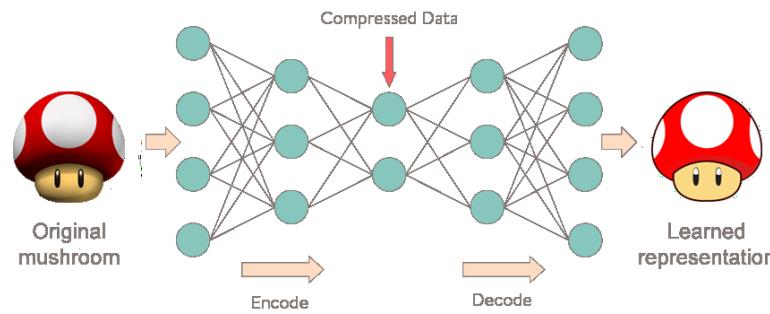
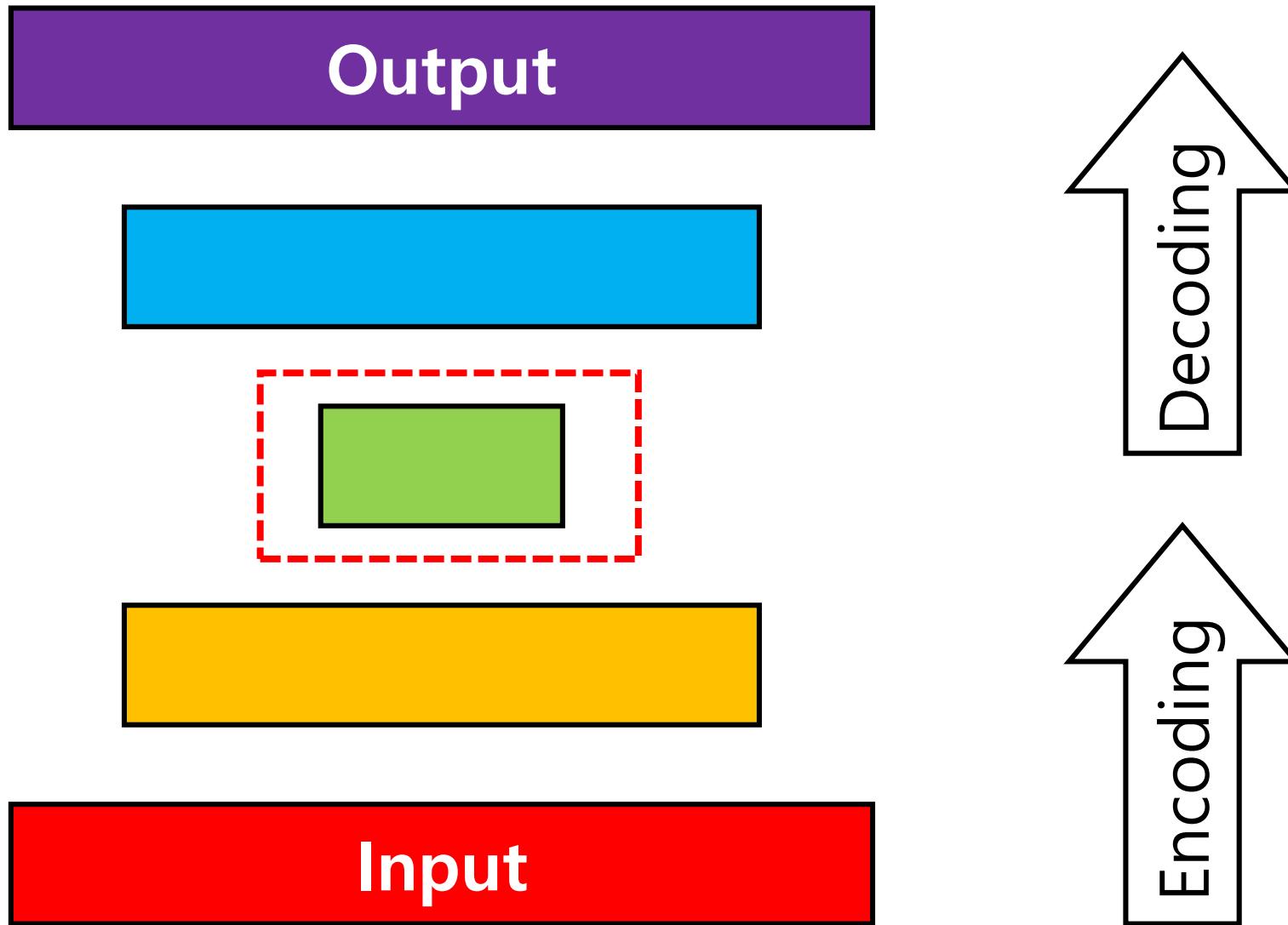


# AutoEncoder



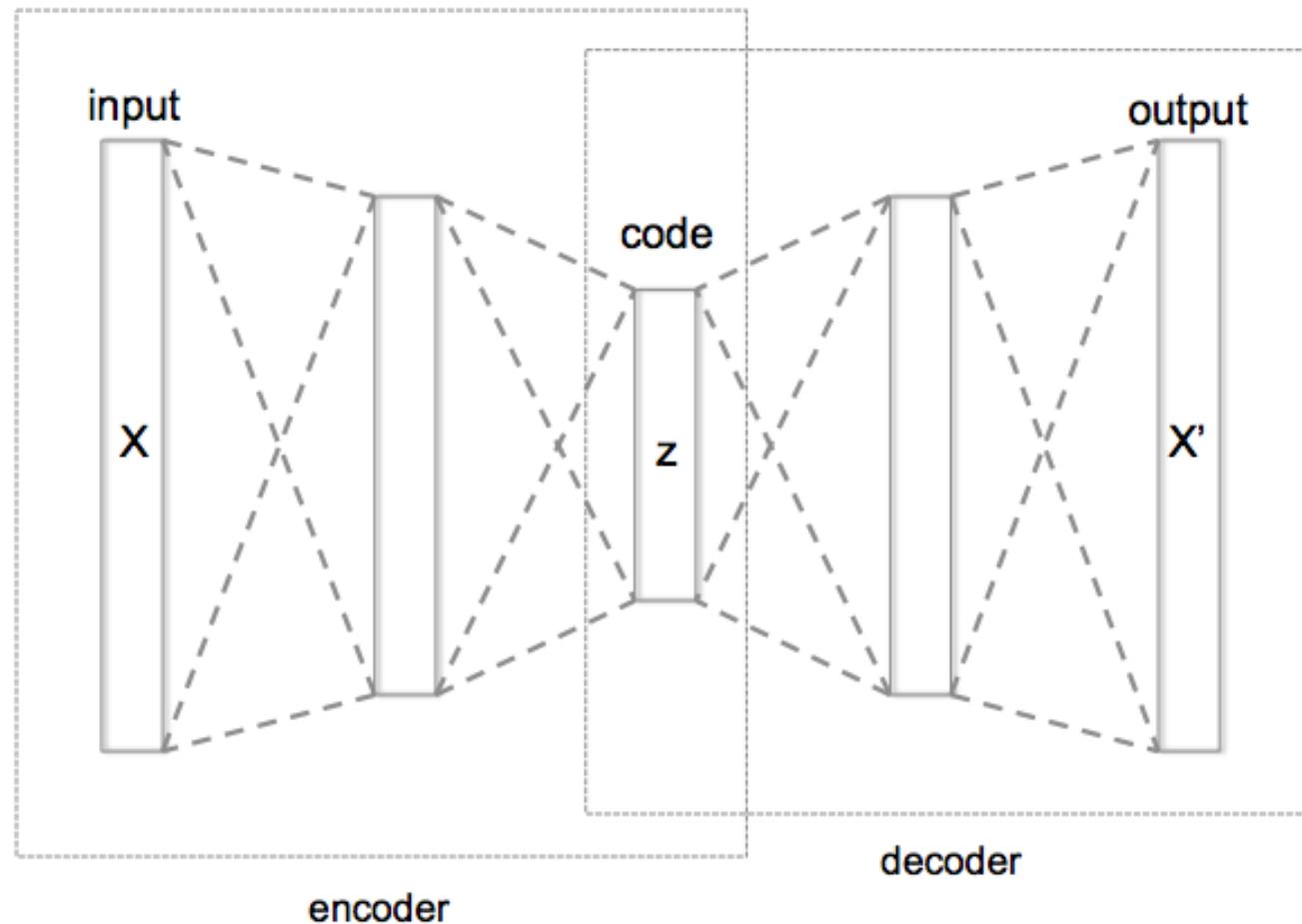
Fast Campus  
Start Deep Learning with TensorFlow

# Auto-Encoder



# Auto-Encoder

- Encoder + Decoder Structure



# Auto-Encoder

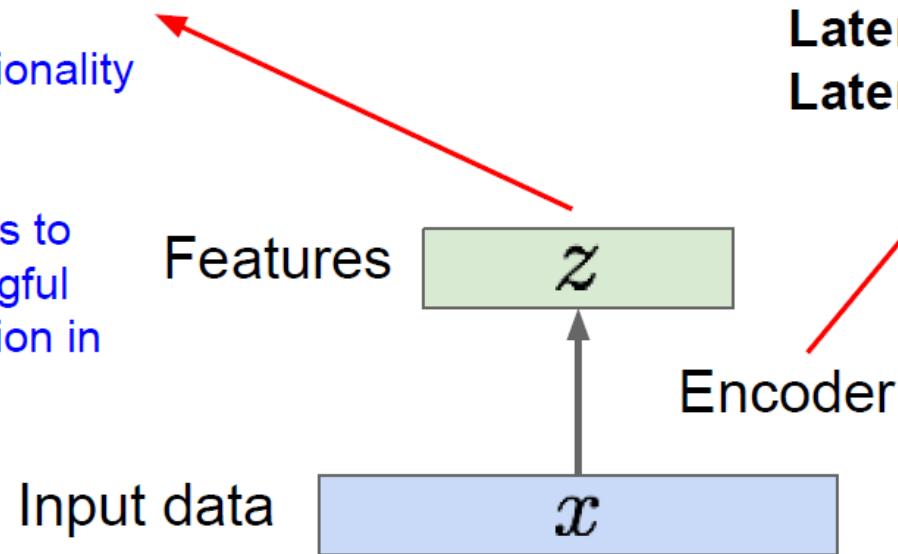
- Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

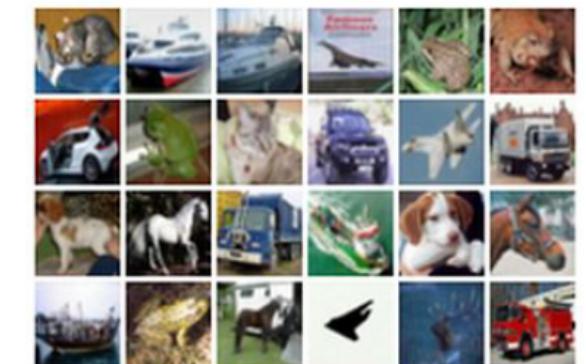
$z$  usually smaller than  $x$   
(dimensionality reduction)

Q: Why dimensionality reduction?

