Deep Learning Bootcamp with PyTorch

Noel Shin

Contents

- **Introduction**
- **Basics**
- **Appendix**

Introduction

Goal of lecture

Getting used to

- **read a PyTorch code of interest,**
- **make a deep learning model using PyTorch.**

What is PyTorch?

PyTorch is an **open-source machine learning library** for Python, based on Torch.

- **Tensor computation with strong GPU acceleration***

- Deep neural networks built on a tape-based **autograd system**

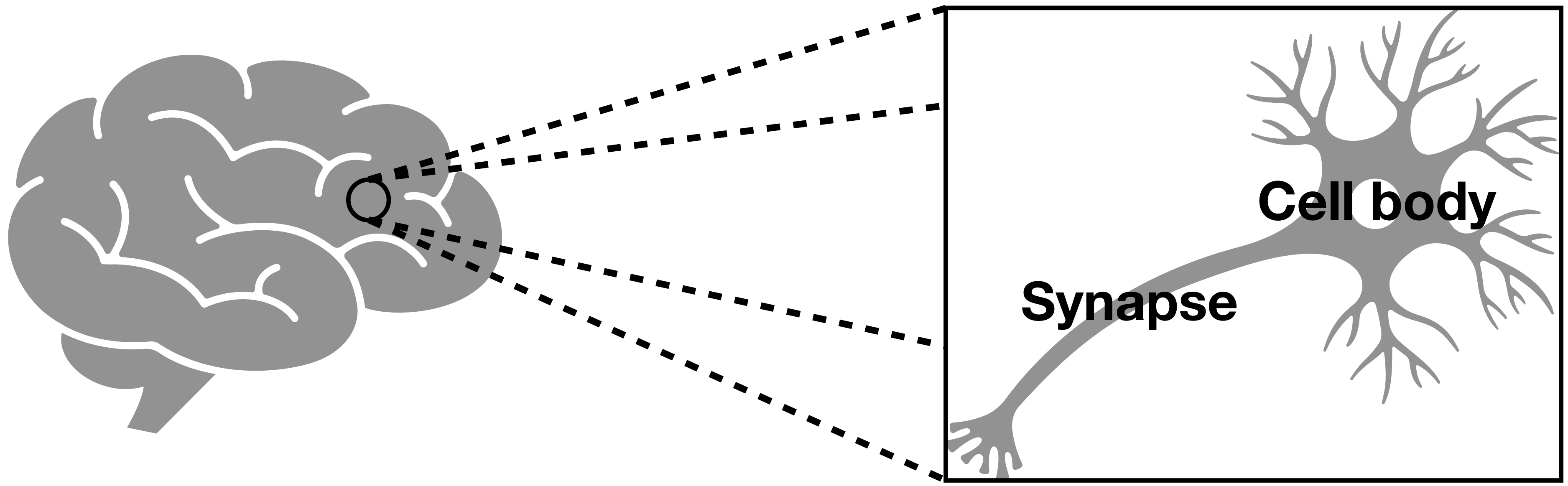
Why PyTorch?

TensorFlow Keras PYTORCH

- A lot easier to learn than to learn TensorFlow.
- Available to customize a model. This is usually infeasible with Keras.
- Public codes for deep learning research papers are usually written with PyTorch.

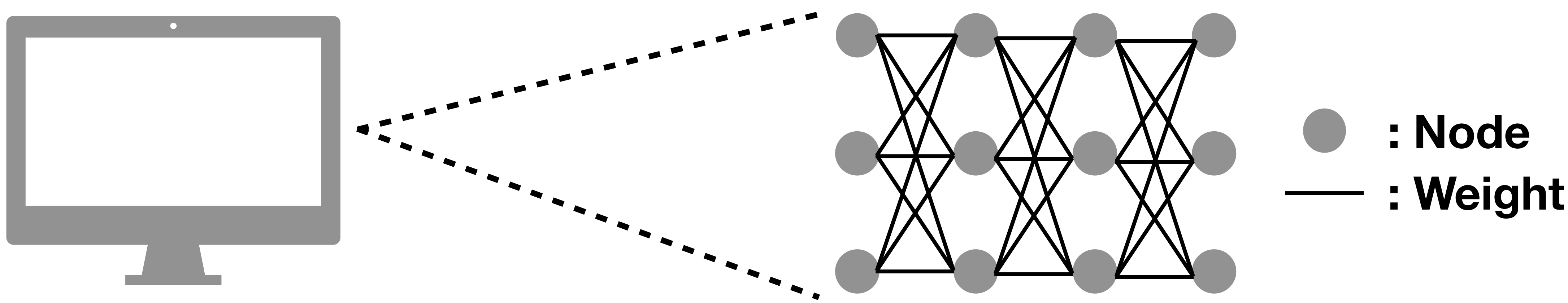
Basics

What is deep learning?



Brain

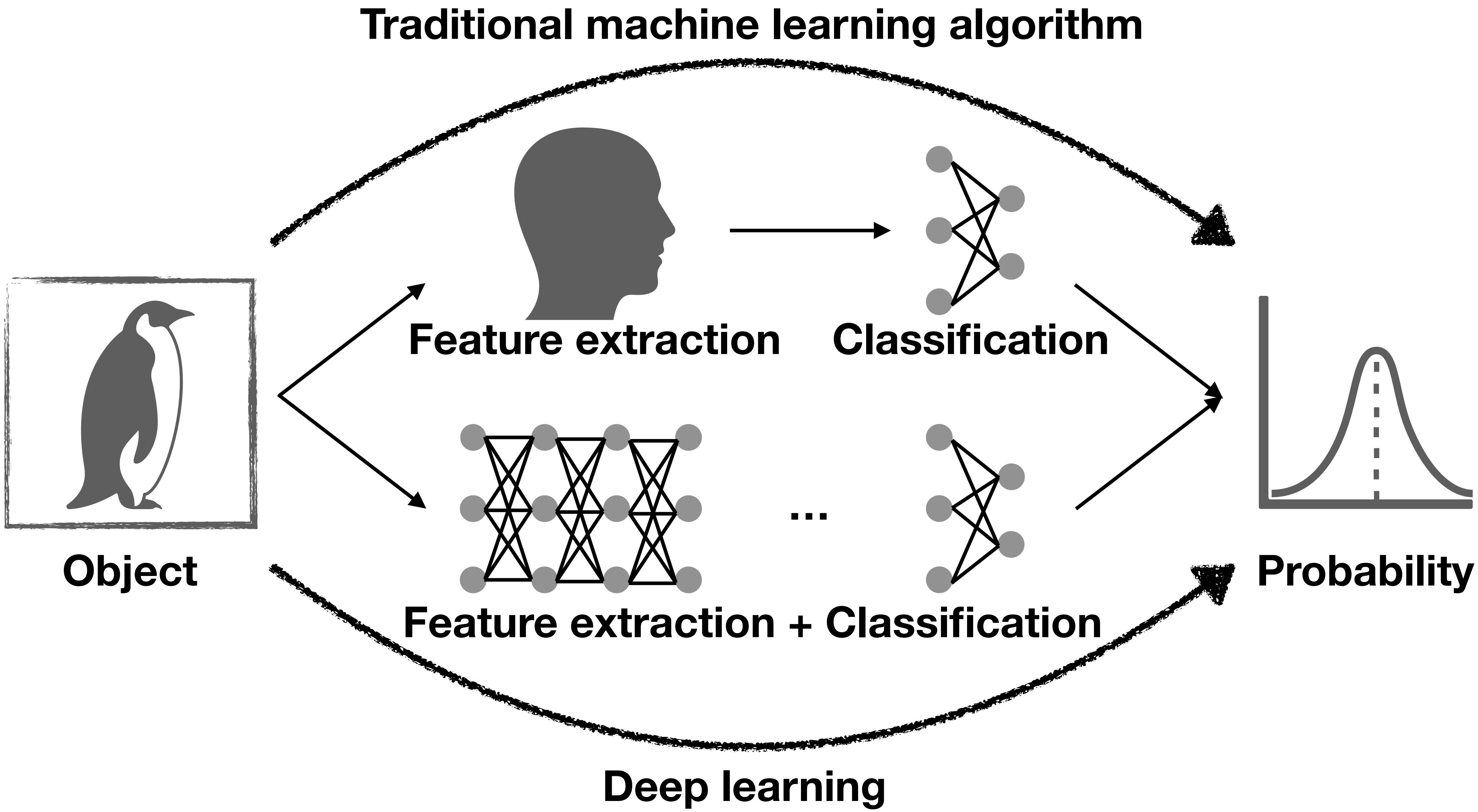
Neuron



Model

Neural Networks

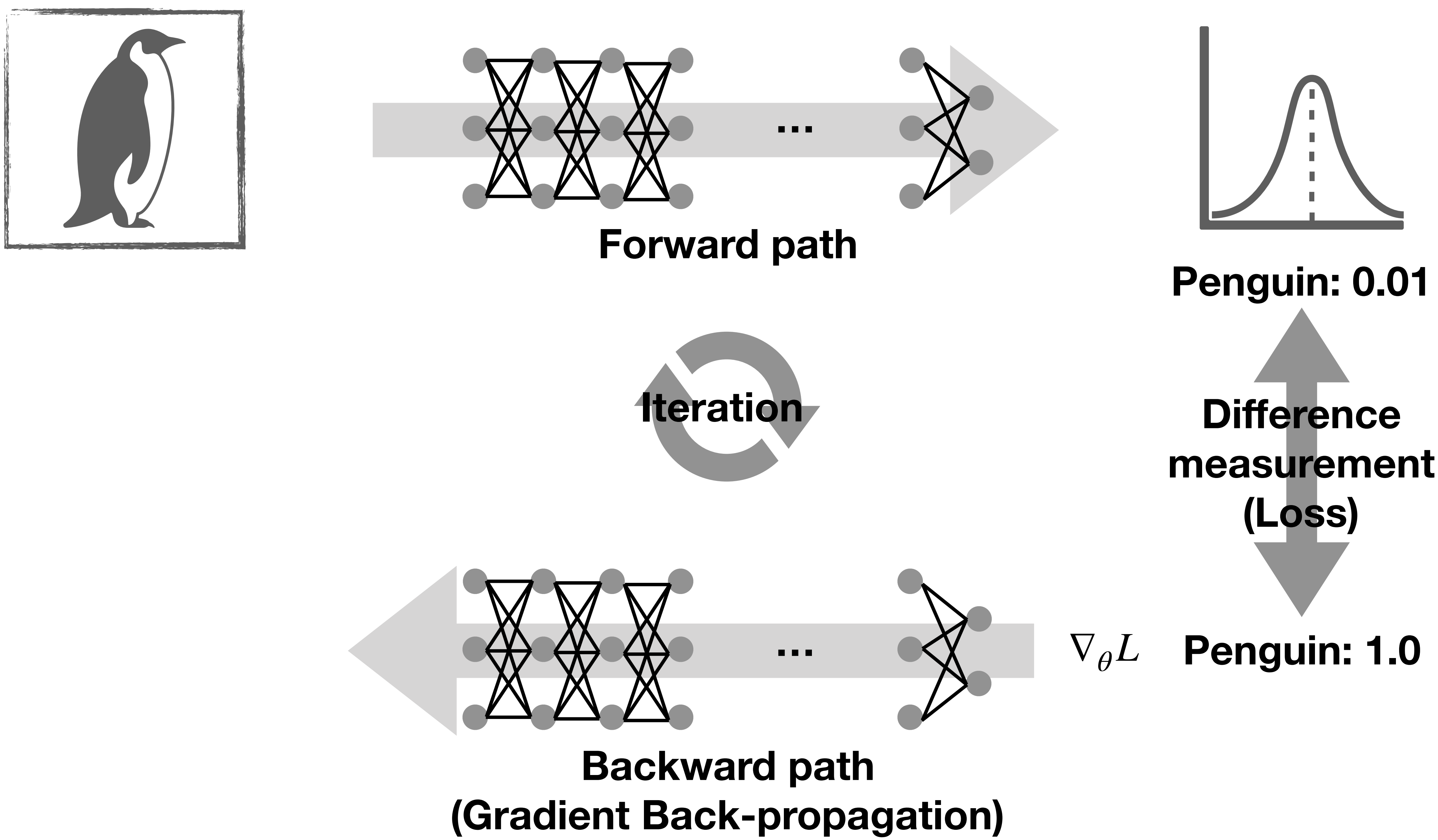
What is deep learning?



