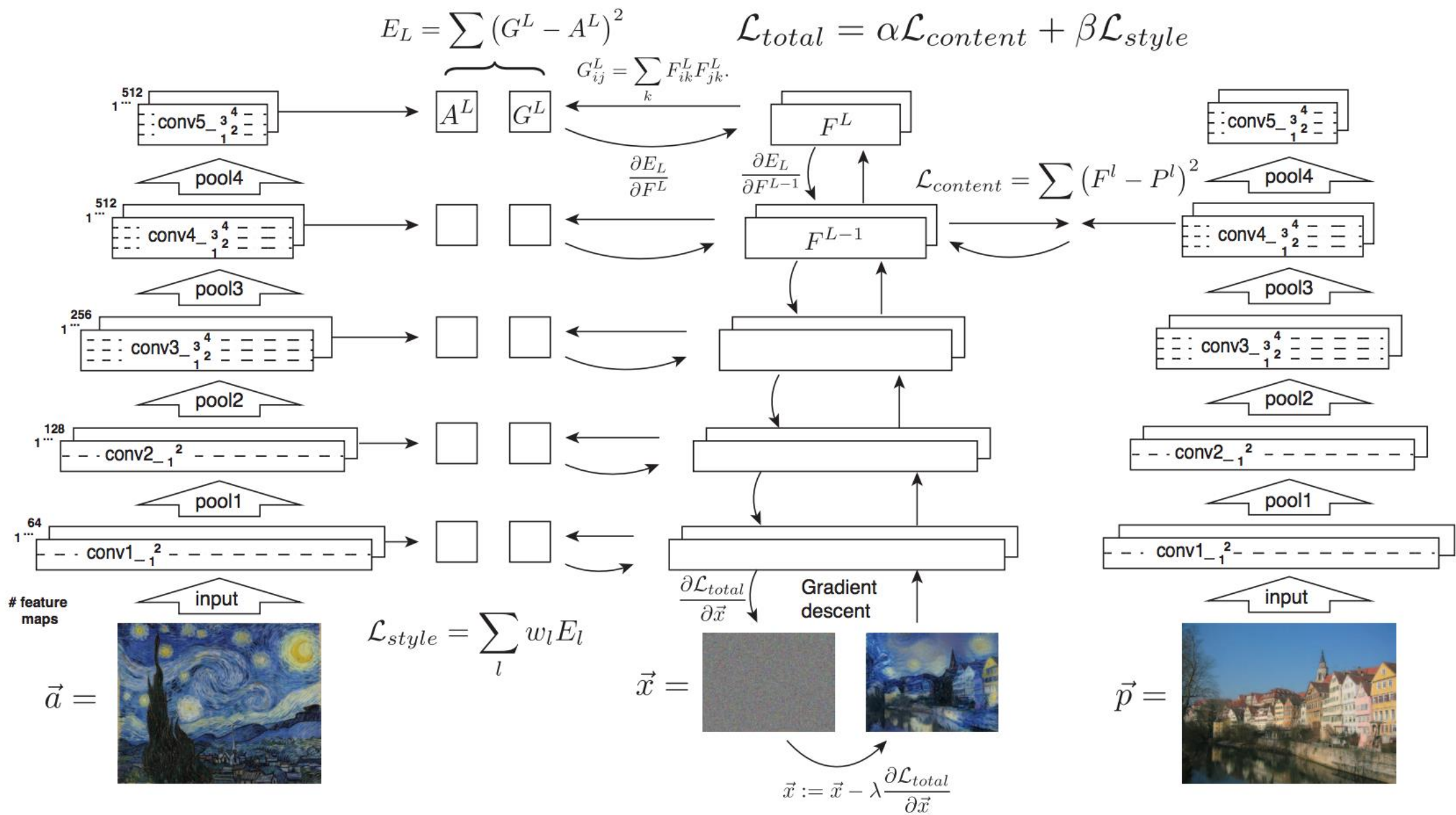
Conditional GANs / Text-conditioned

this bird is red with white and has a very short beak