A: Want features to capture meaningful factors of variation in data

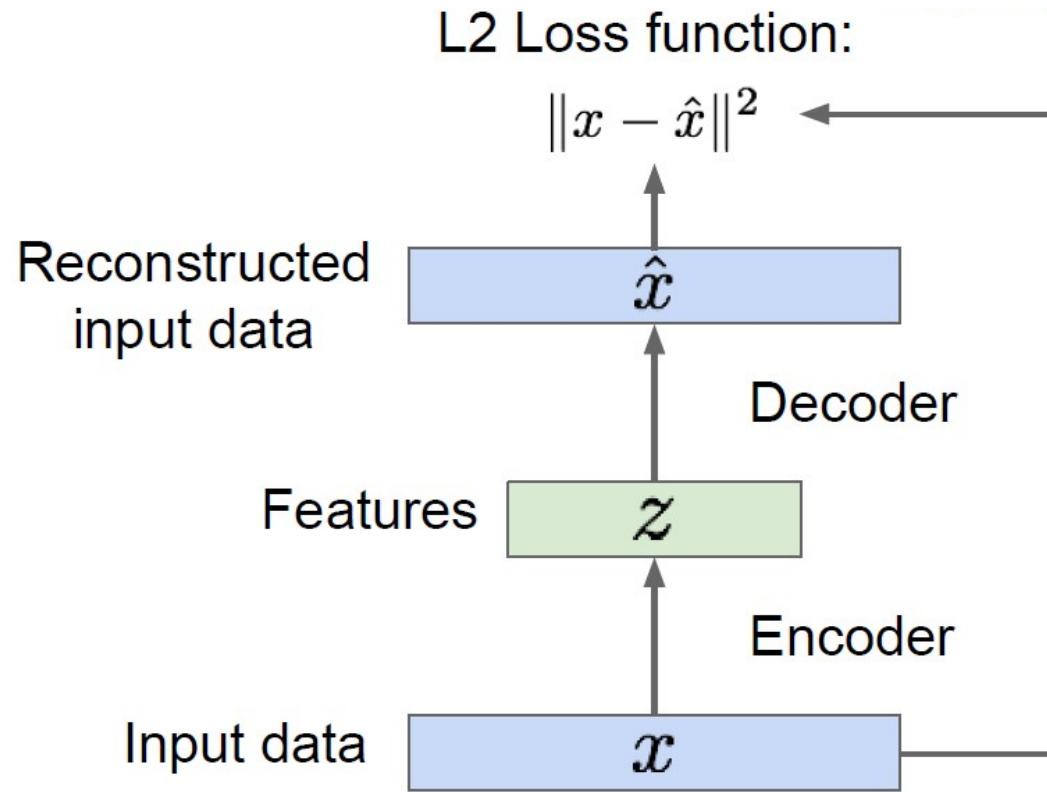


**Originally:** Linear +  
nonlinearity (sigmoid)  
**Later:** Deep, fully-connected  
**Later:** ReLU CNN



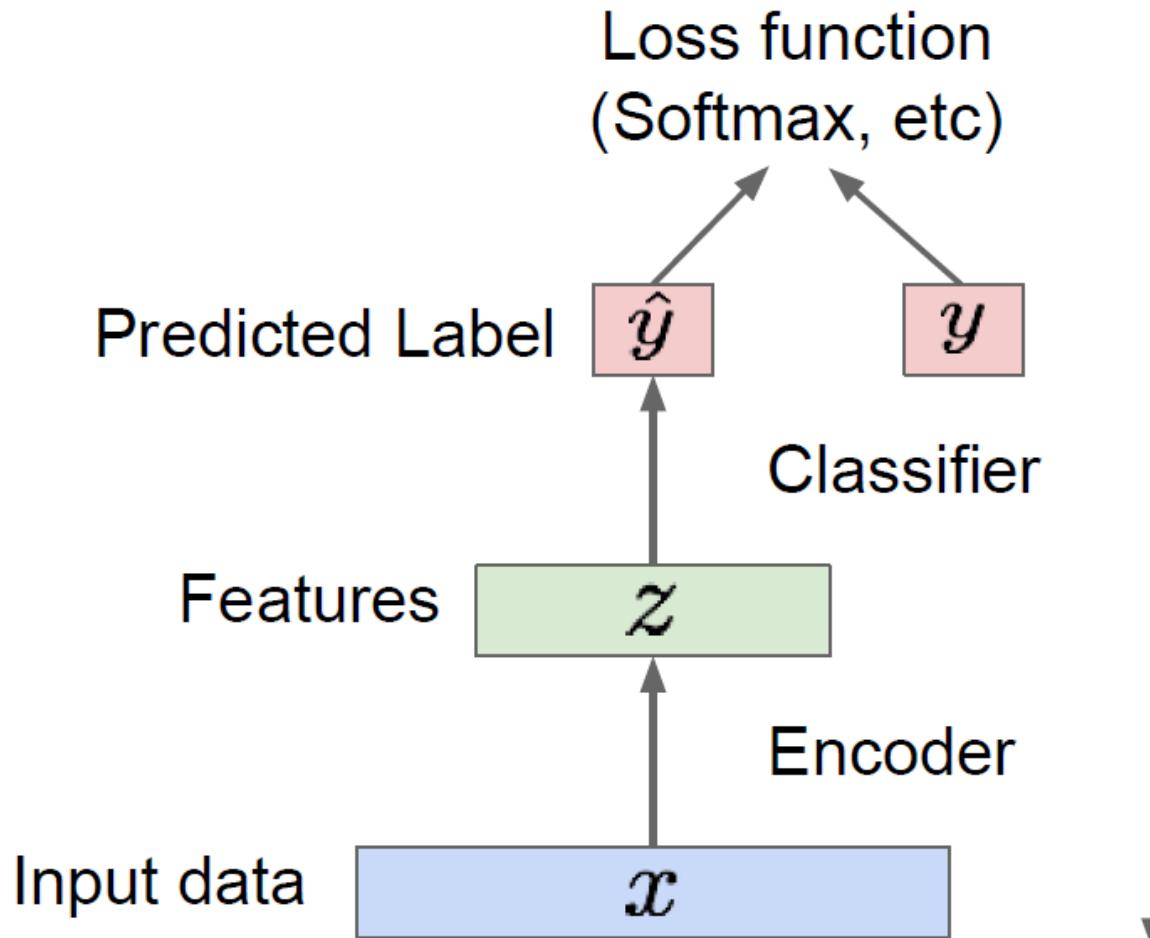
# Auto-Encoder

- Train such that features can be used to reconstruct original data  
“Autoencoding” - encoding itself



# Initialize a Supervised Model

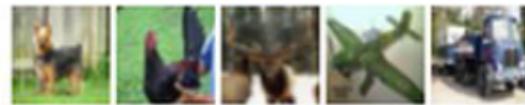
Encoder can be used to initialize a **supervised** model



Fine-tune encoder jointly with classifier

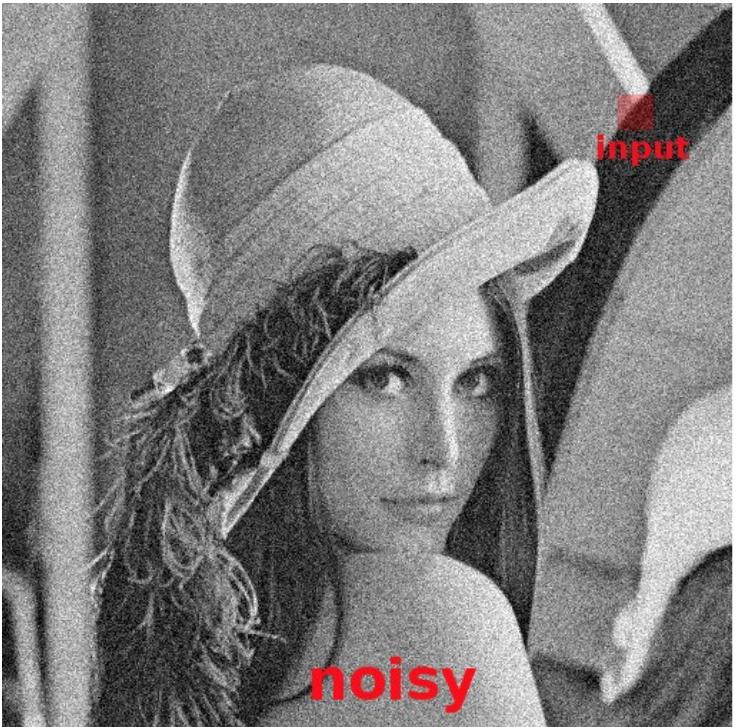
bird      plane  
dog      deer      truck

Train for final task (sometimes with small data)

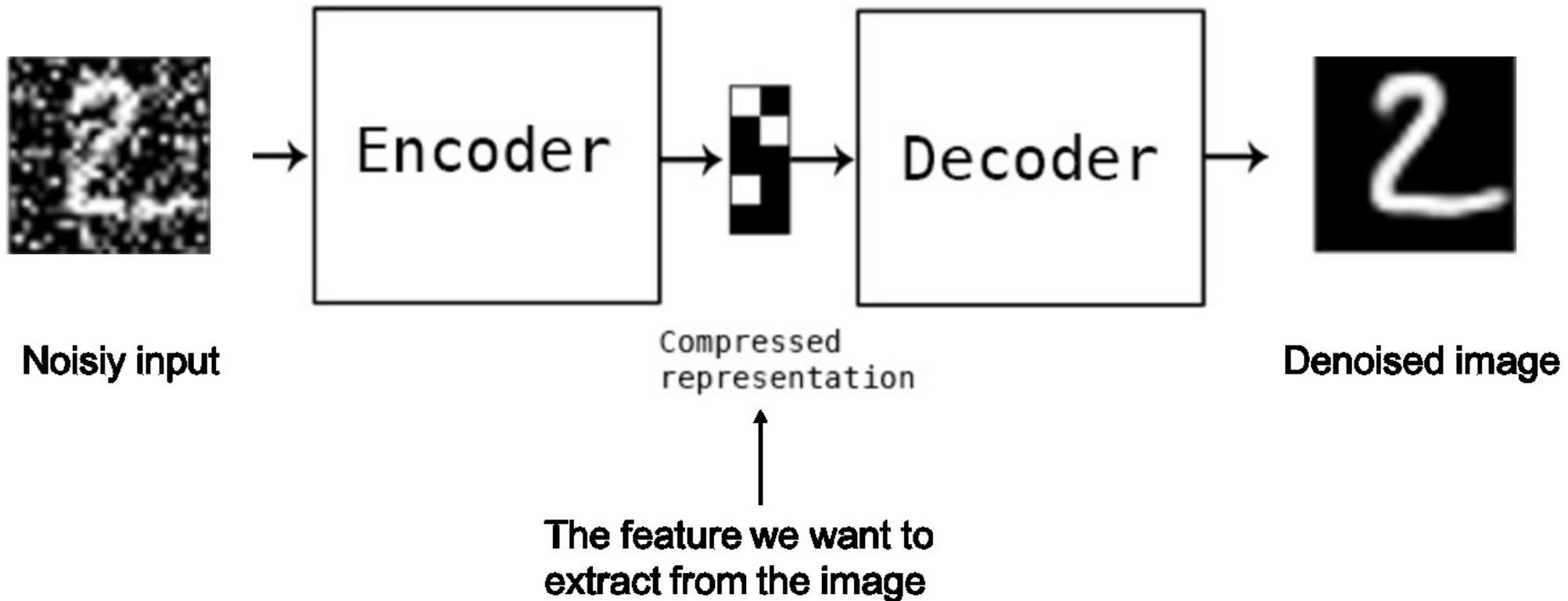


# Denoising Auto-Encoder

- Input and output data of the auto-encoder is basically **identical**
- Then, how can we make this network to have **denoising power**?