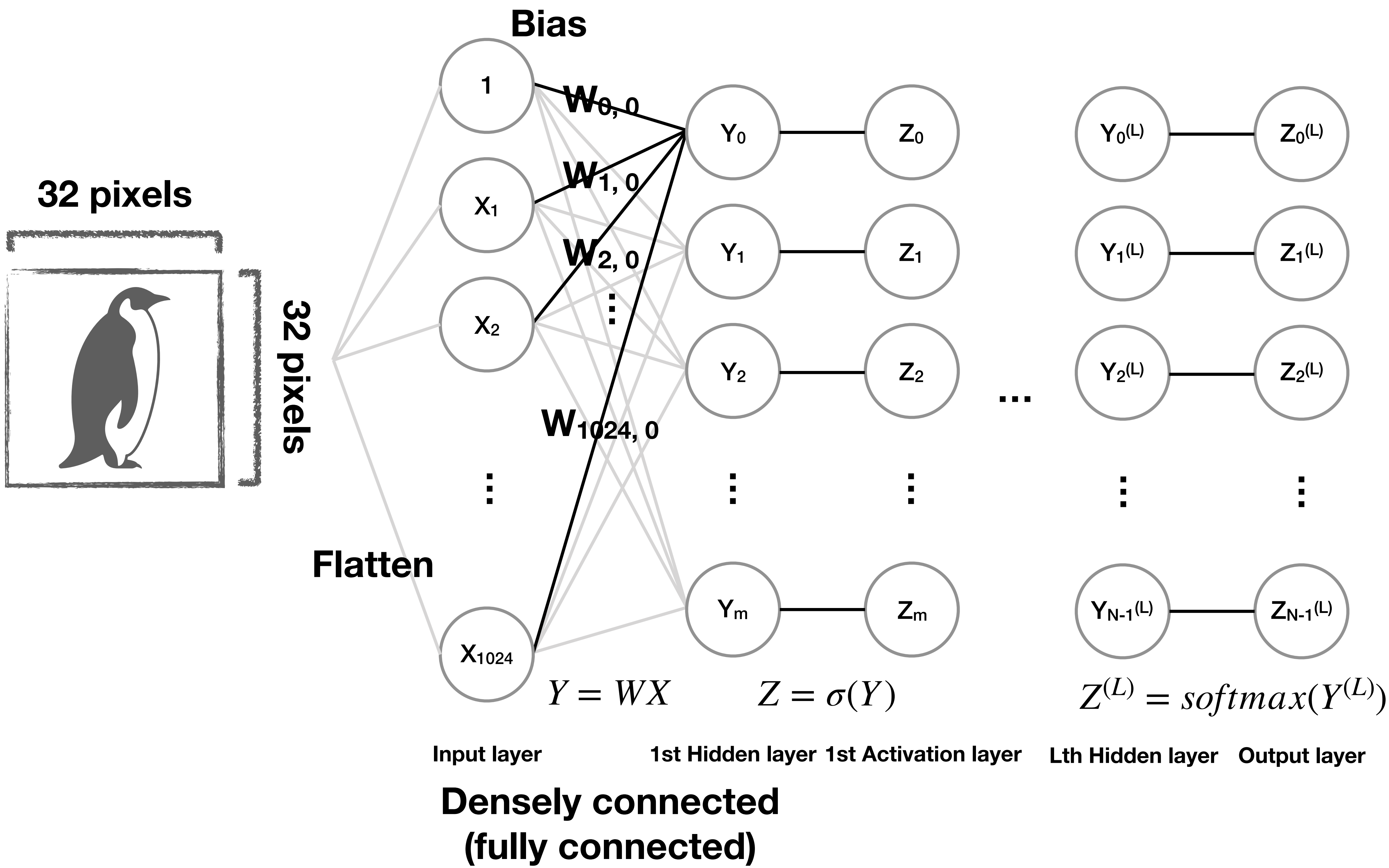
E.g. classification

Classification Model

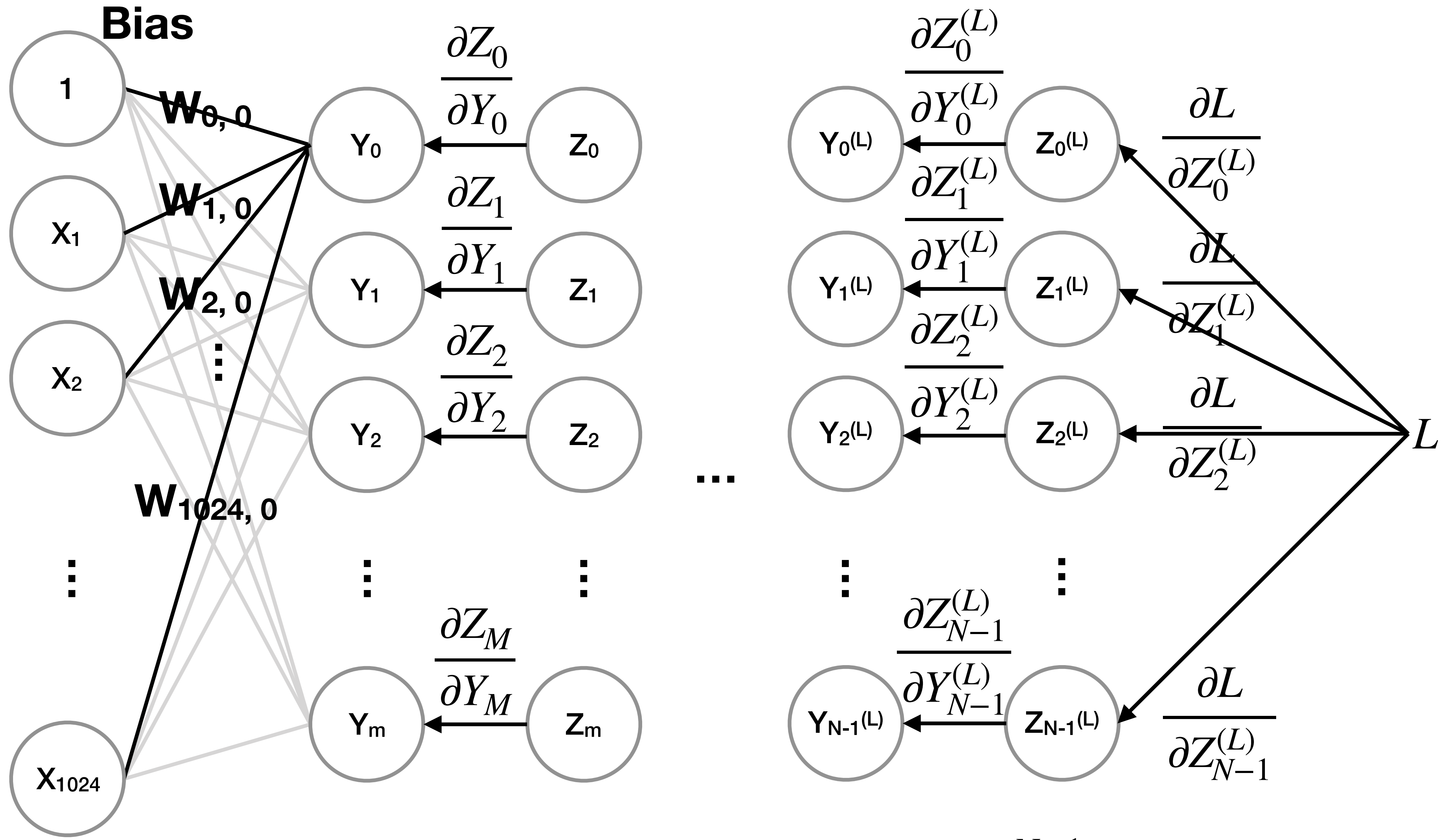
How does it work?