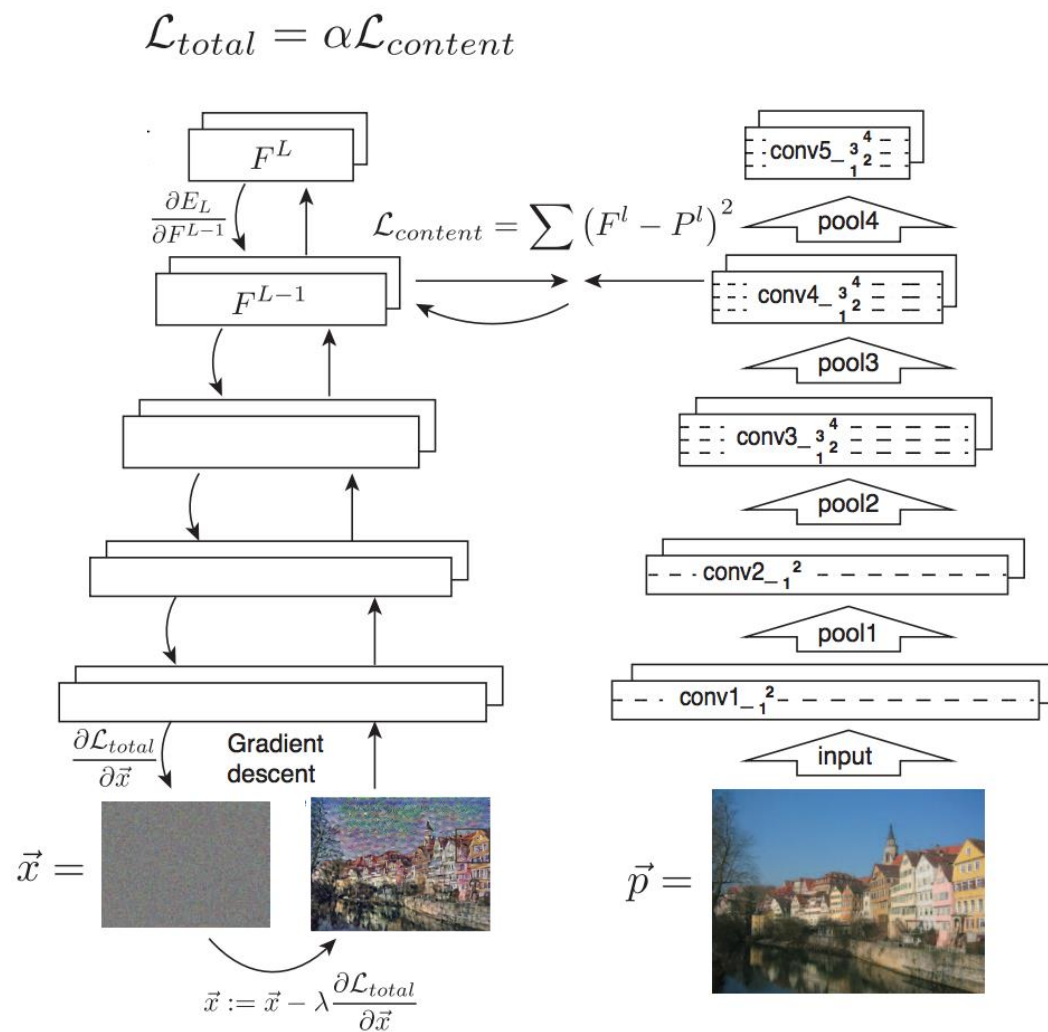


More on the Idea of Feature Space Optimization

Gatys et. al. Image Style Transfer Using Convolutional Neural Networks. CVPR 2016