# Denoising Auto-Encoder

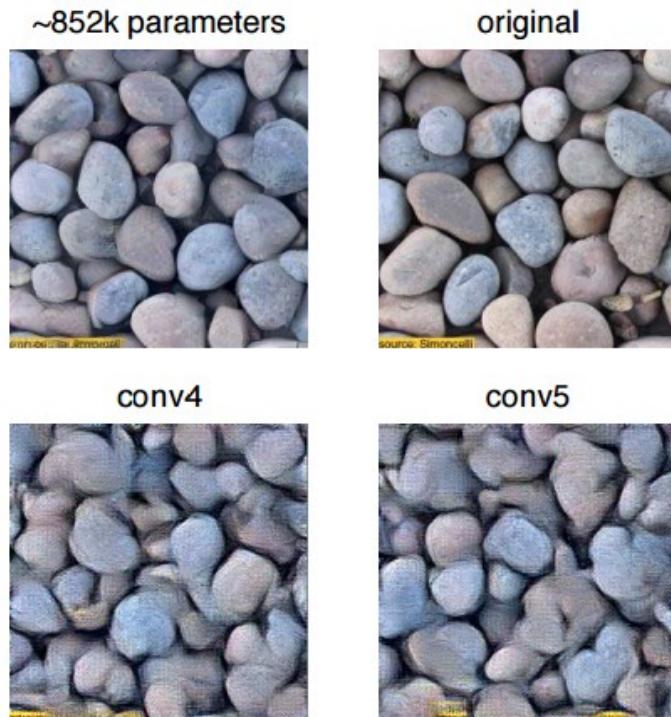


# Neural Style & Generative Adversarial Network

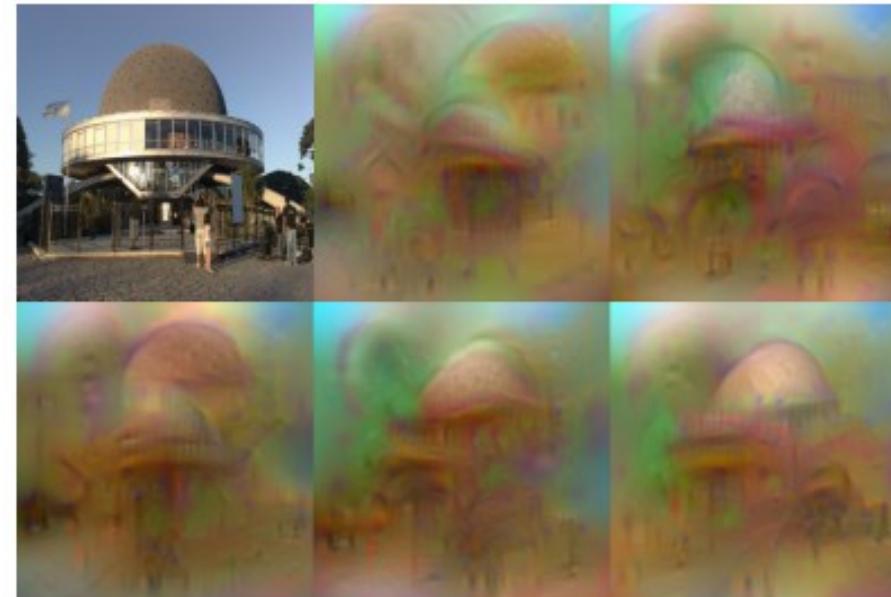
Golfzon DeepLearning Course

# Preliminaries

Texture Synthesis Using  
Convolutional Neural Networks  
**NIPS2015**



Understanding Deep Image Representations by  
Inverting Them  
**CVPR2015**



# A Neural Algorithm of Artistic Style



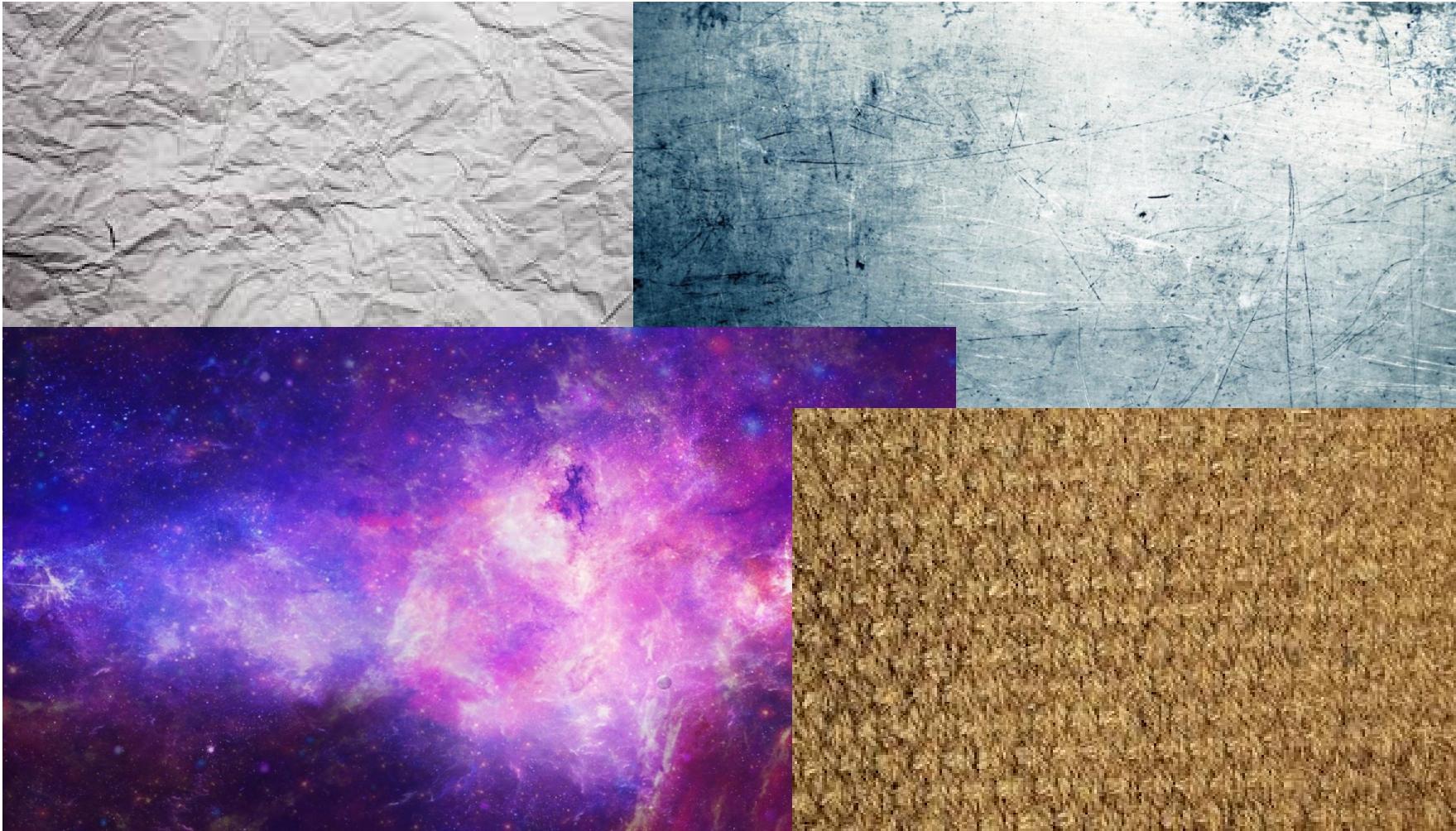
# **Texture Synthesis Using Convolutional Neural Networks**

## **-NIPS2015**

# A Neural Algorithm of Artistic Style



# Texture

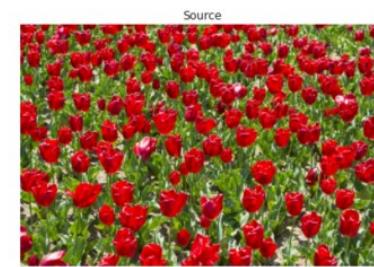
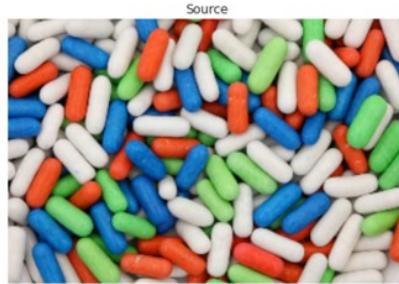
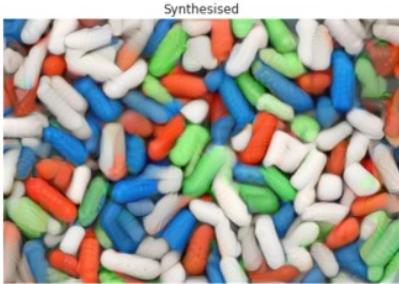