Forward path



Backward path



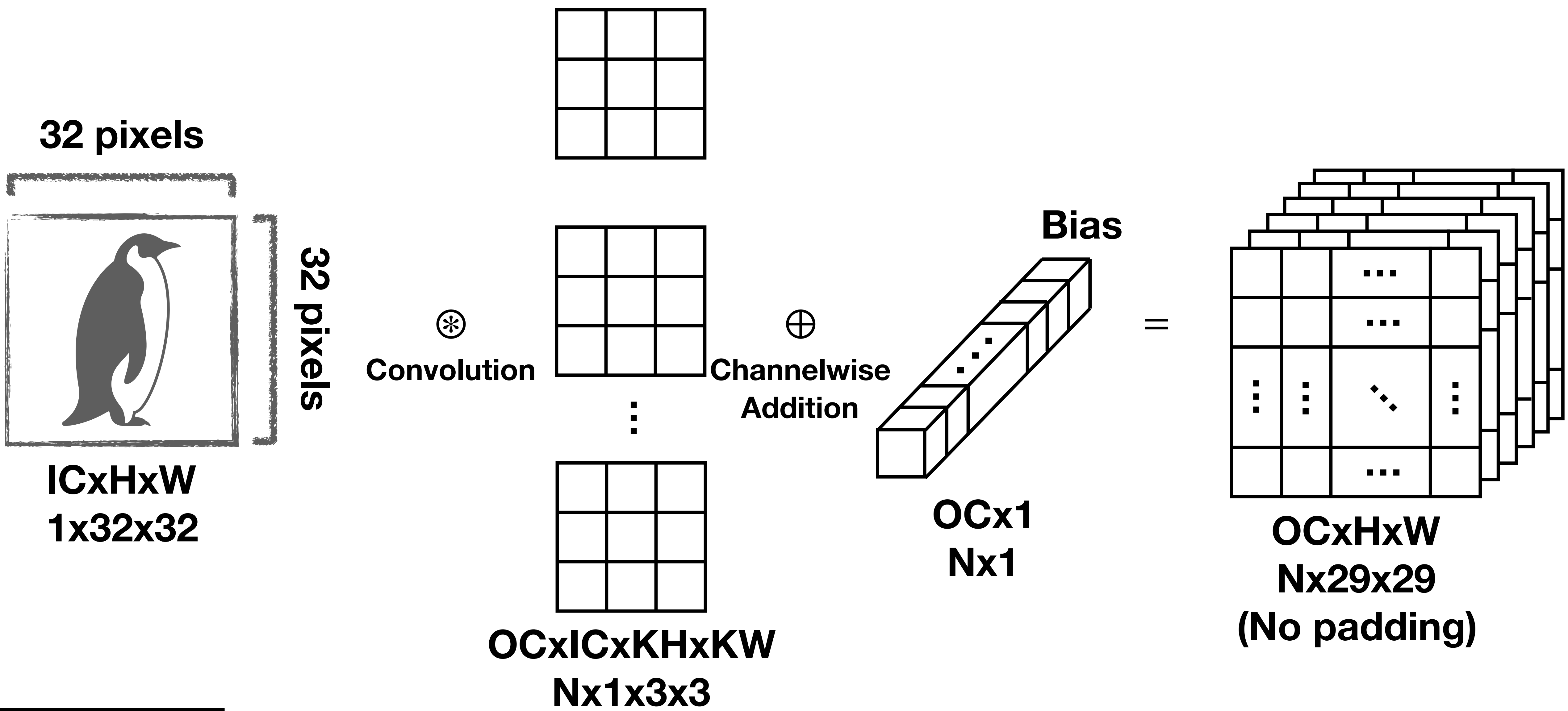
$$\frac{\partial L}{\partial W0,\ 0} = \frac{\partial L}{\partial Y_0}\frac{\partial Y_0}{W_{0,\ 0}}$$

$$W_{0,\ 0} = W_{0,\ 0} - \rho\frac{\partial L}{\partial W_{0,\ 0}}$$

$$L = -\sum_{k=0}^{N-1} t_k log Z_k = -\ log Z_i$$

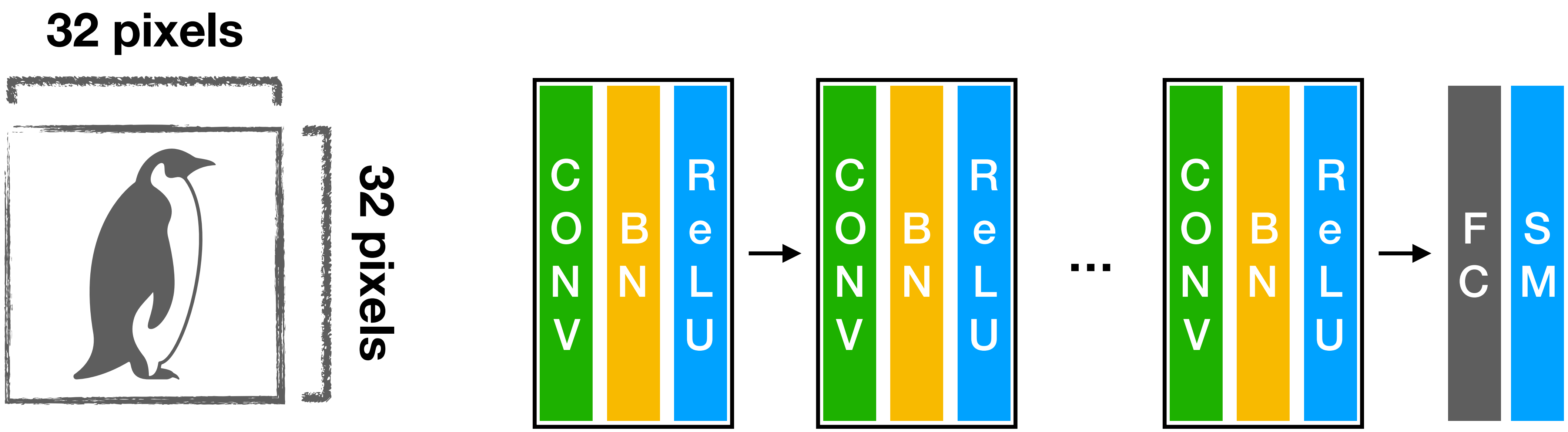
Cross-entropy loss

Convolution*



H: Height
W: Width
OC: Output Channel
IC: Input Channel
KH: Kernel Height
KW: Kernel Width

Convolutional neural network



BN: Batch Normalization
CONV: CONVolution
FC: Fully Connected layer
ReLU: Rectified Linear Unit
SM: SoftMax activation layer

Practice

MNIST database

Training set: 60,000 images and labels
Test set: 10,000 images and labels

The MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for training various image processing systems.