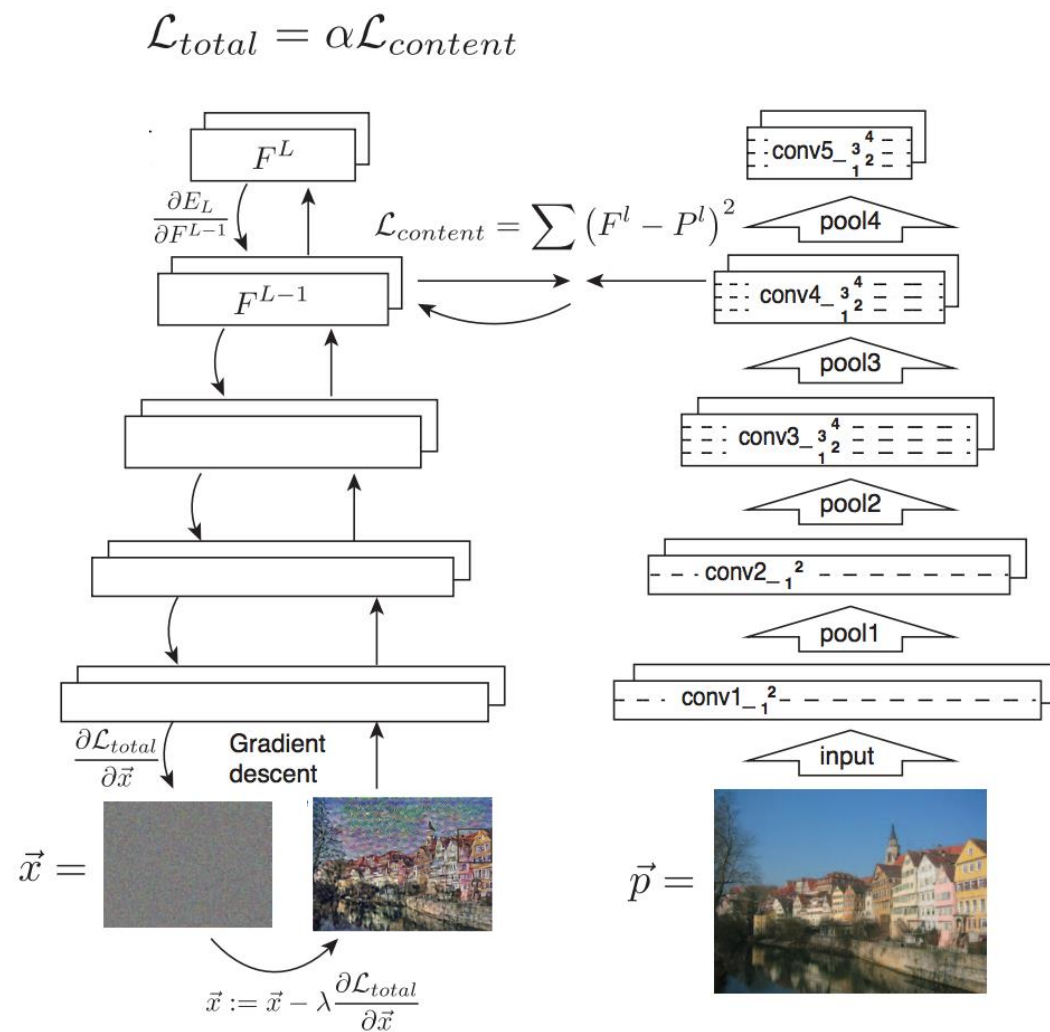


Idea 1: Image Reconstruction from Features

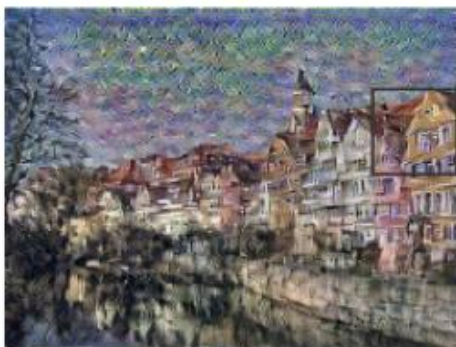


Idea 1: Image Reconstruction from Features

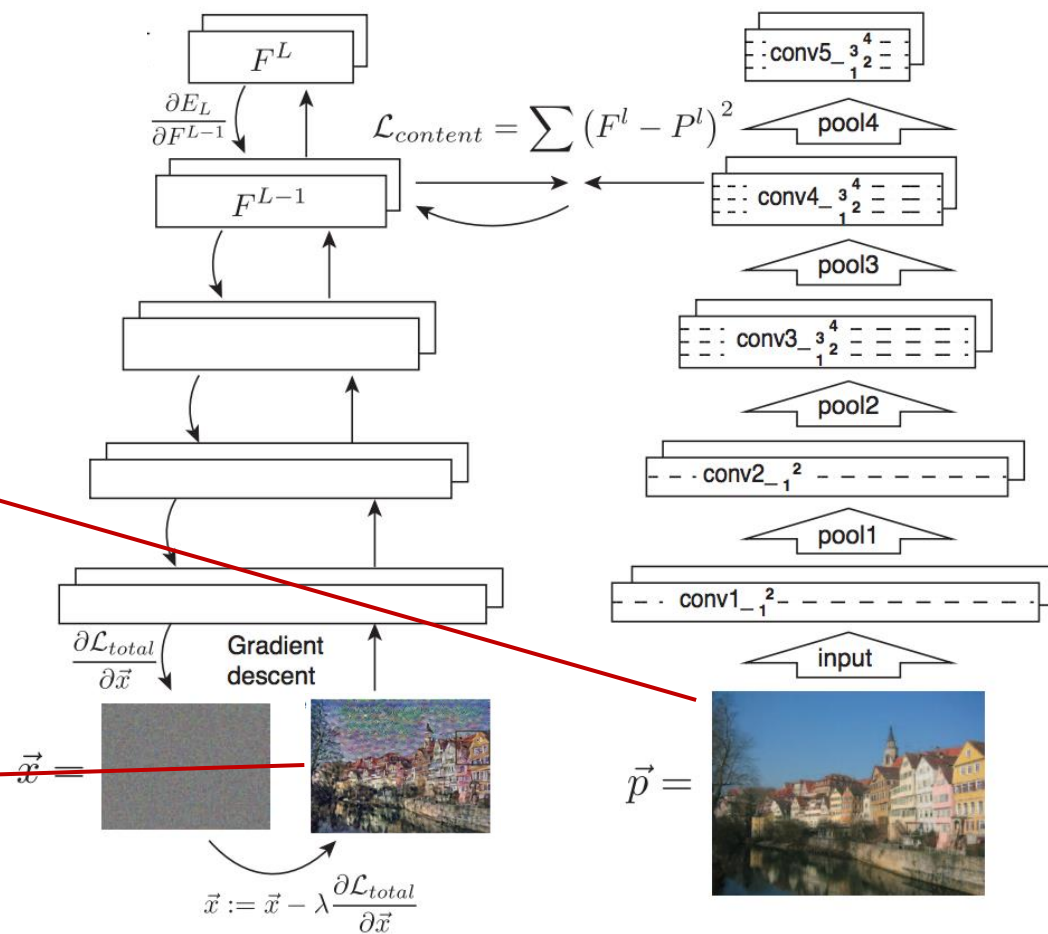
$$\mathcal{L}_{content} = \sum (F^l - P^l)^2$$