# Visual texture synthesis

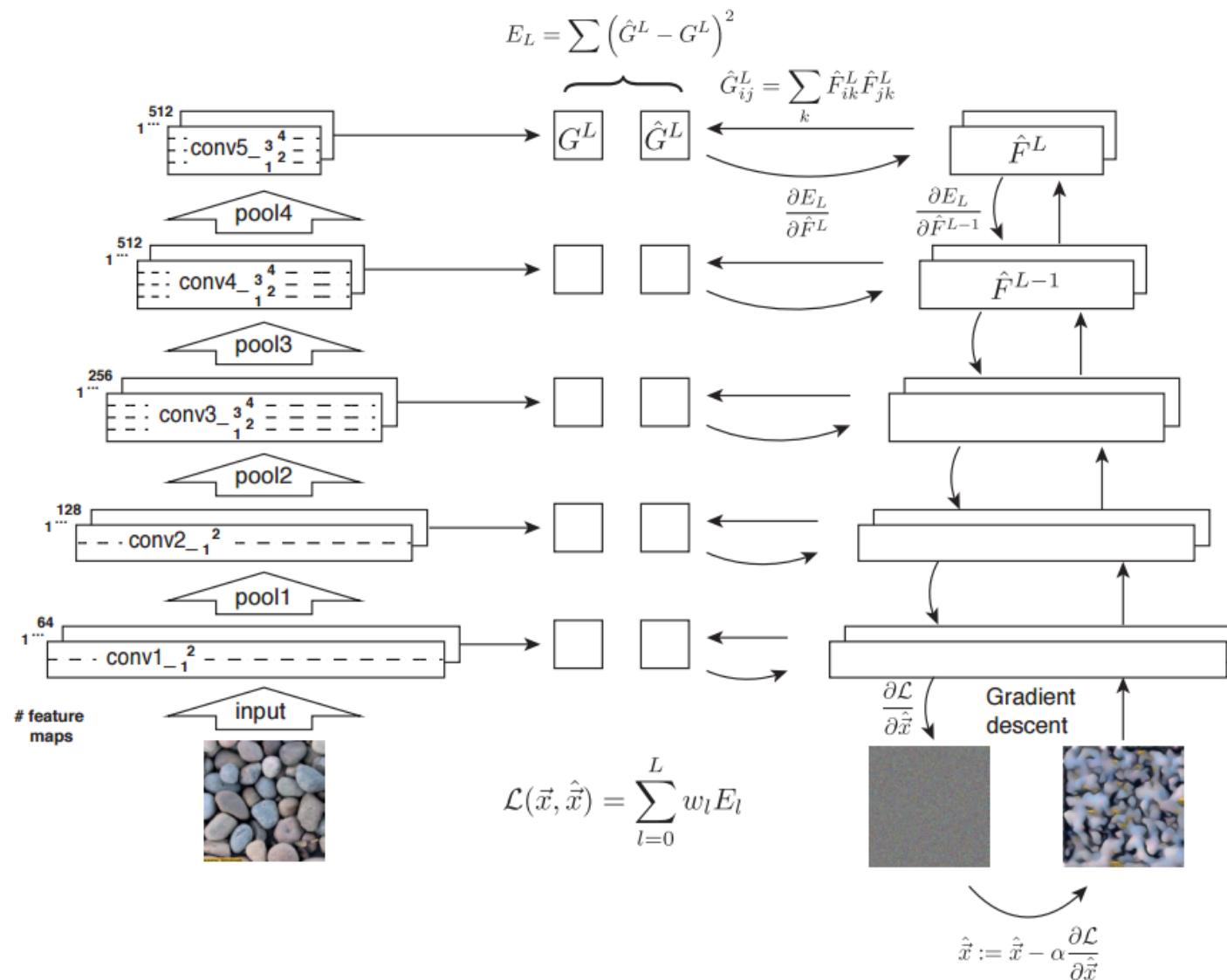


- Which one do you think is real?
- Right one is real.
- Goal of **texture synthesis** is to produce (arbitrarily many) new samples from an example texture.

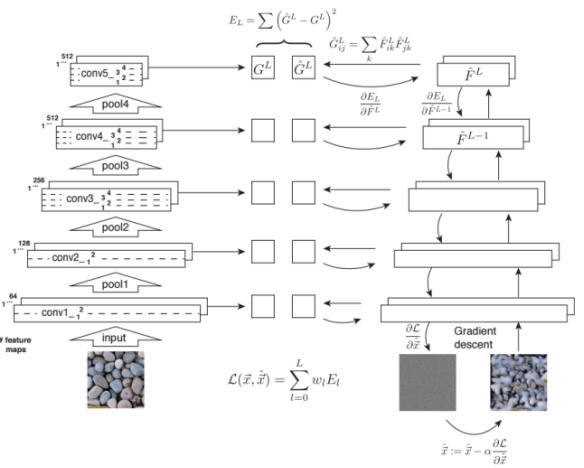
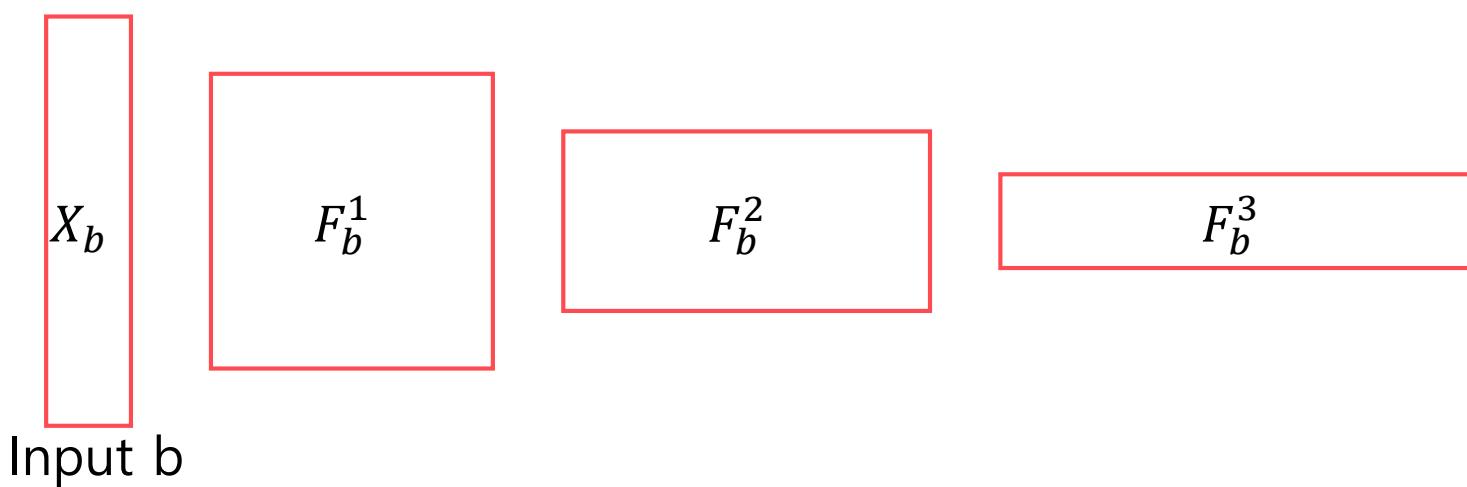
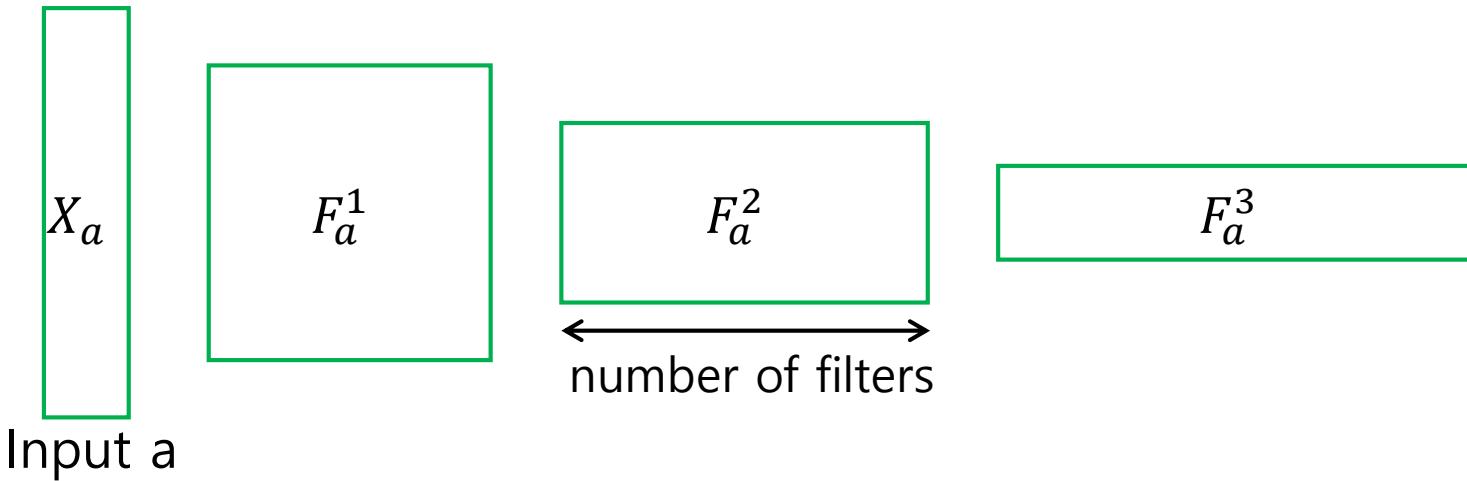
# Results of this work



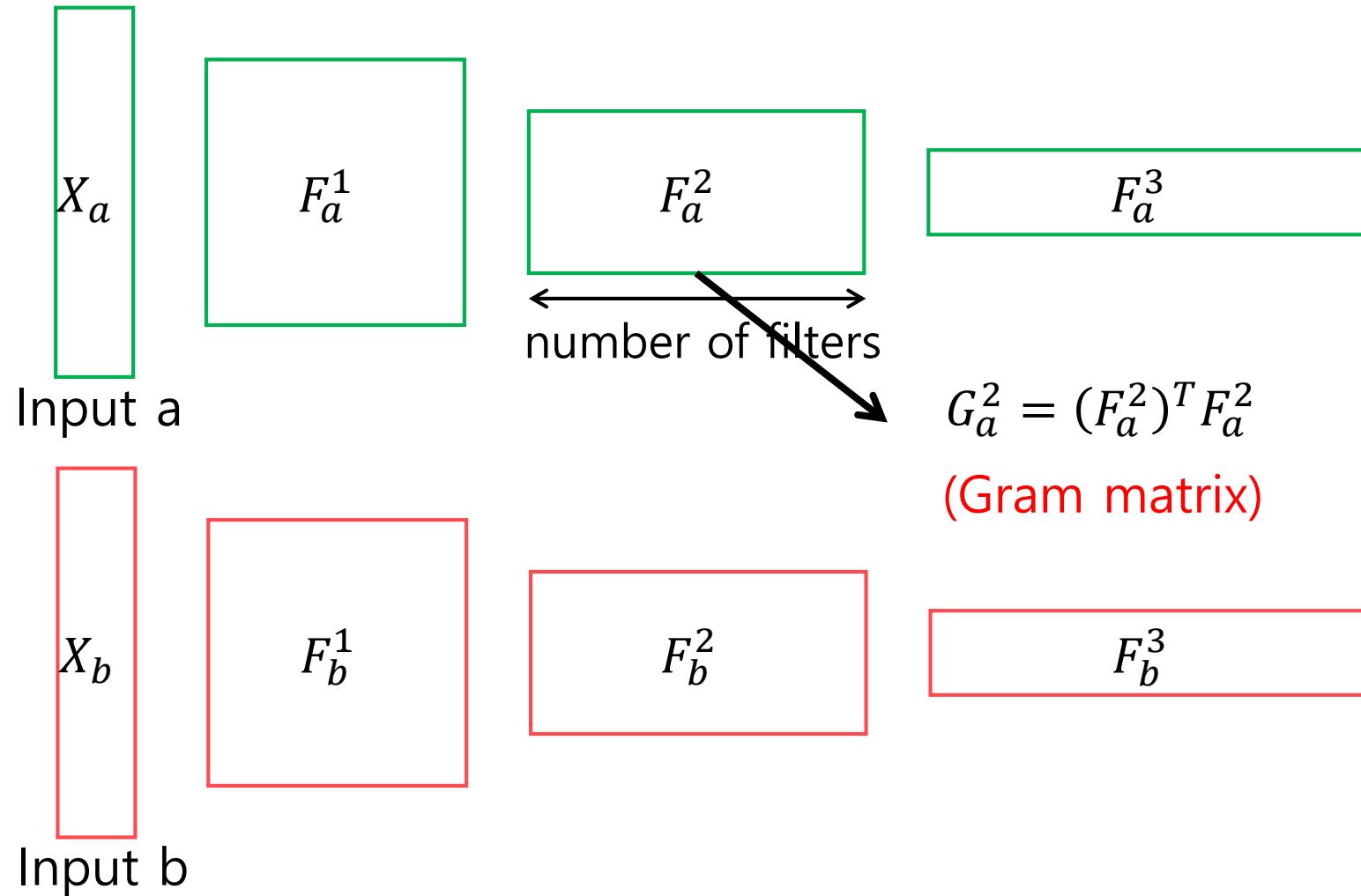
# How?