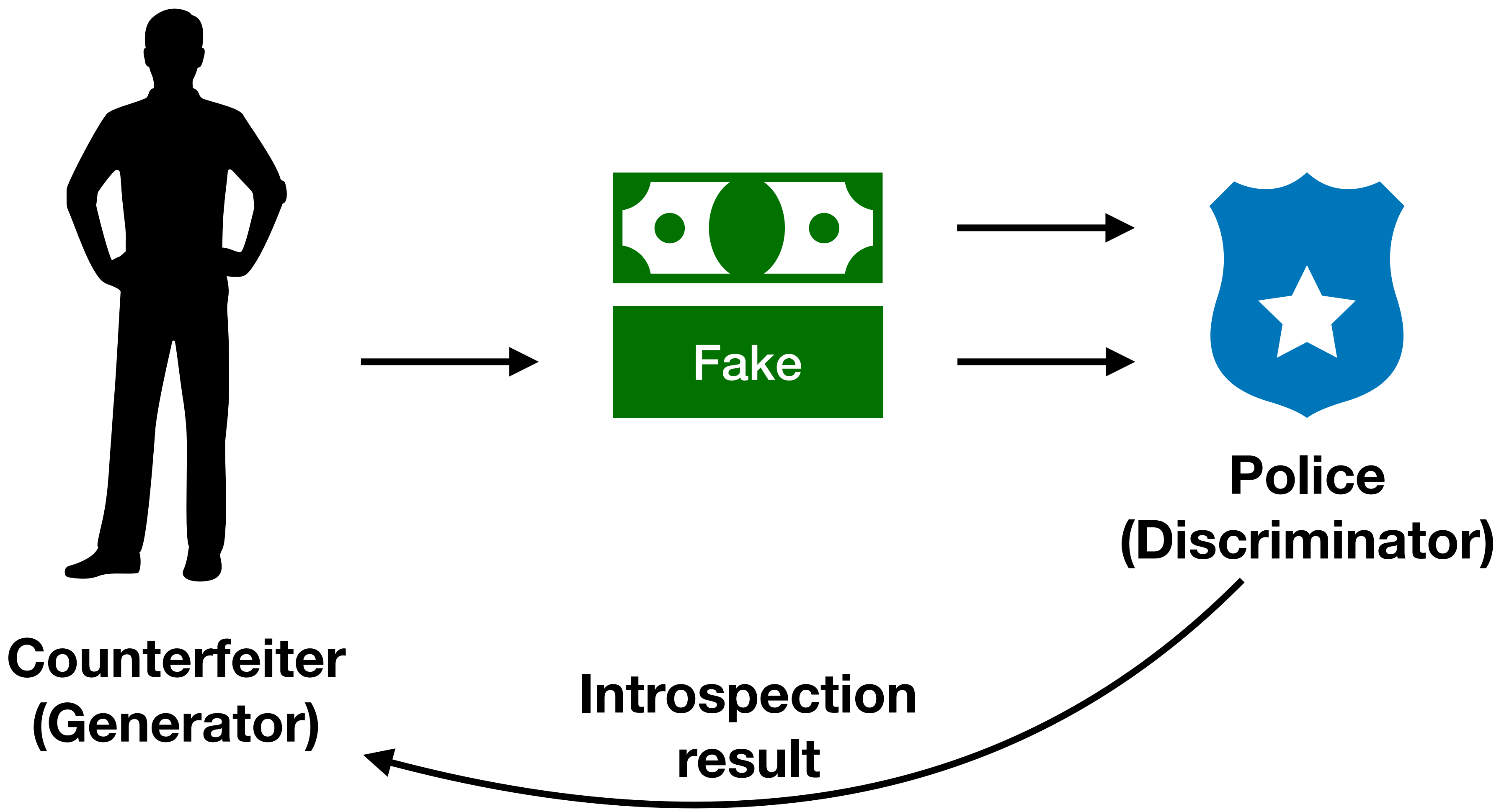
Generative Model

Various generative models

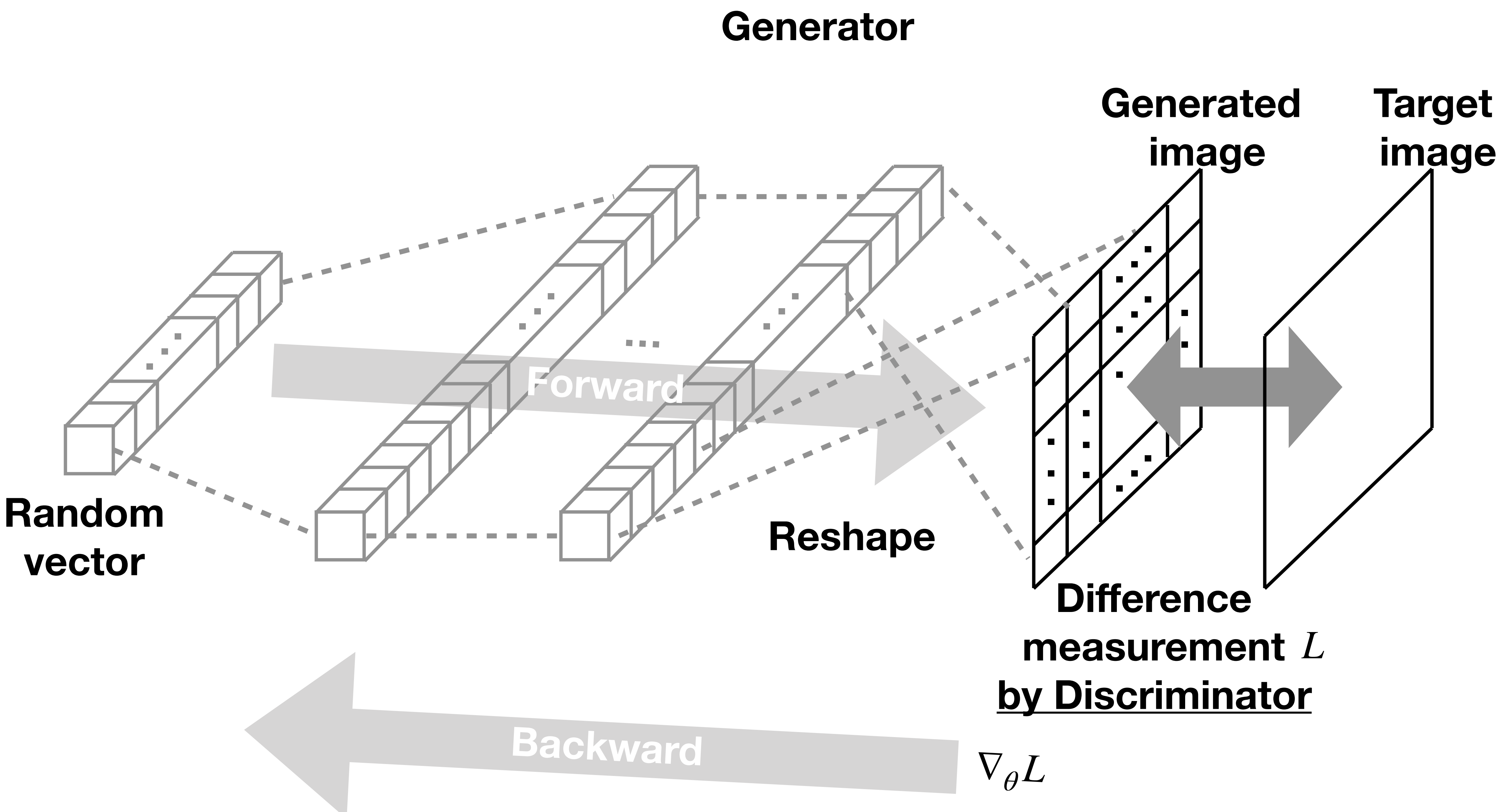
- Hidden Markov Model (HMM)
- Restricted Boltzmann Machine (RBM)
- Variational Auto-Encoder (VAE)
- Recurrent Neural Network (RNN)
- **Generative Adversarial Network (GAN)**

GAN

What is GAN?

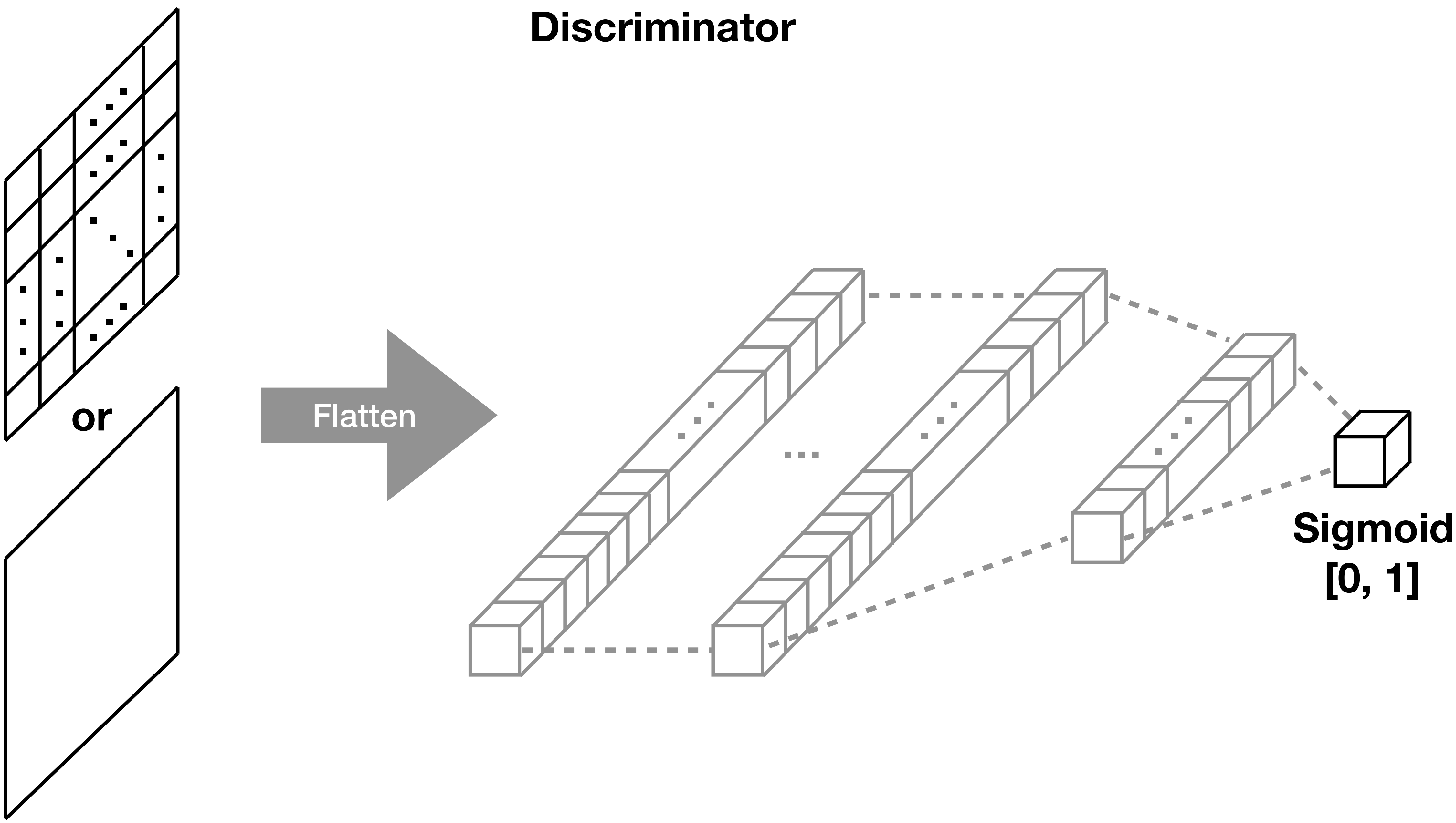

Fake
Police
(Discriminator)
Counterfeiter
(Generator)
Introspection
result

How does GAN work?



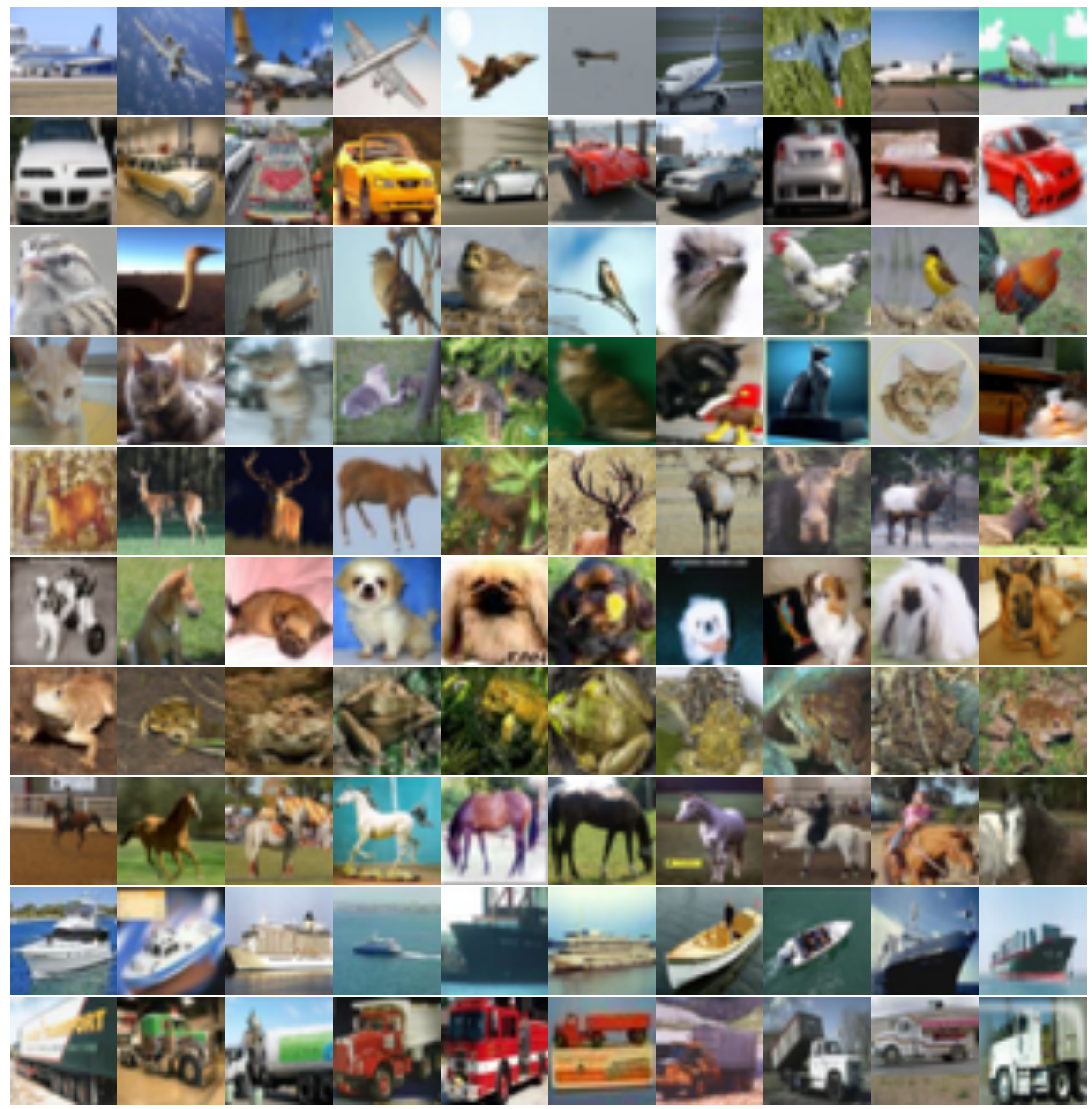
E.g. image generation model

How does GAN work?