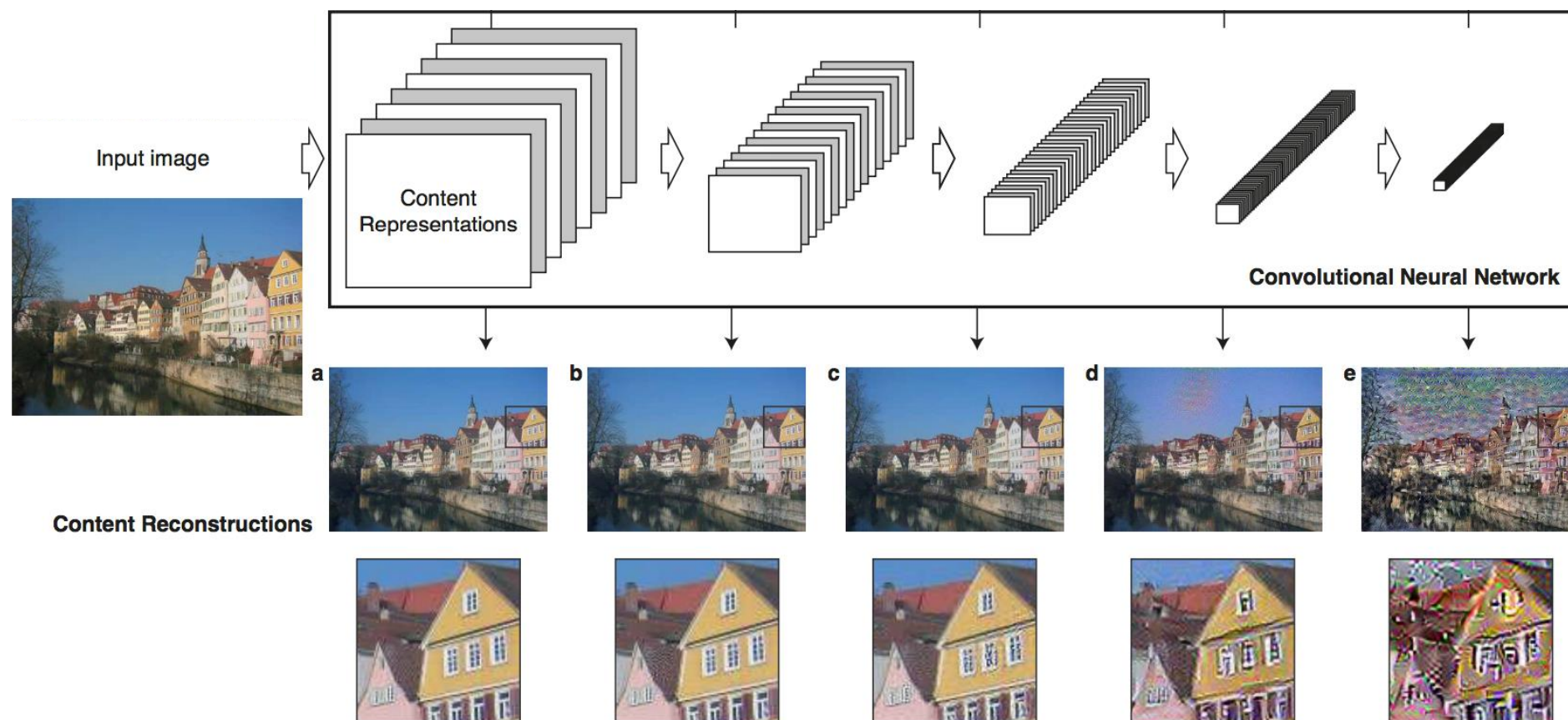
Idea 1: Image Reconstruction from Features