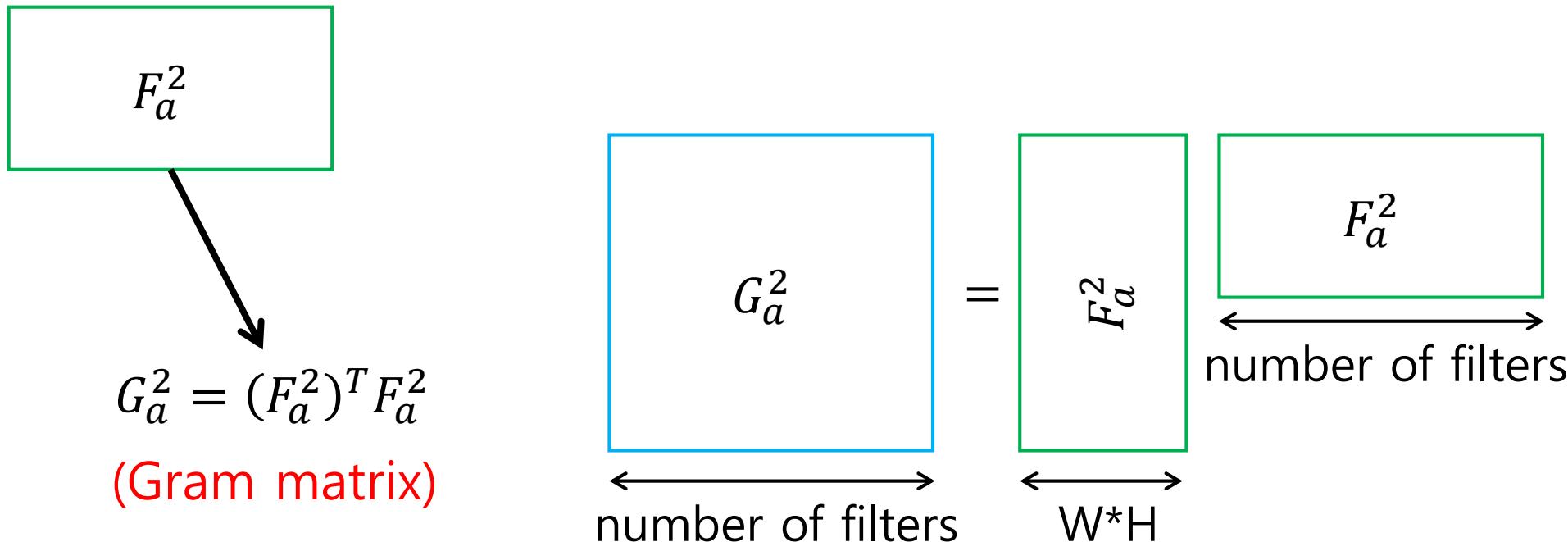
# Texture Model



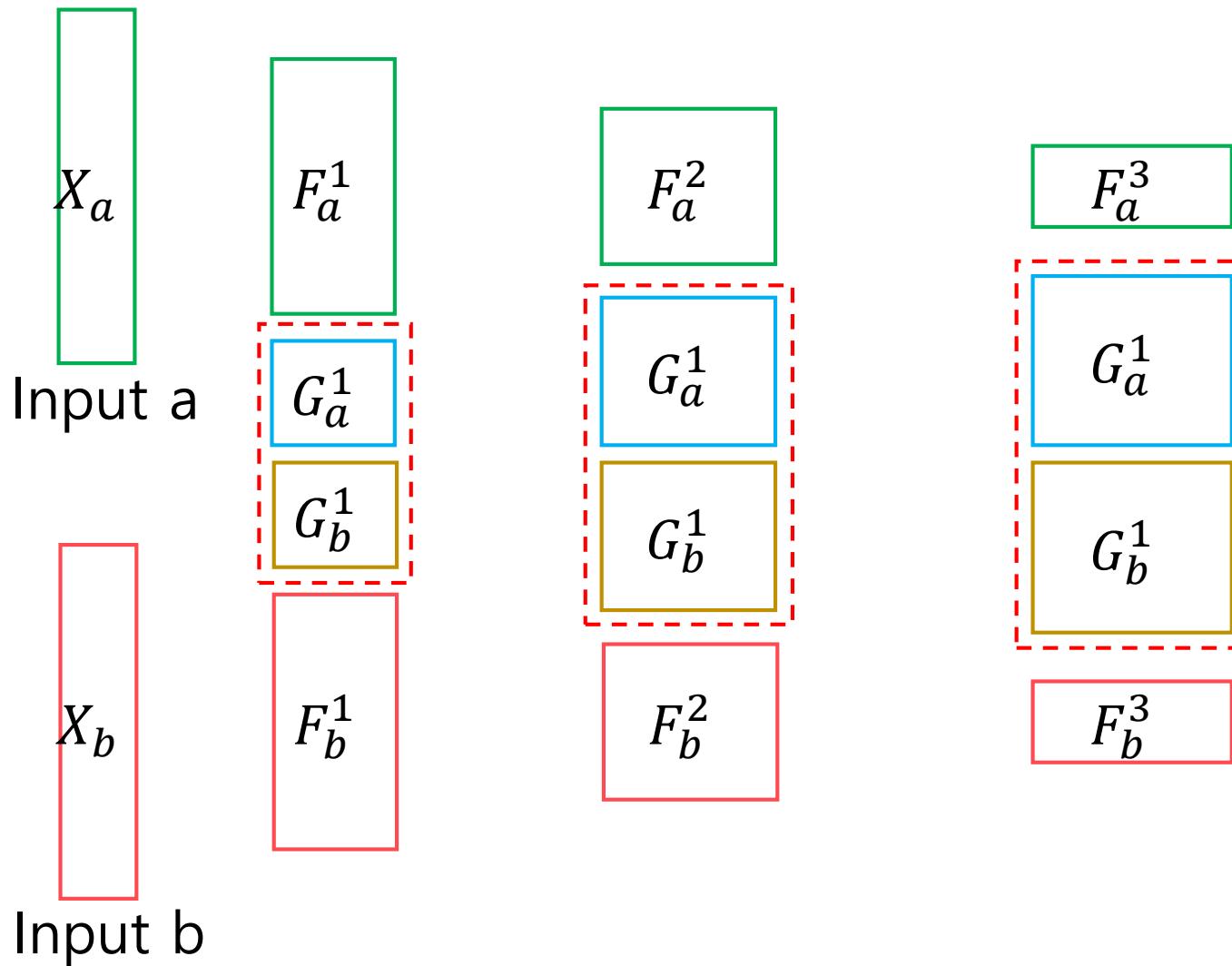
# Feature Correlations



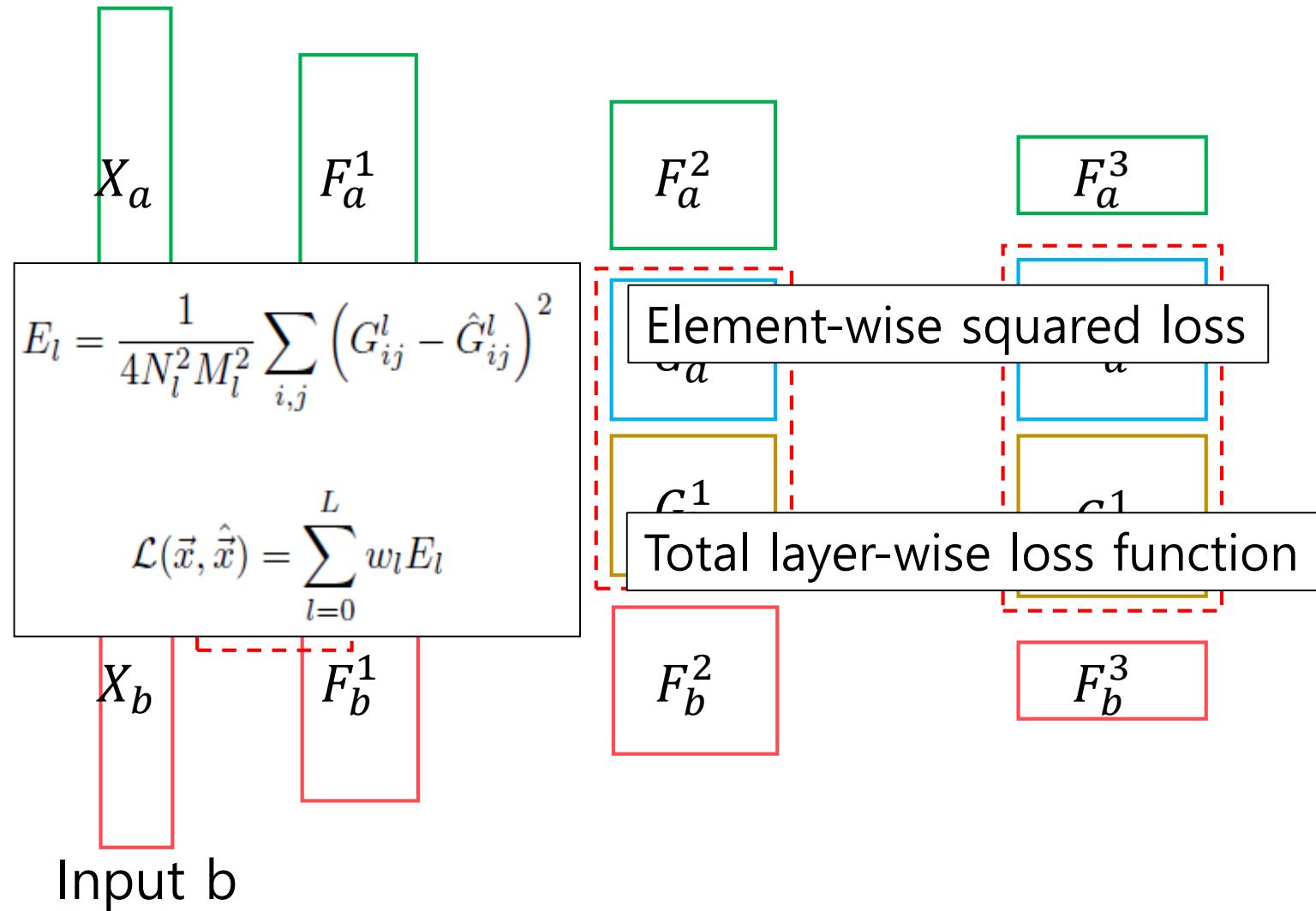
# Feature Correlations



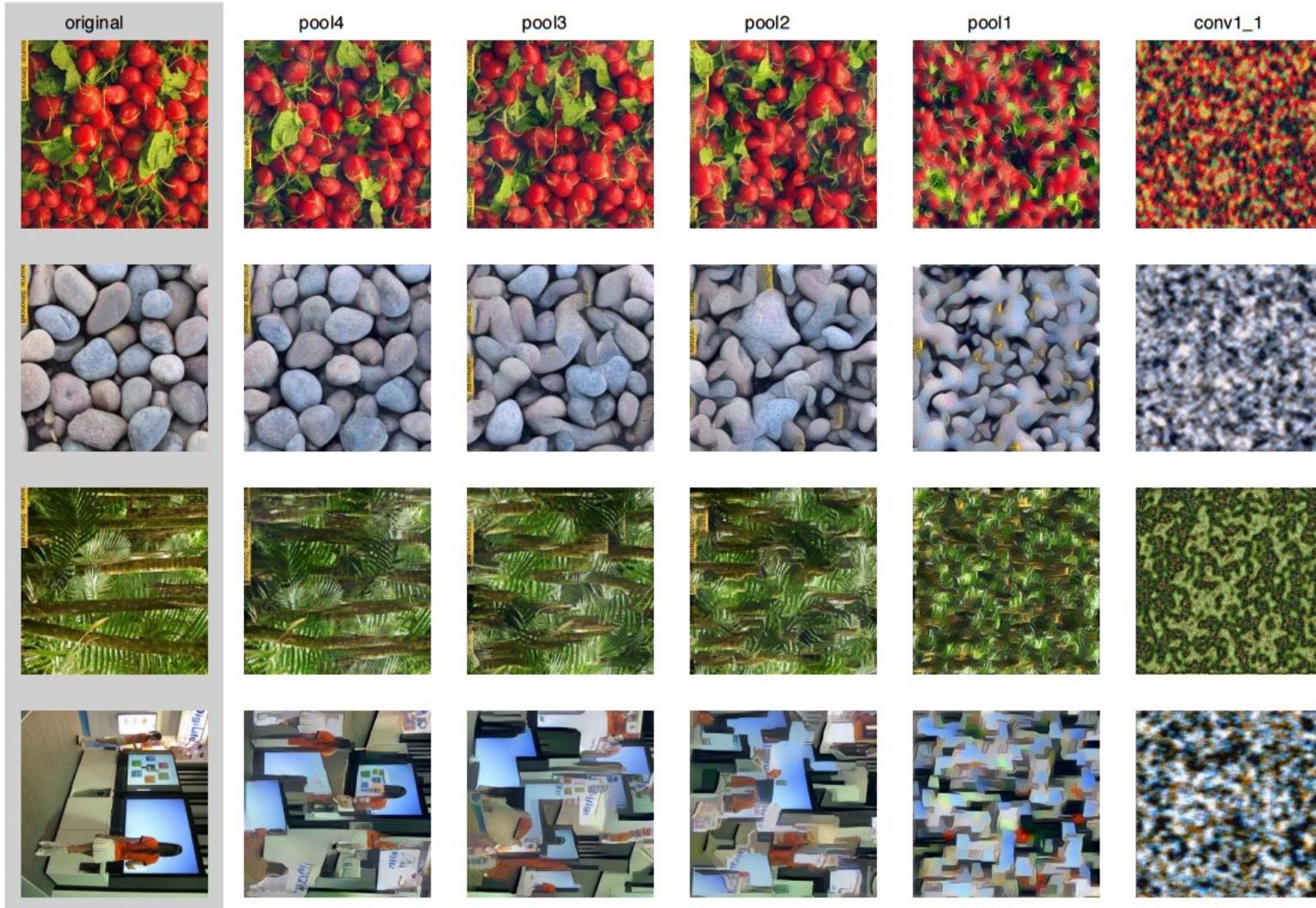
# Texture Generation



# Texture Generation



# Results



# Results

**A** ~1k parameters



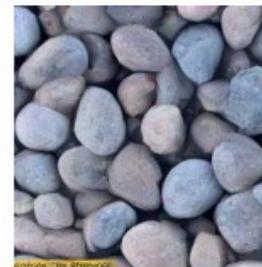
~10k parameters



~177k parameters



~852k parameters



original



**B** conv1



conv2



conv3



conv4



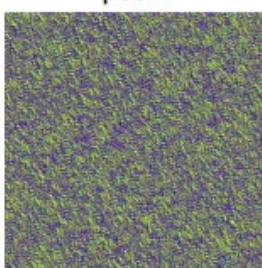
conv5



**C** conv1\_1



pool1



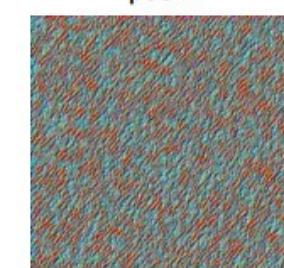
pool2