Practice

CIFAR10

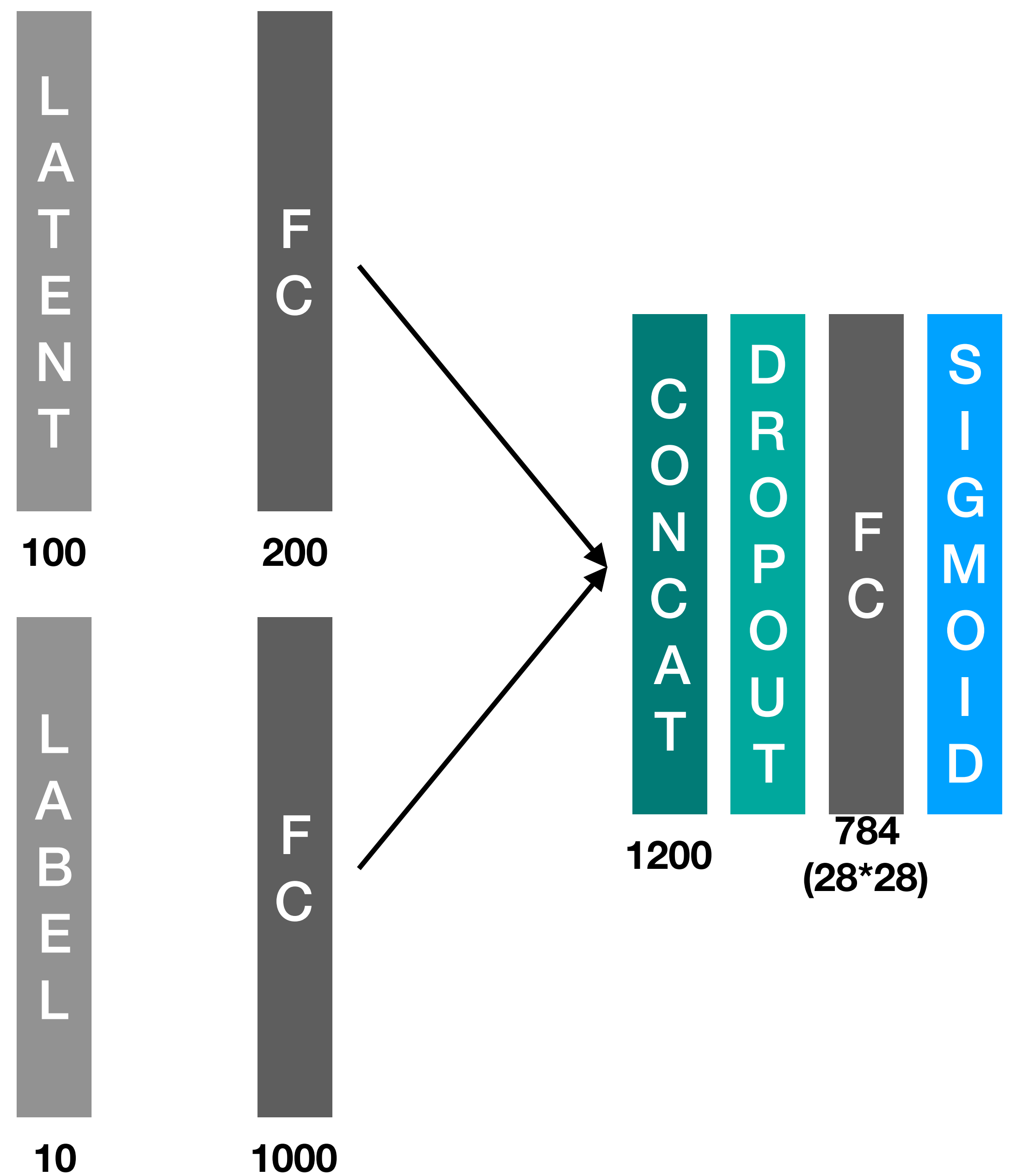


Credit. Learning Multiple Layers of Features from Tiny Images, Alex Krizhevsky, 2009.

The CIFAR-10 dataset consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images.

Conditional GAN

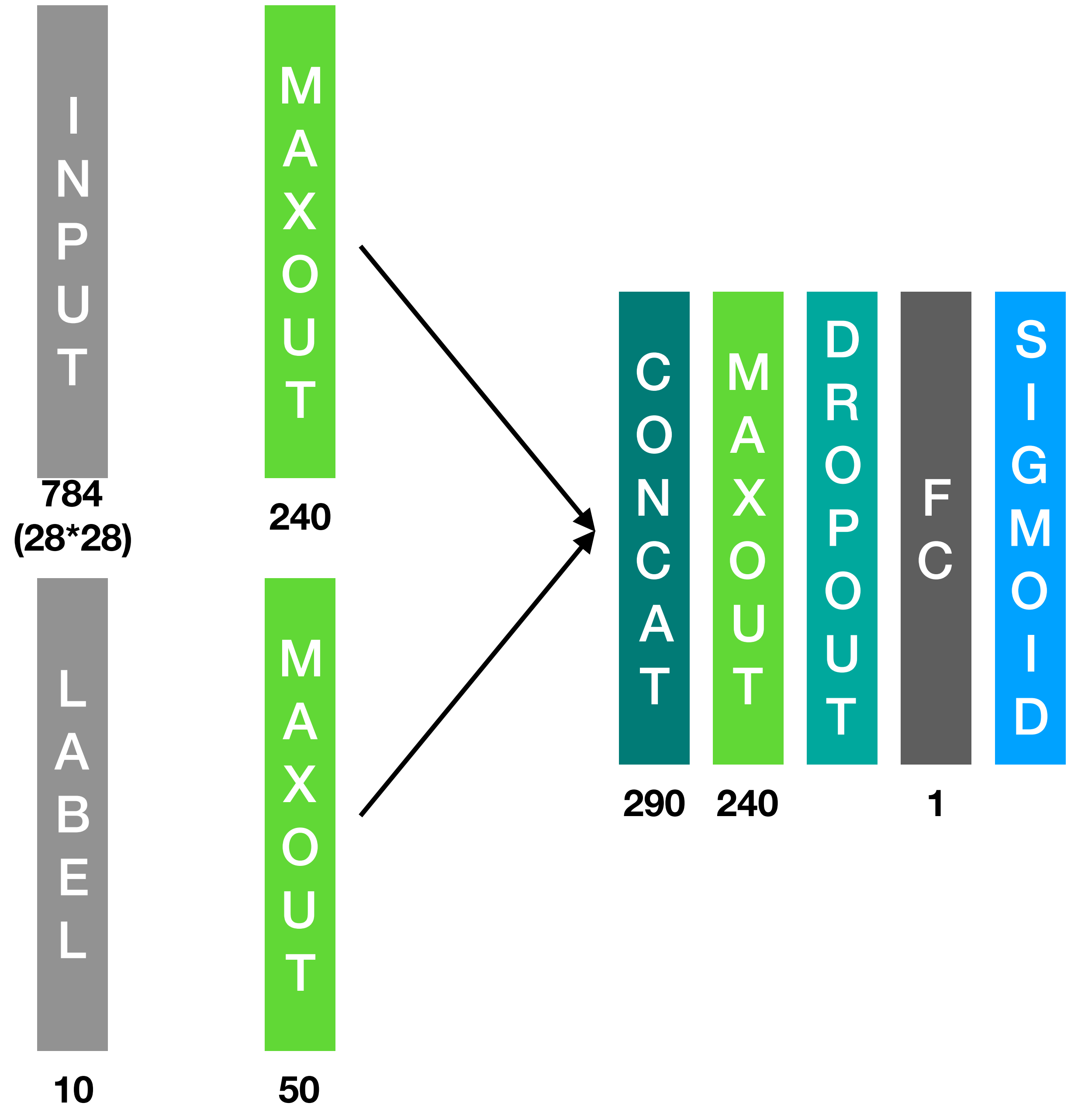
Generator



FC: Fully Connected layer

Note. This model is for MNIST dataset. LABEL is one-hot encoded label. Dropout rate is 0.5.

Discriminator

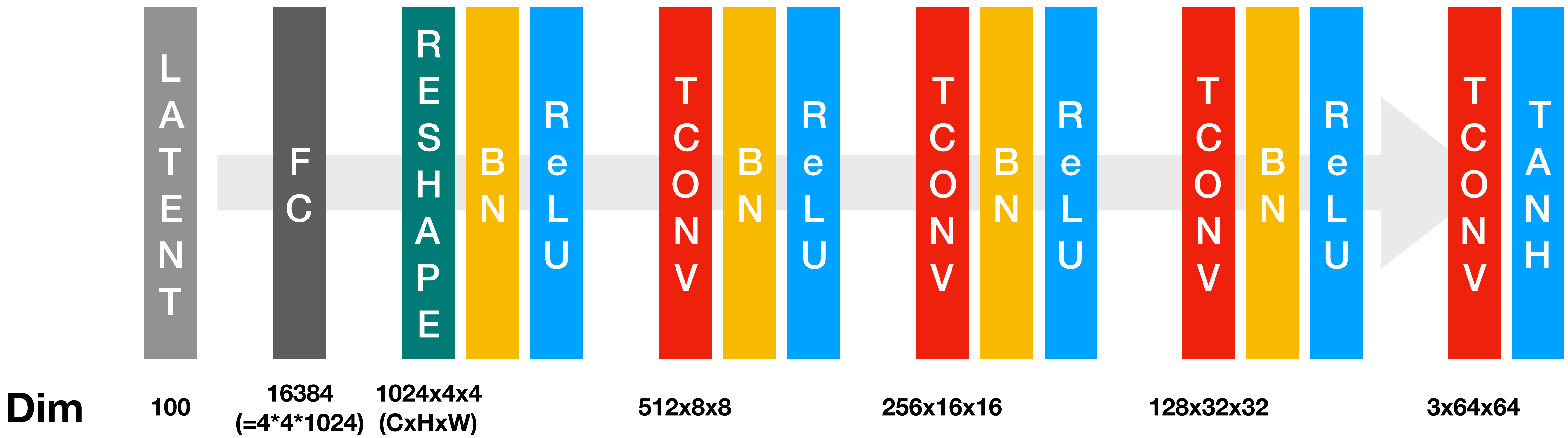


FC: Fully Connected layer

Note. This model is for MNIST dataset. LABEL is one-hot encoded label. Dropout rate is 0.5. MAXOUT layer includes dropout layer implicitly with rate 0.5. MAXOUT parameter k is set to 5 except for the last MAXOUT set to be 4.