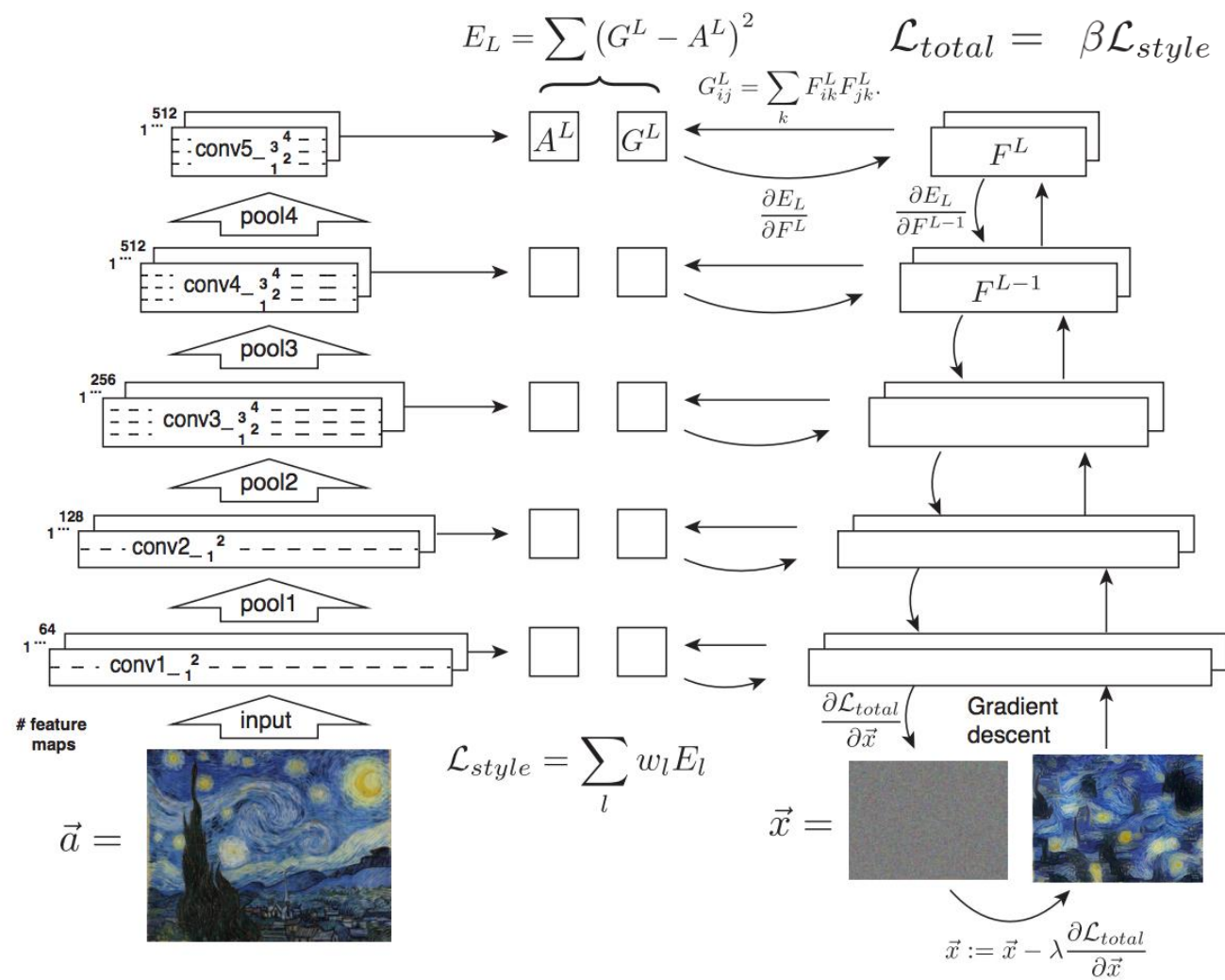


$$\mathcal{L}_{total} = \alpha \mathcal{L}_{content}$$

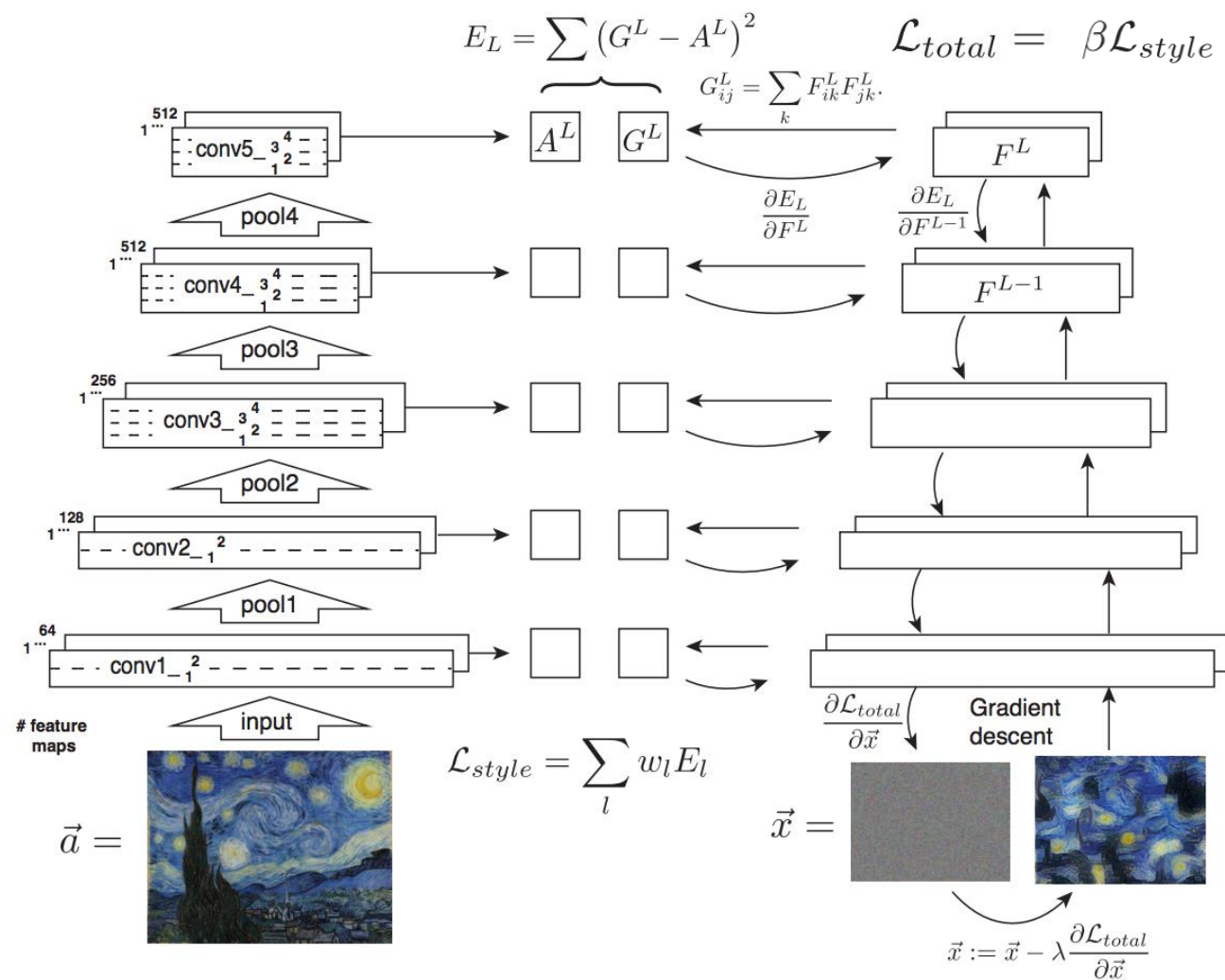