pool3



pool4



# **Understanding Deep Image Representations by Inverting Them**

## **-CVPR2015**

# Reconstruction from feature map

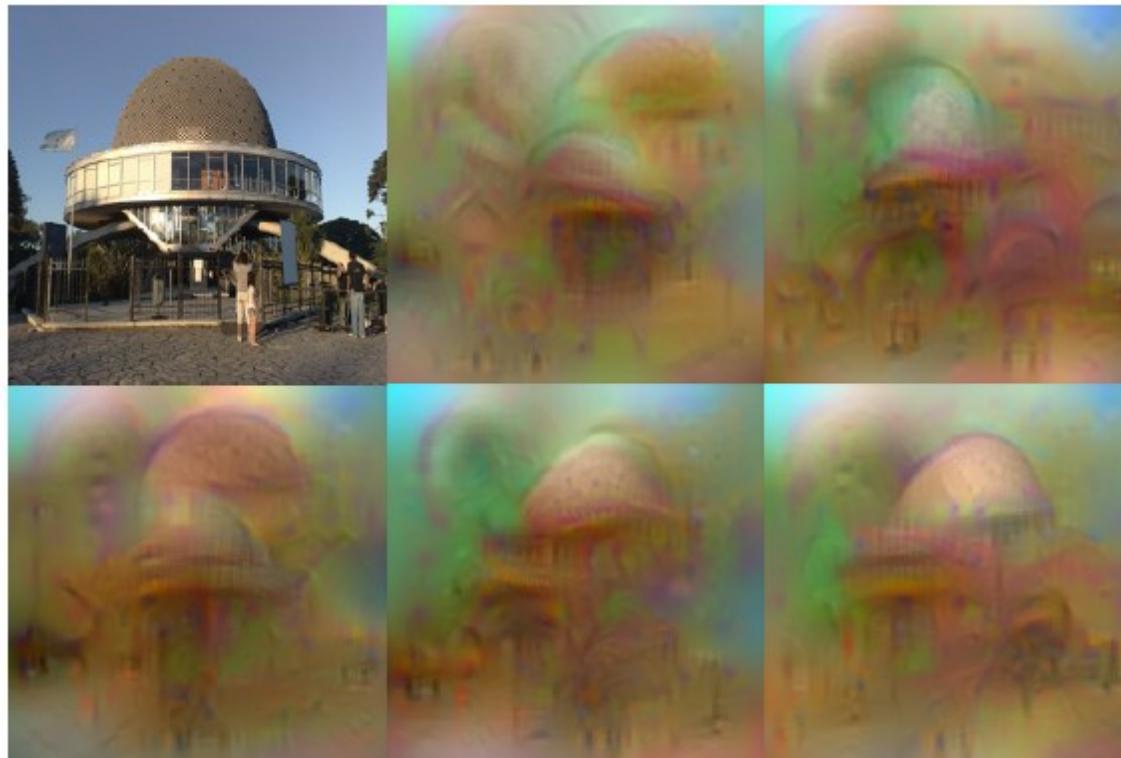
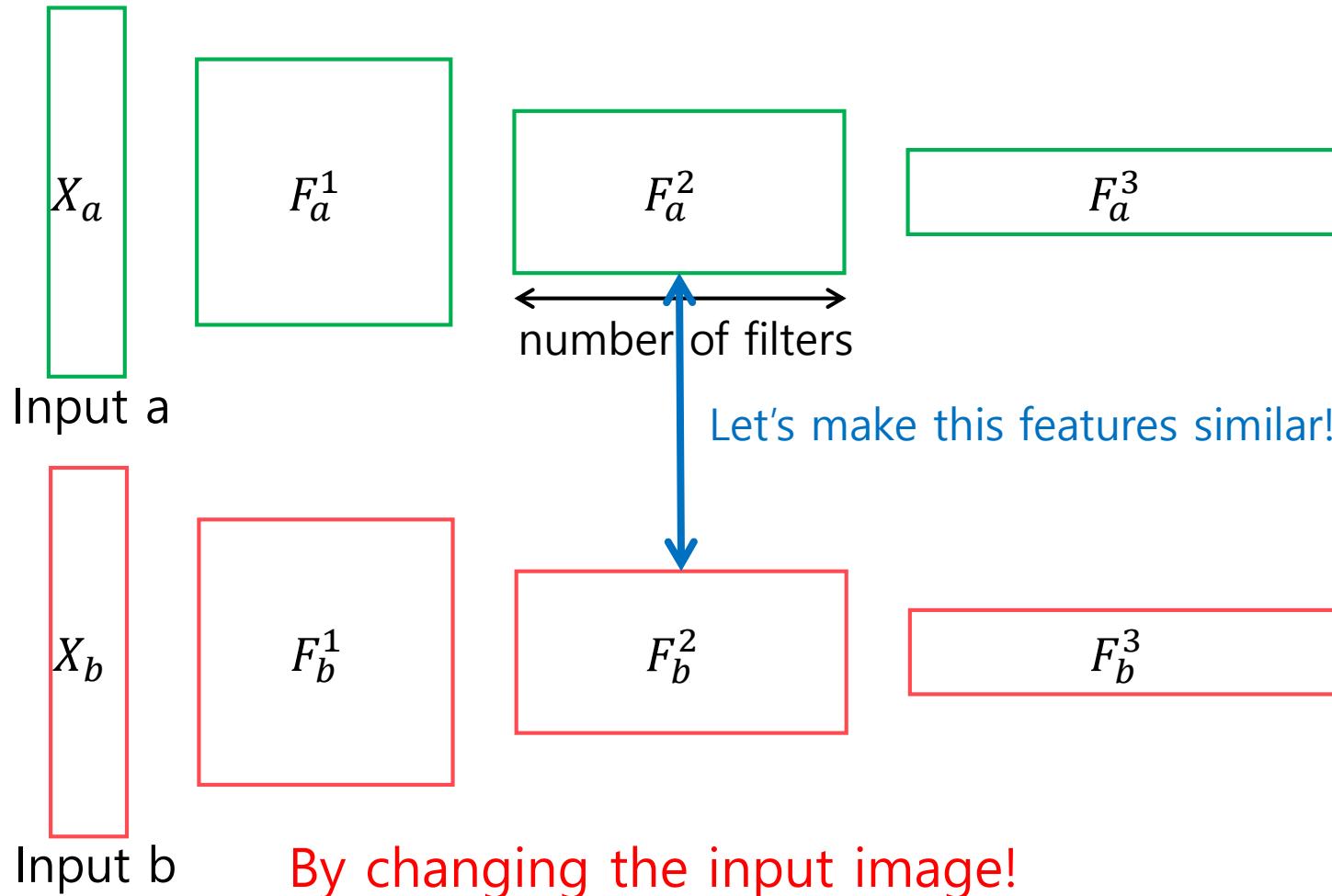


Figure 1. **What is encoded by a CNN?** The figure shows five possible reconstructions of the reference image obtained from the 1,000-dimensional code extracted at the penultimate layer of a reference CNN[15] (before the softmax is applied) trained on the ImageNet data. From the viewpoint of the model, all these images are practically equivalent. This image is best viewed in color/screen.