DCGAN

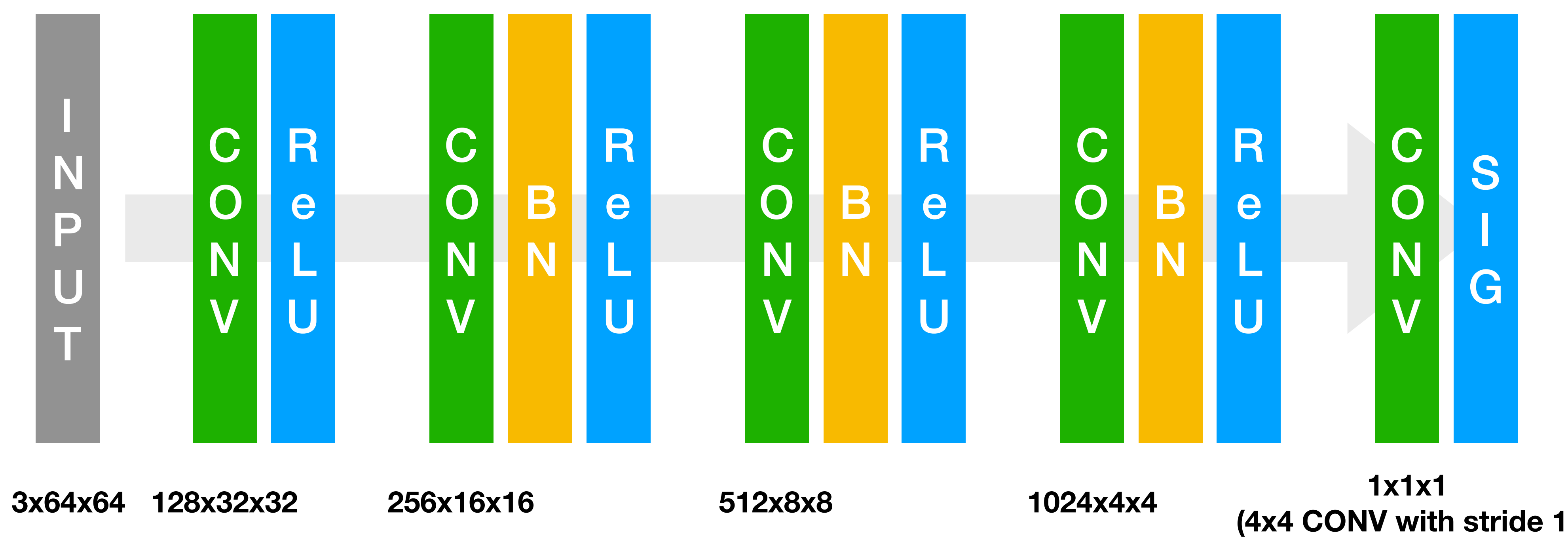
Generator



BN: Batch Normalization
C: Channel
FC: Fully Connected layer
H: Height
ReLU: Rectified Linear Unit
TCONV: 5x5 Transposed CONVolution (stride 2)
W: Width

Note. This is for the case of LSUN dataset. The number of TCONV layer can be varied with your target dataset.

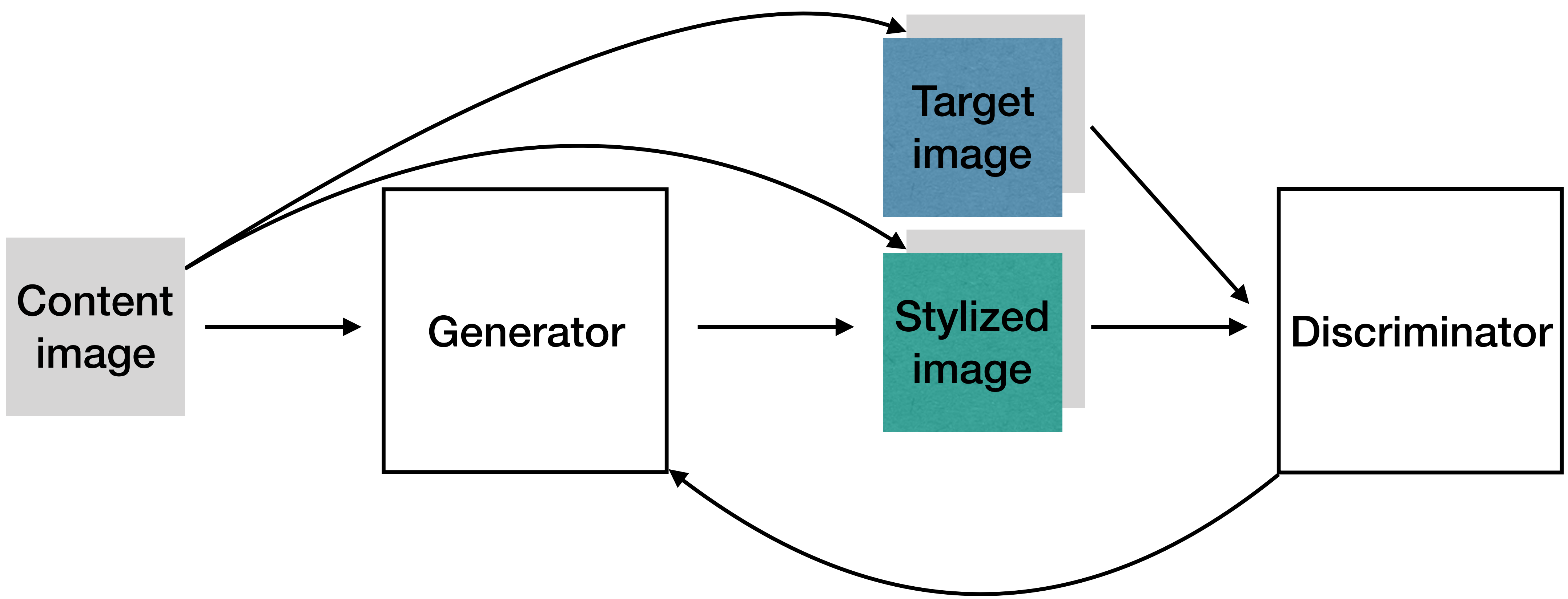
Discriminator



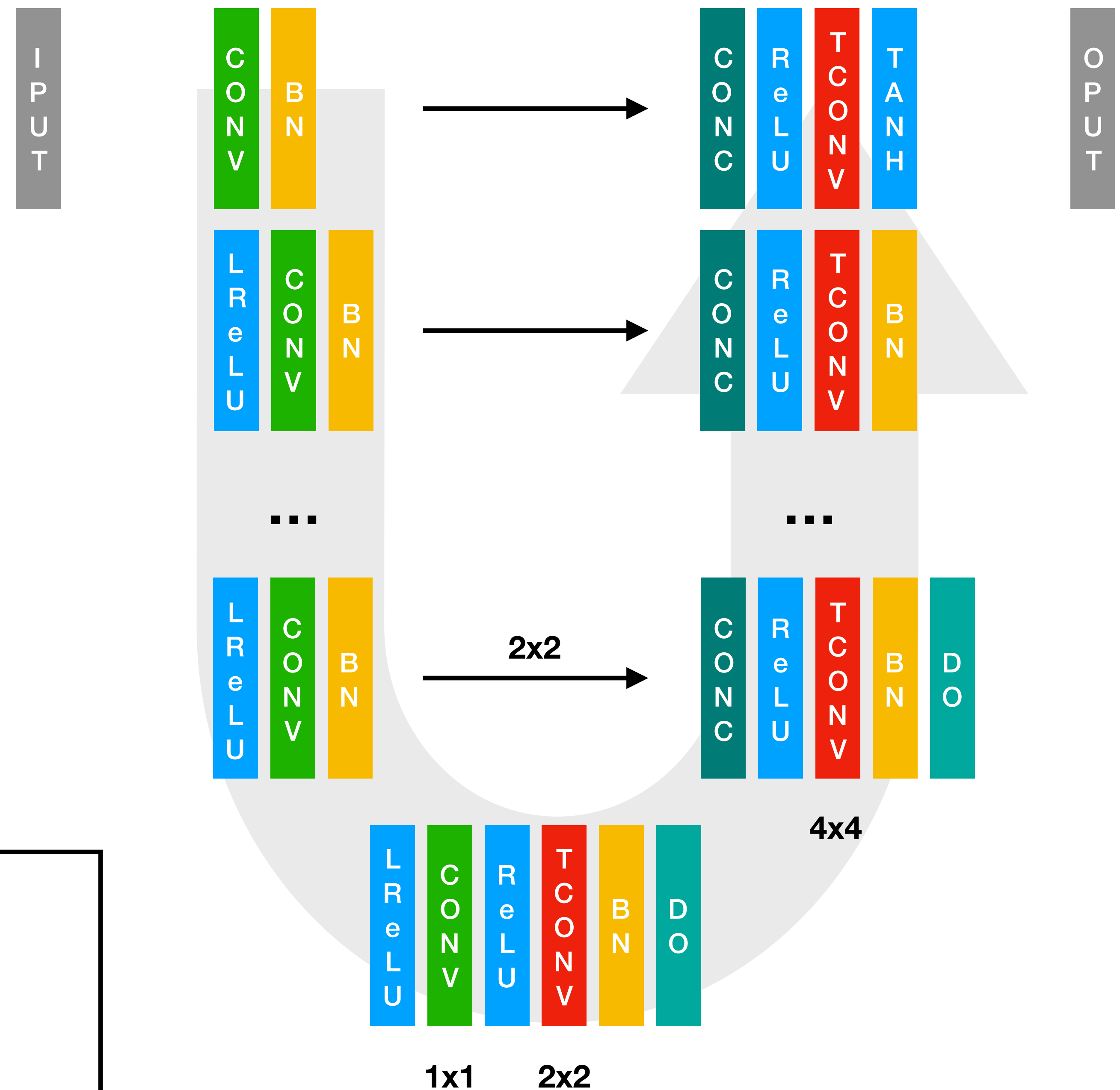
BN: Batch Normalization
C: Channel
H: Height
ReLU: Rectified Linear Unit
CONV: 5x5 CONVolution (stride 2)
W: Width

pix2pix

What is pix2pix?