$$\mathcal{L}_{content} = \sum (F^l - P^l)^2$$





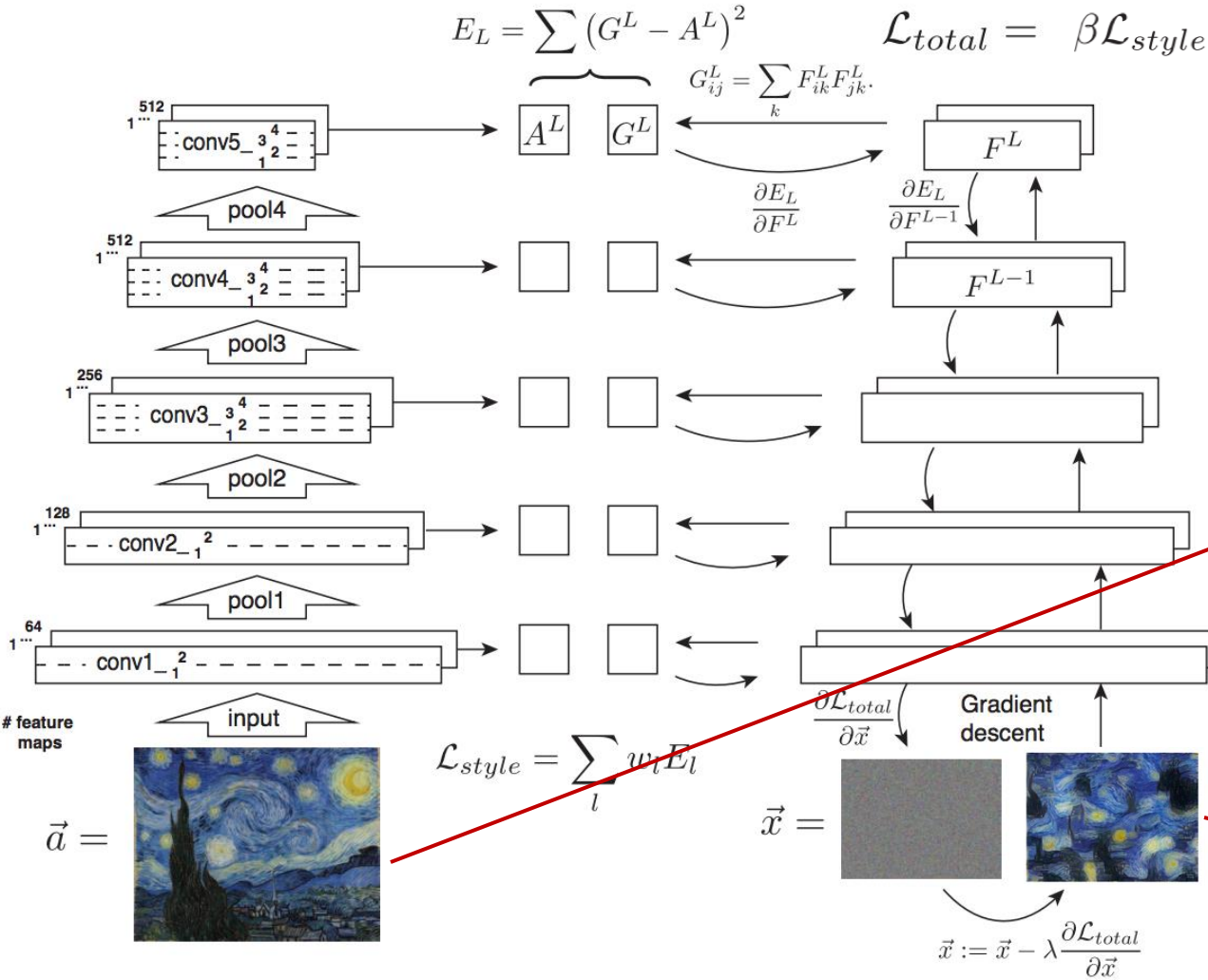
Idea 2:
Backpropagation of
Style