# Reconstruction from feature map



# Receptive Field

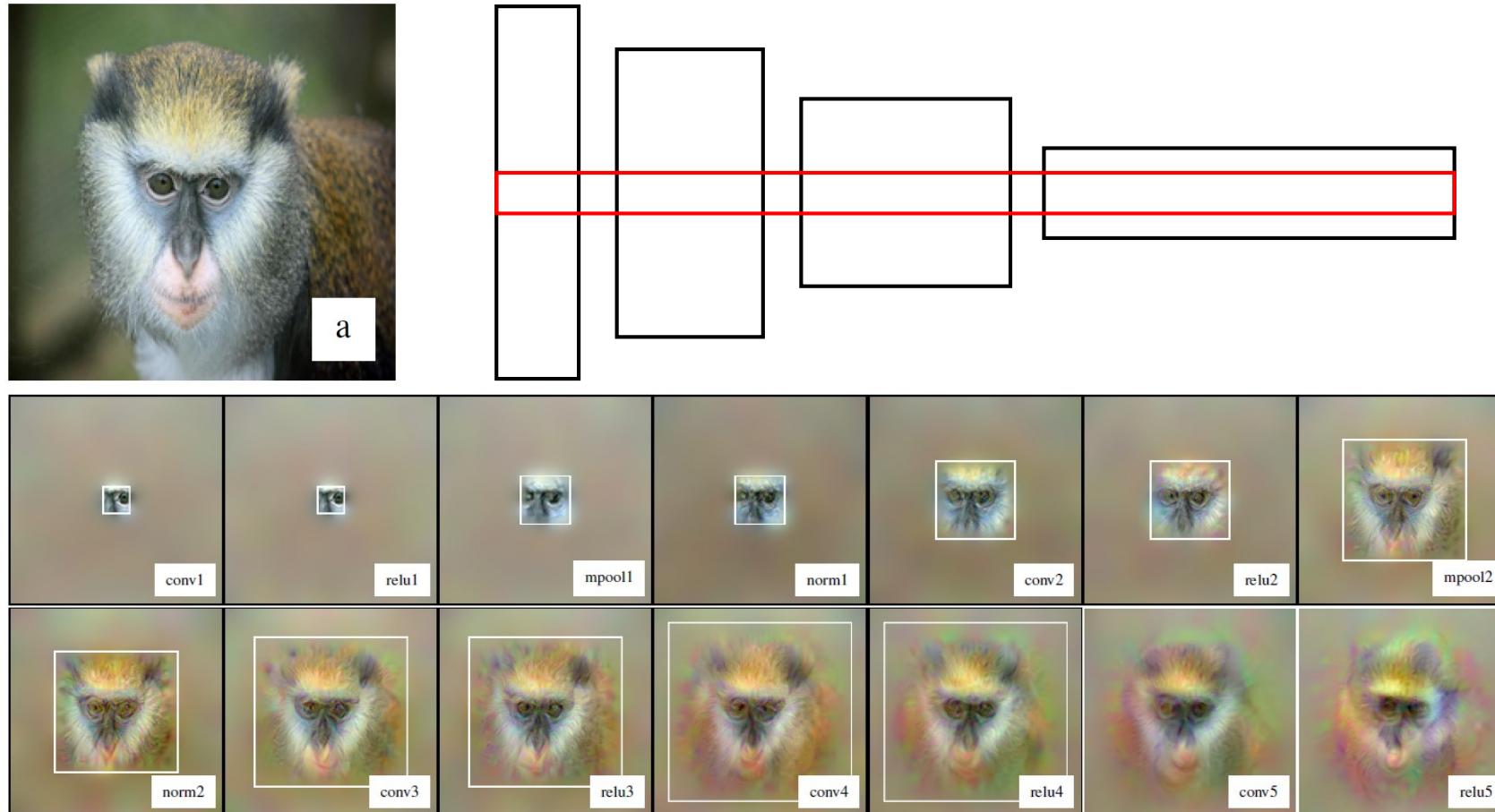
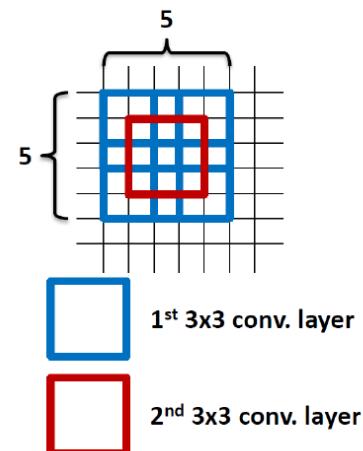


Figure 9. **CNN receptive field.** Reconstructions of the image of Fig. 5.a from the central  $5 \times 5$  neuron fields at different depths of CNN-A. The white box marks the field of view of the  $5 \times 5$  neuron field. The field of view is the entire image for conv5 and relu5.

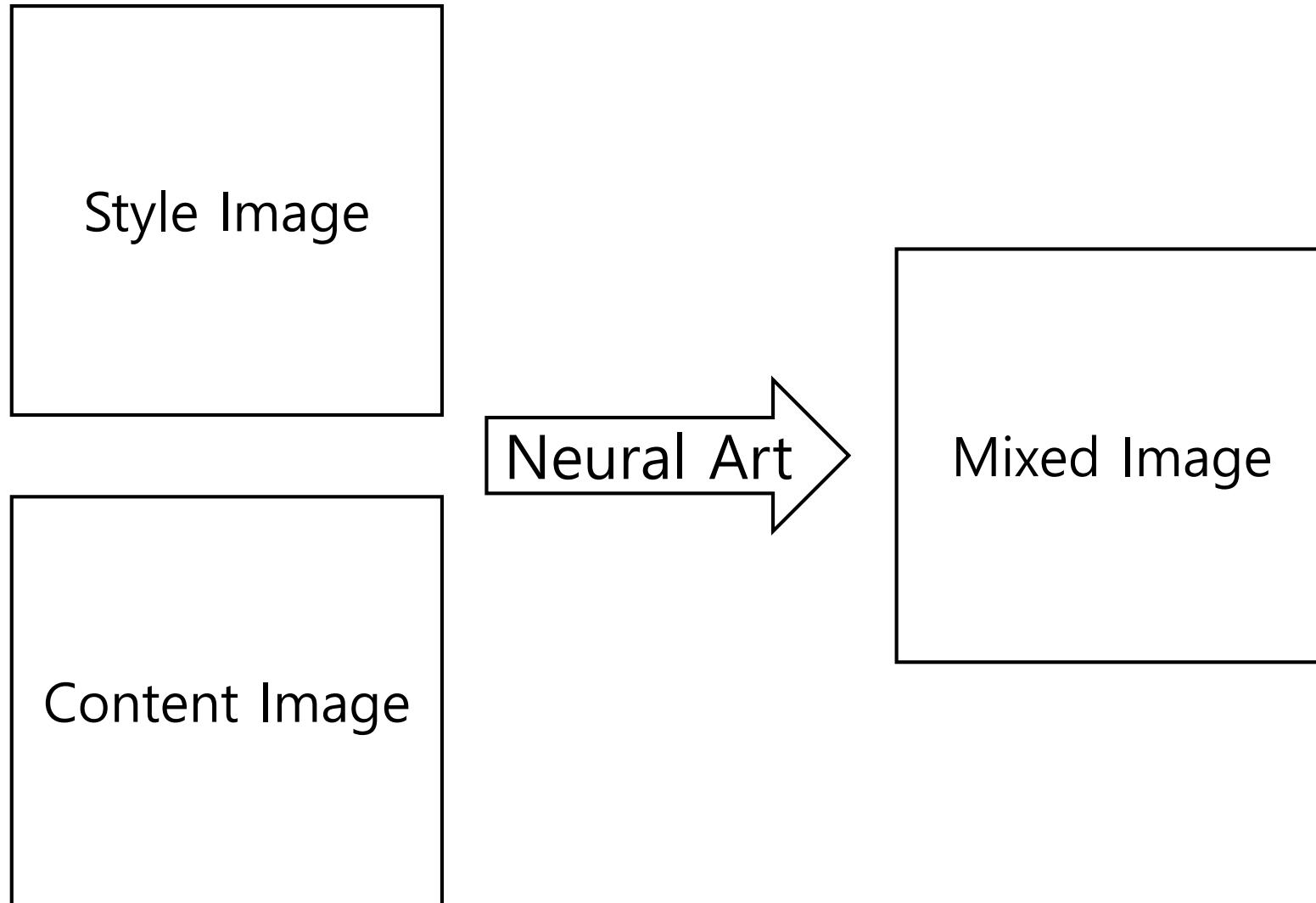
# VGG

- 3x3 filter만 반복해서 사용
- Why??
  - Convolution filter를 stack하면 더 큰 receptive field를 가질 수 있음
    - 2개의 3x3 filter = 5x5 filter
    - 3개의 3x3 filter = 7x7 filter
  - Parameter수는 큰 filter 사용하는 경우에 비하여 감소 → regularization 효과