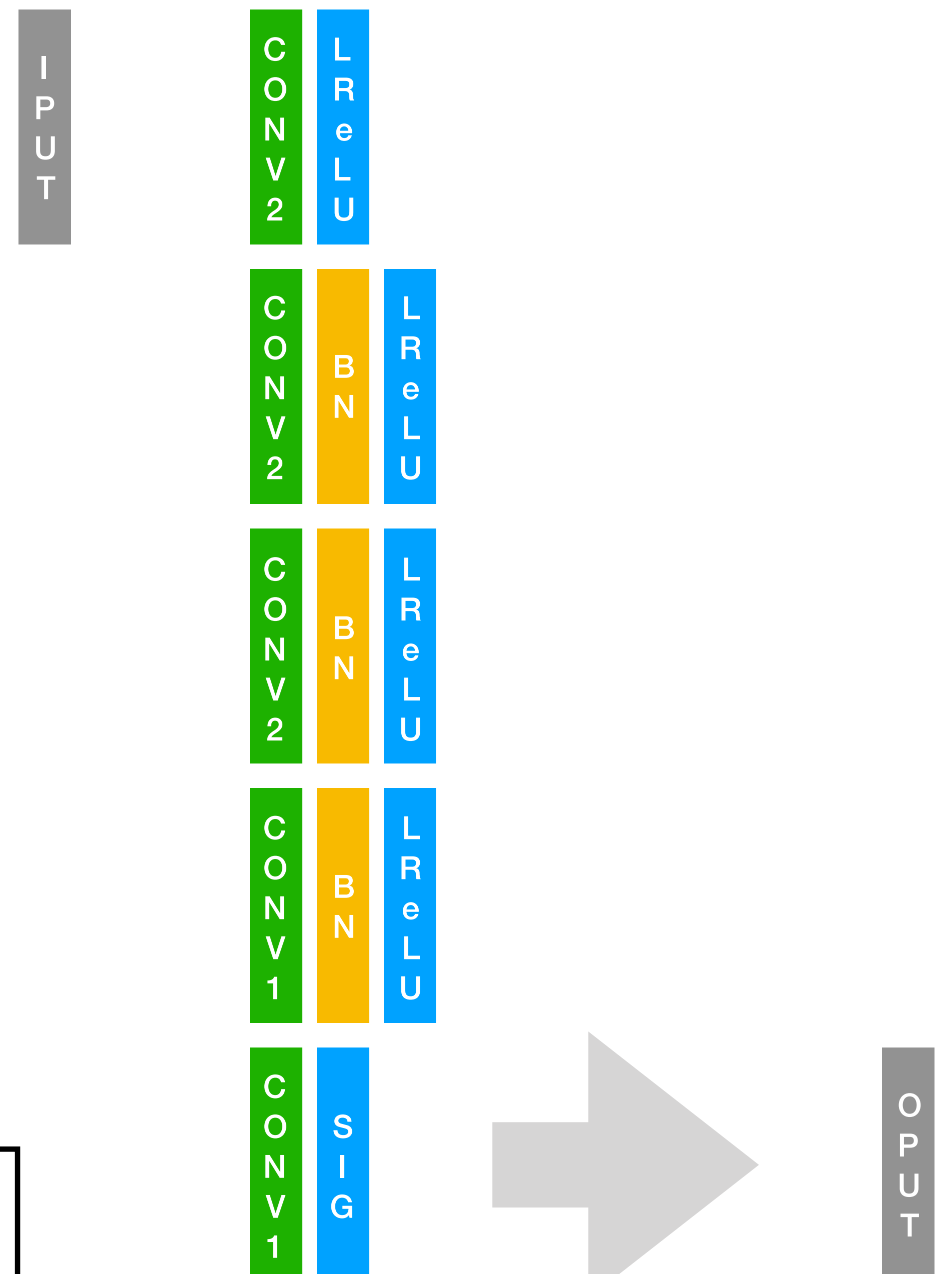
Generator



BN: Batch Normalization
CONC: CONCatenation
CONV: 4x4 CONVolution (stride 2)
DO: DropOut (p = 0.5)
IPUT: InPUT
LReLU: Leaky ReLU with slope 0.2
OPUT: OutPUT
ReLU: Rectified Linear Unit
TCONV: 4x4 Transposed CONV (stride 2)

Note. Dropout is applied where feature map size is 2x2, 4x4, and 8x8 in the decoder part.

Discriminator



1 channel output

BN: Batch Normalization
CONV1: 4x4 CONVolution (stride 1)
CONV2: 4x4 CONVolution (stride 2)
IPUT: InPUT
LReLU: Leaky ReLU with slope 0.2
OPUT: OutPUT

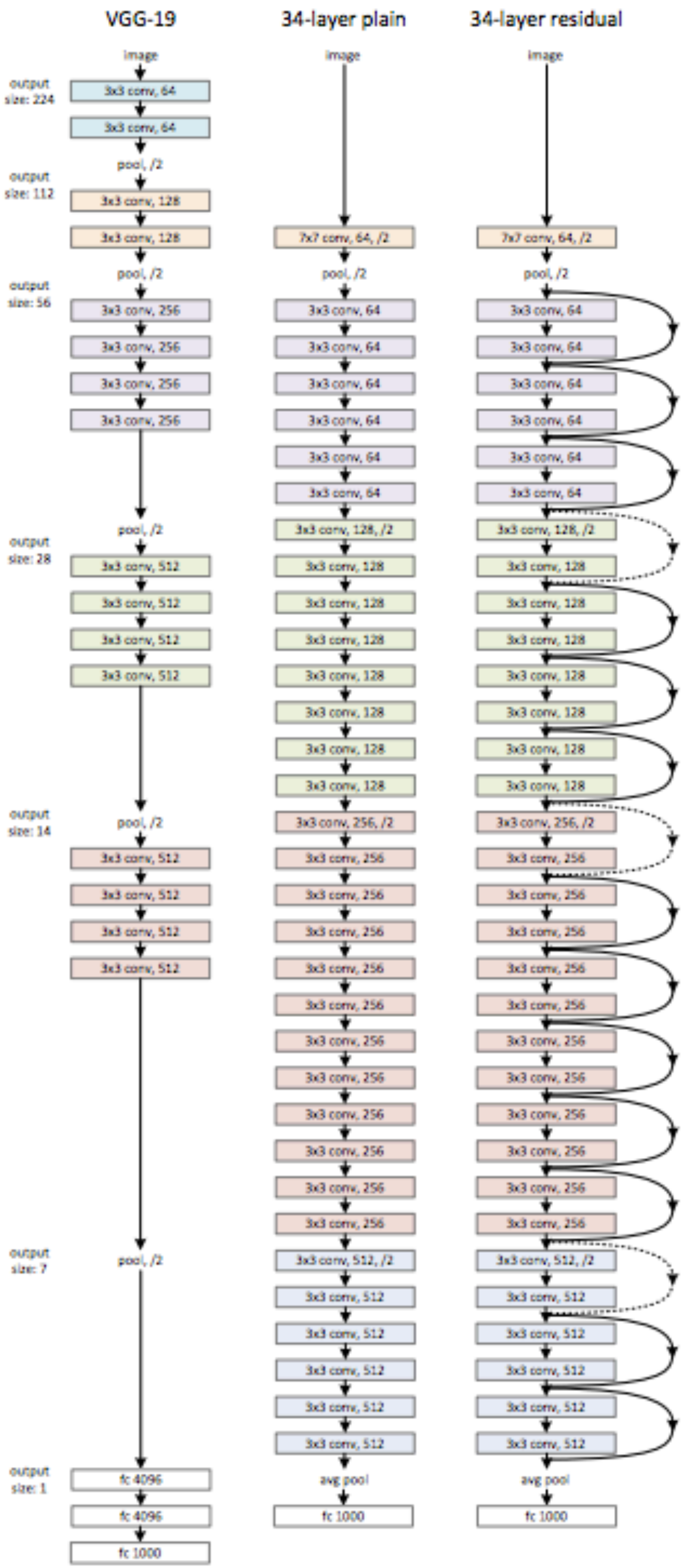
Note. This is for the case of 70x70 receptive field. Layers should be varied to change receptive field size.