Idea 2:
Backpropagation of
Style

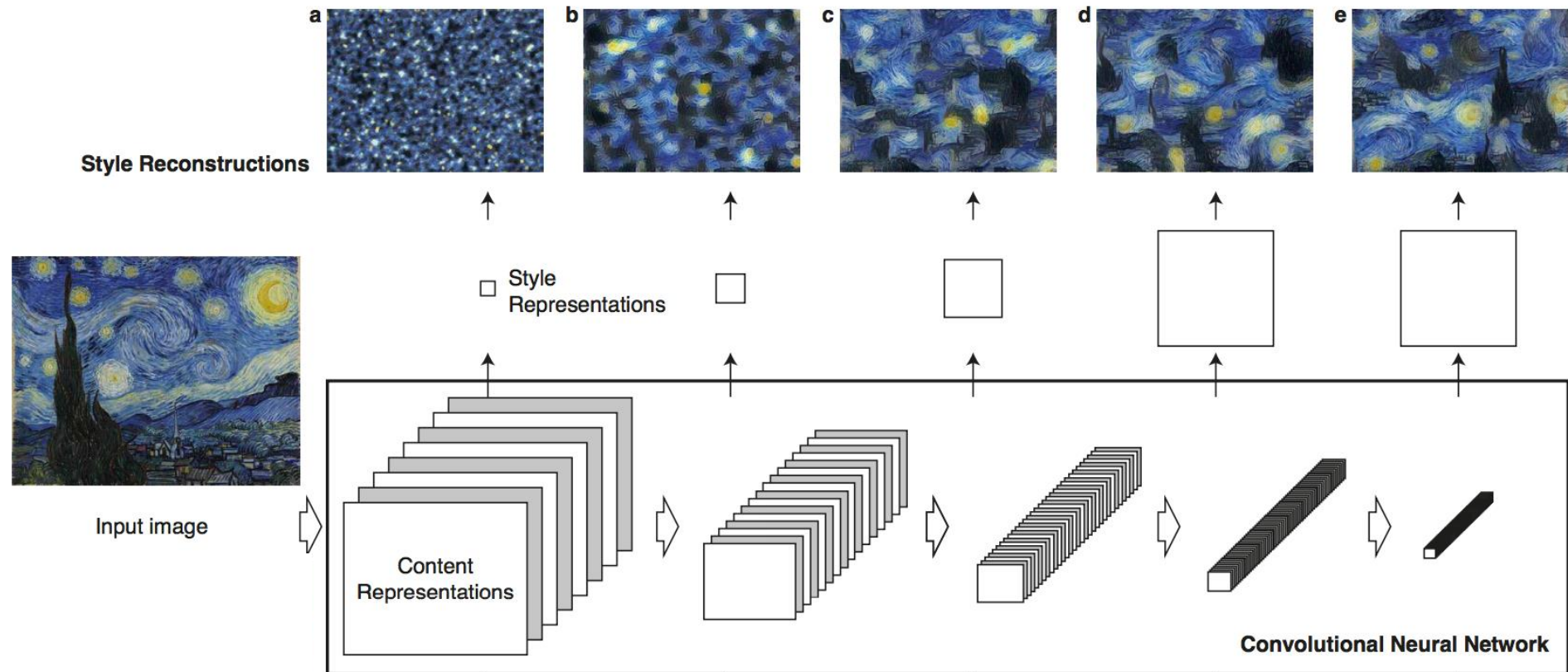
$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l$$