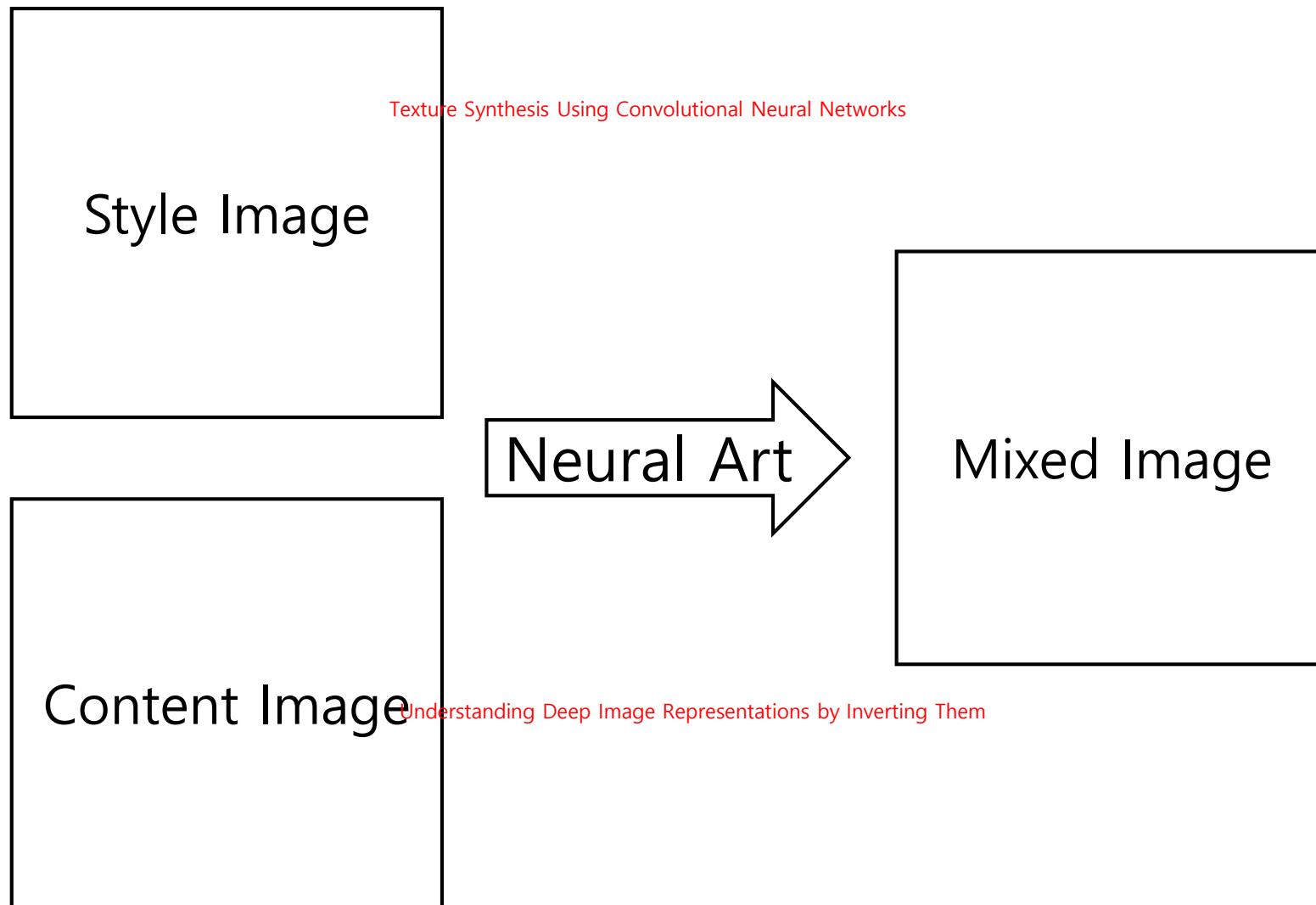


# **A Neural Algorithm of Artistic Style**

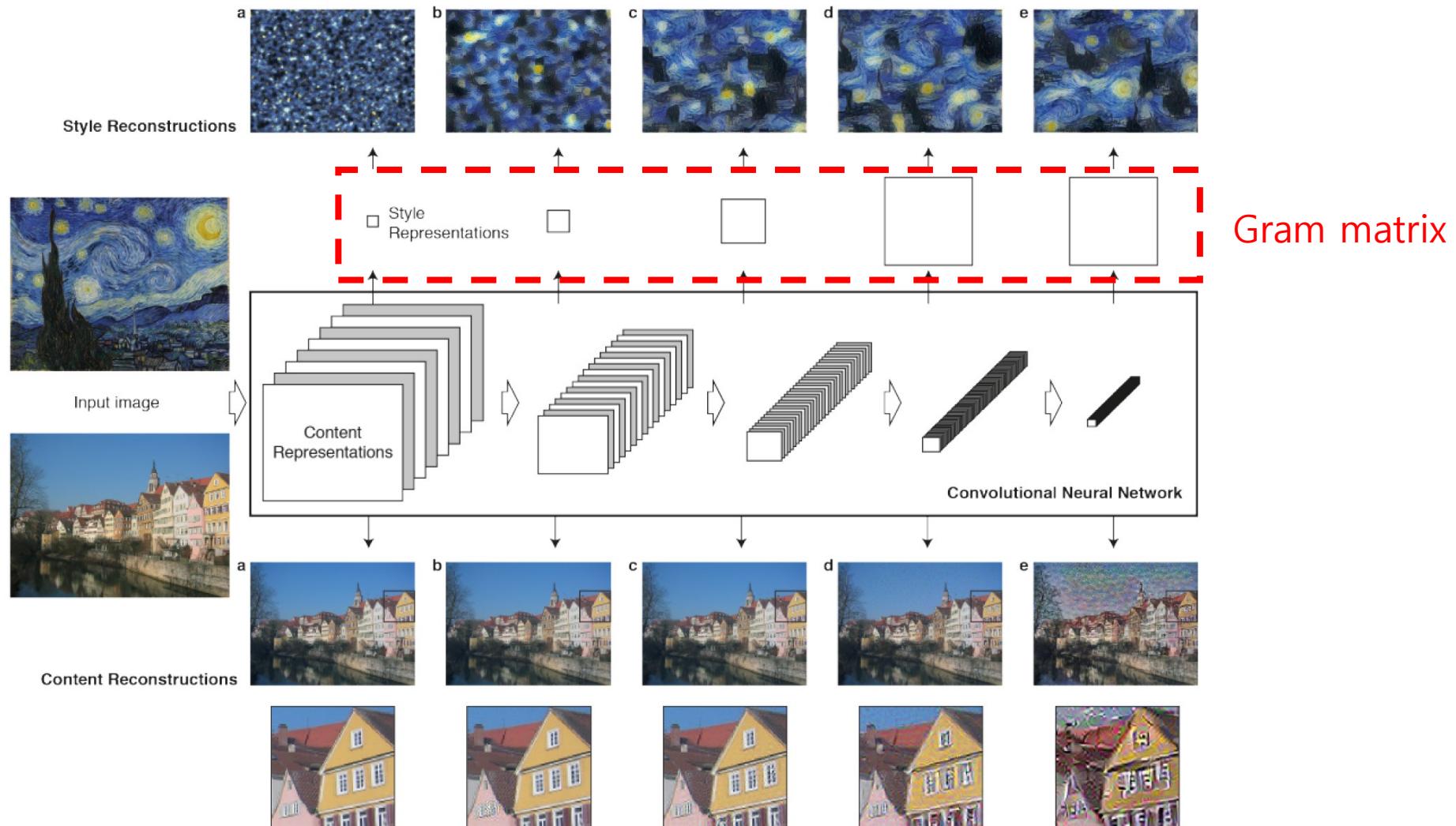
# How?



# How?



# How?



# Neural Art

$\vec{p}$ : original photo,  $\vec{a}$ : original artwork  
 $\vec{x}$ : image to be generated

## Content

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

$$\frac{\partial \mathcal{L}_{content}}{\partial F_{ij}^l} = \begin{cases} (F^l - P^l)_{ij} & \text{if } F_{ij}^l > 0 \\ 0 & \text{if } F_{ij}^l < 0 . \end{cases}$$

## Style

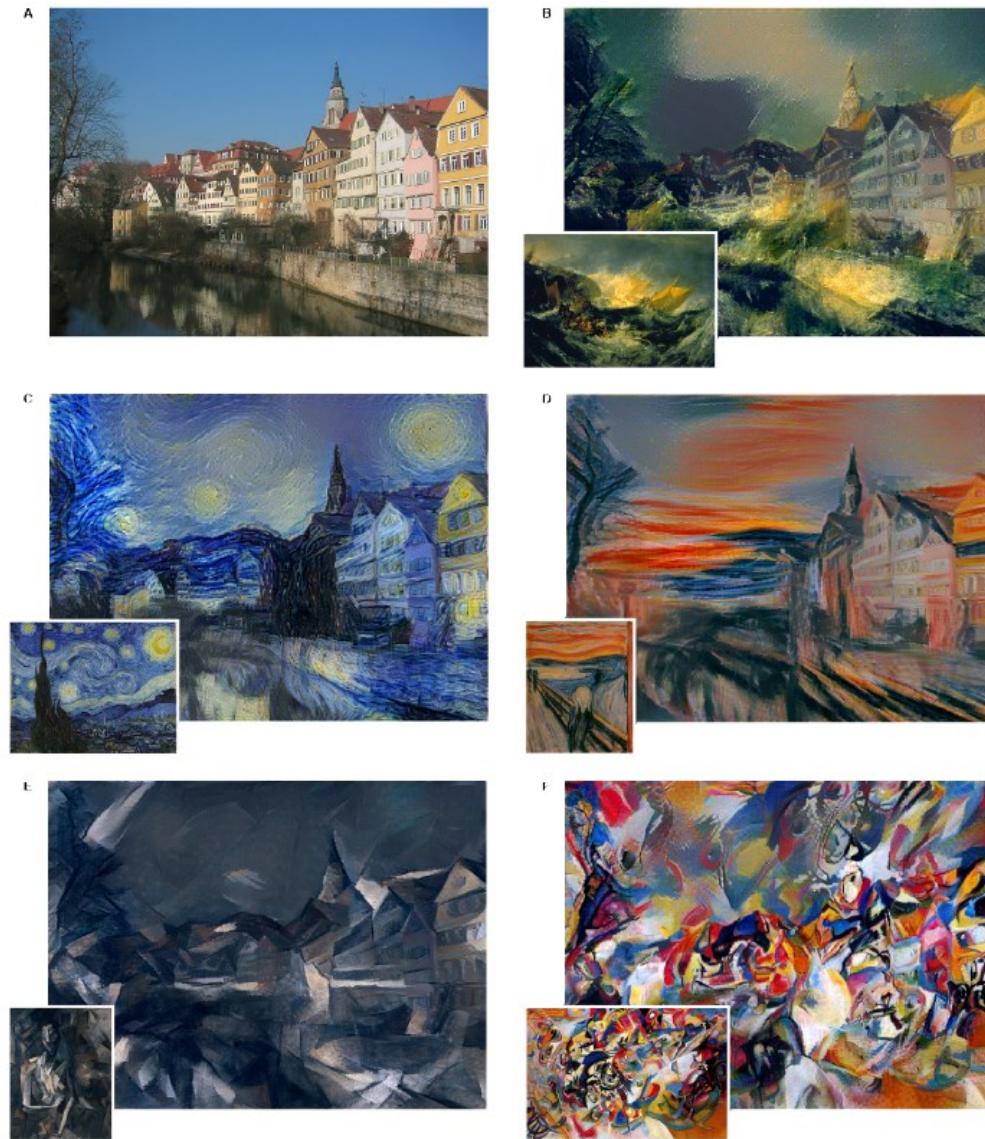
$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_{ij}^l - A_{ij}^l)^2 \quad G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l.$$

$$\mathcal{L}_{style}(\vec{a}, \vec{x}) = \sum_{l=0}^L w_l E_l$$

Total loss = content loss + style loss

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

# Results



# Results