$$E_L = \sum (G^L - A^L)^2$$



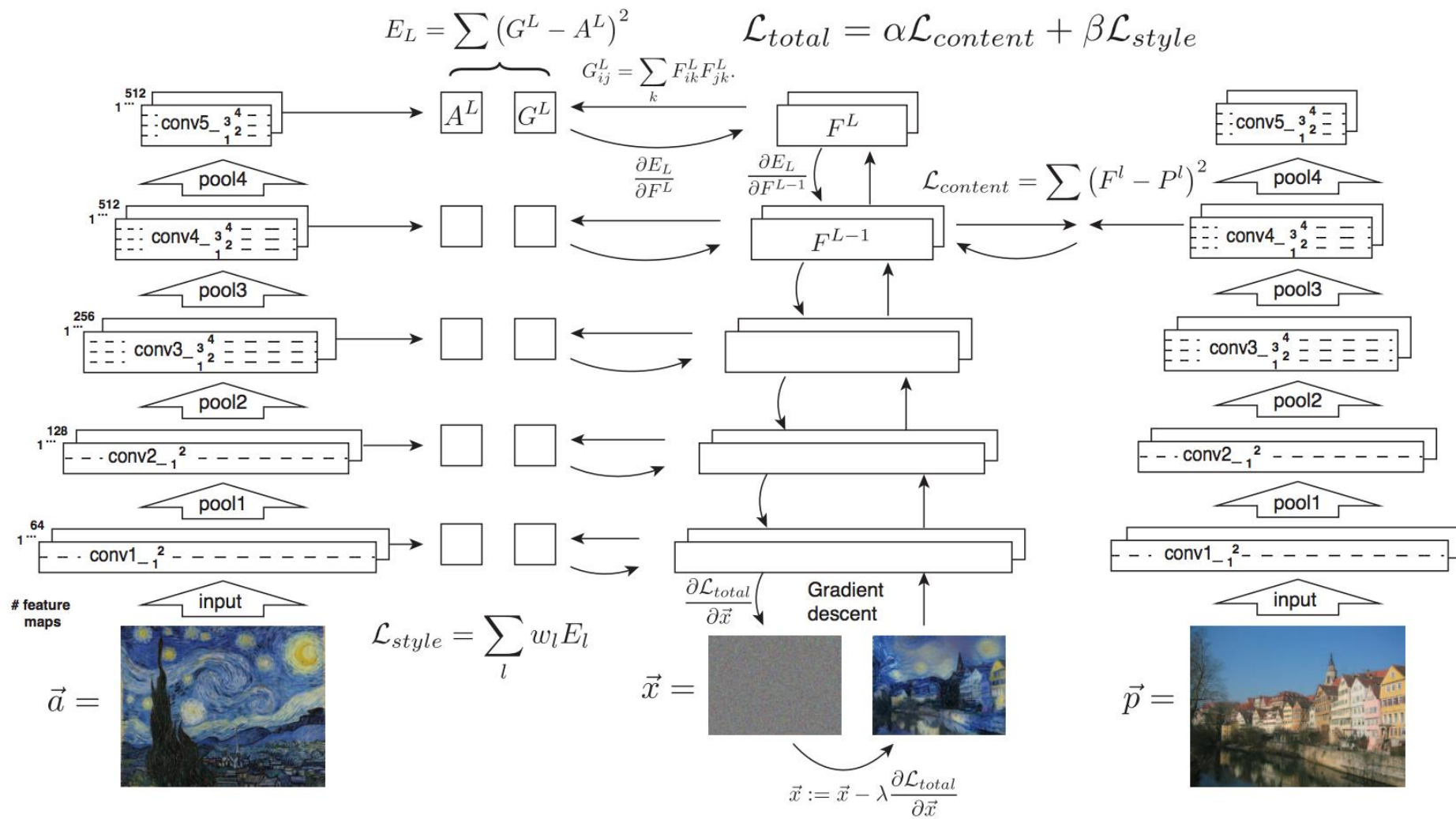
Idea 2:
Backpropagation of
Style





$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l.$$

$$E_L = \sum (G^L - A^L)^2$$





Questions