Practice

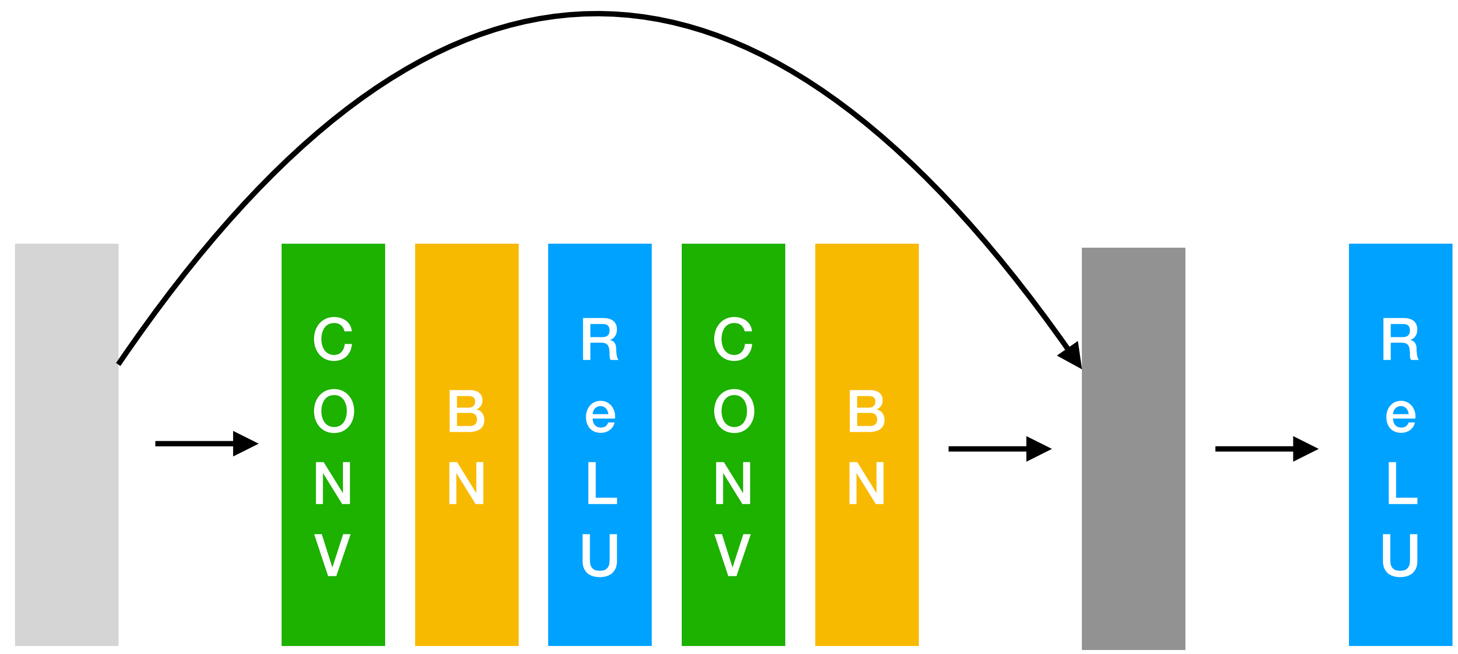
Intermediate

Classification Model

Residual Network



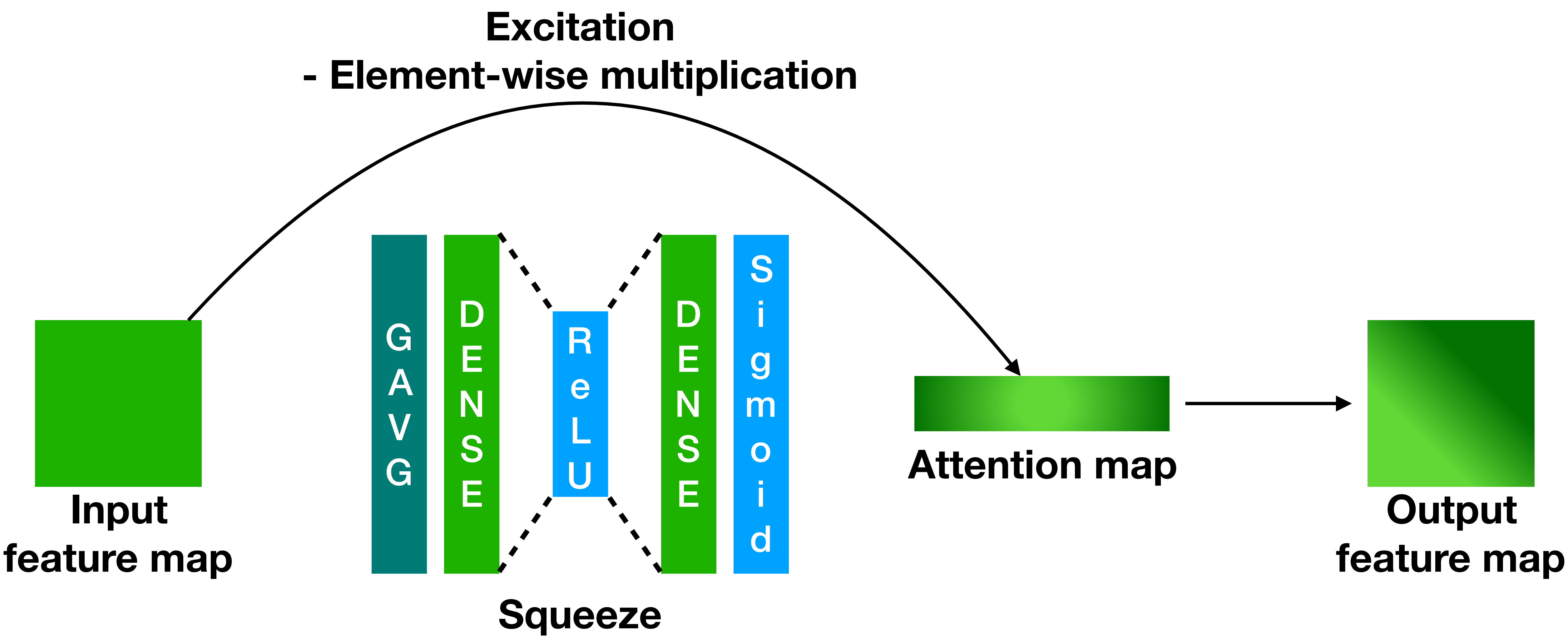
Element-wise summation



When feature map size changes, 1x1 convolution with stride 2 is applied such that the shortcut connection can be presented every two convolution layers.

Squeeze-and-Excitement Network

Squeeze-and-Excitation Block

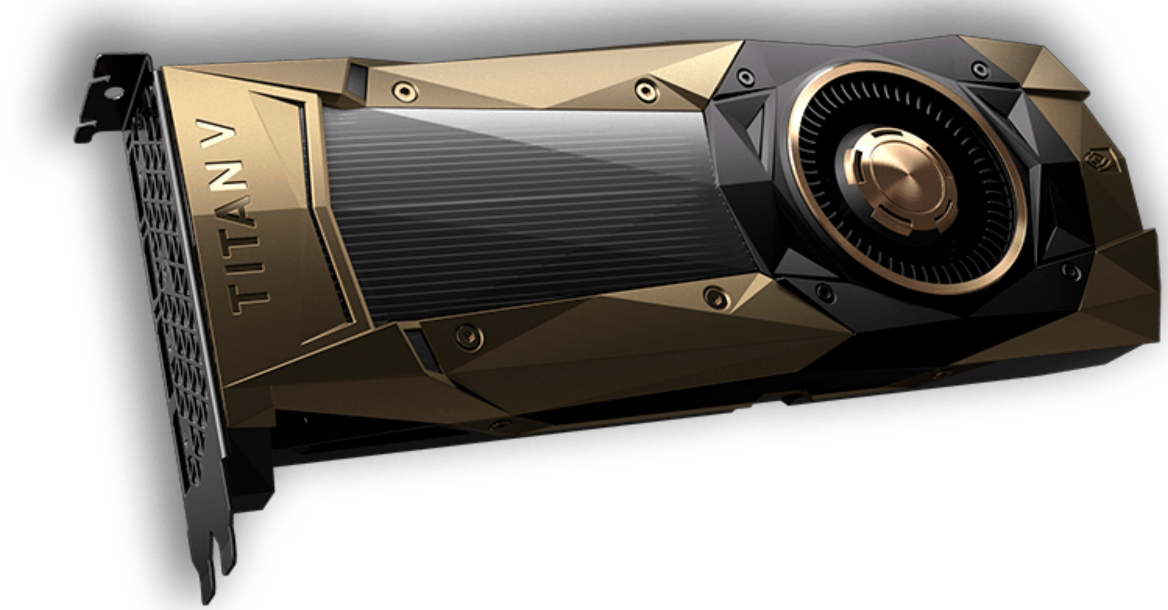


DENSE: Fully-connected layer
GAVG: Global AVeraGe pooling
ReLU: Rectified Linear Unit

Appendix

CPU vs. GPU

credit. hothardware.com

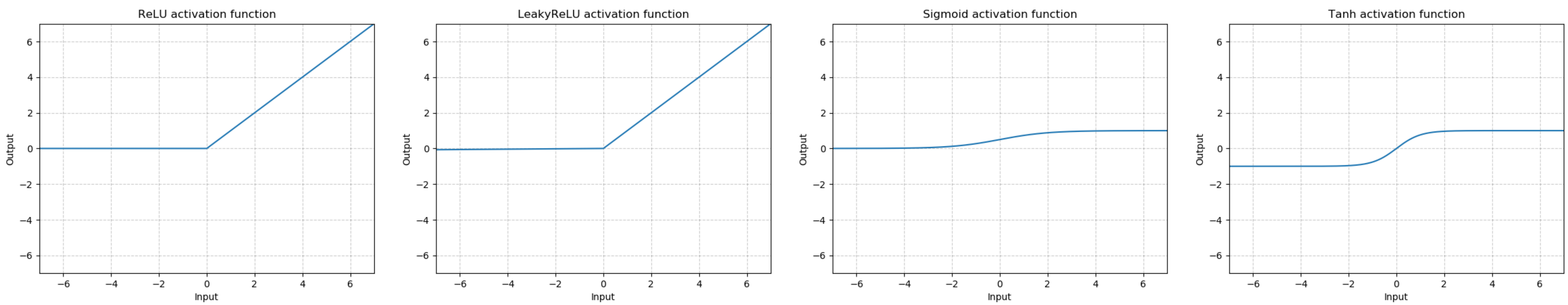


credit. NVIDIA

A central processing unit (CPU), also called a central processor or main processor, is the electronic circuitry within a computer that carries out the instructions of a computer program by performing the basic arithmetic, logic, controlling, and input/output operations specified by the instructions.

A graphics processing unit (GPU) is a specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device.

Activation functions

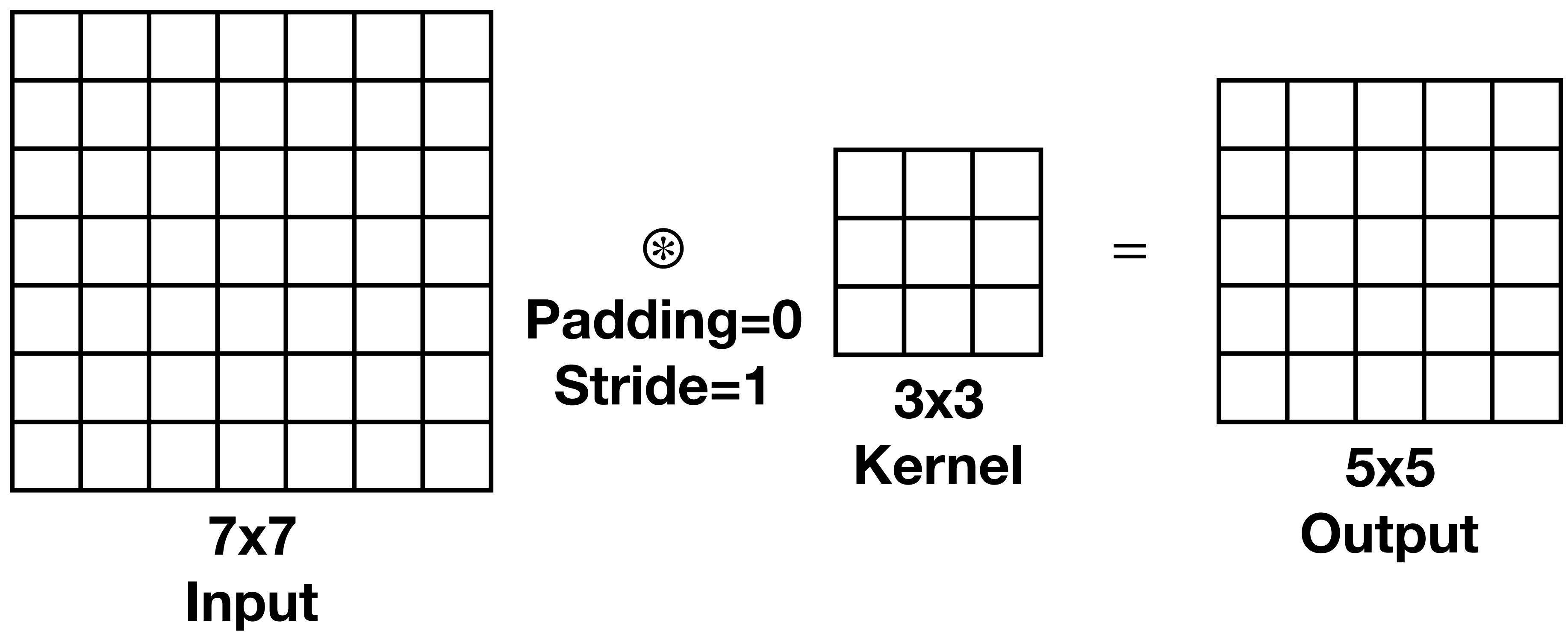


- ReLU (Rectified linear unit) : $max(0, x)$
- Leaky ReLU : $max(0, x) \; + \; negative\ slope \times min(0, x)$
- Hyperbolic tangent (tanh): $\dfrac{e^x - e^{-x}}{e^x + e^{-x}}$
- Logistic sigmoid : $\dfrac{1}{1 + e^{-x}} = \dfrac{e^x}{e^x + 1} = \dfrac{1}{2} + \dfrac{1}{2} tanh(\dfrac{x}{2})$

Loss functions

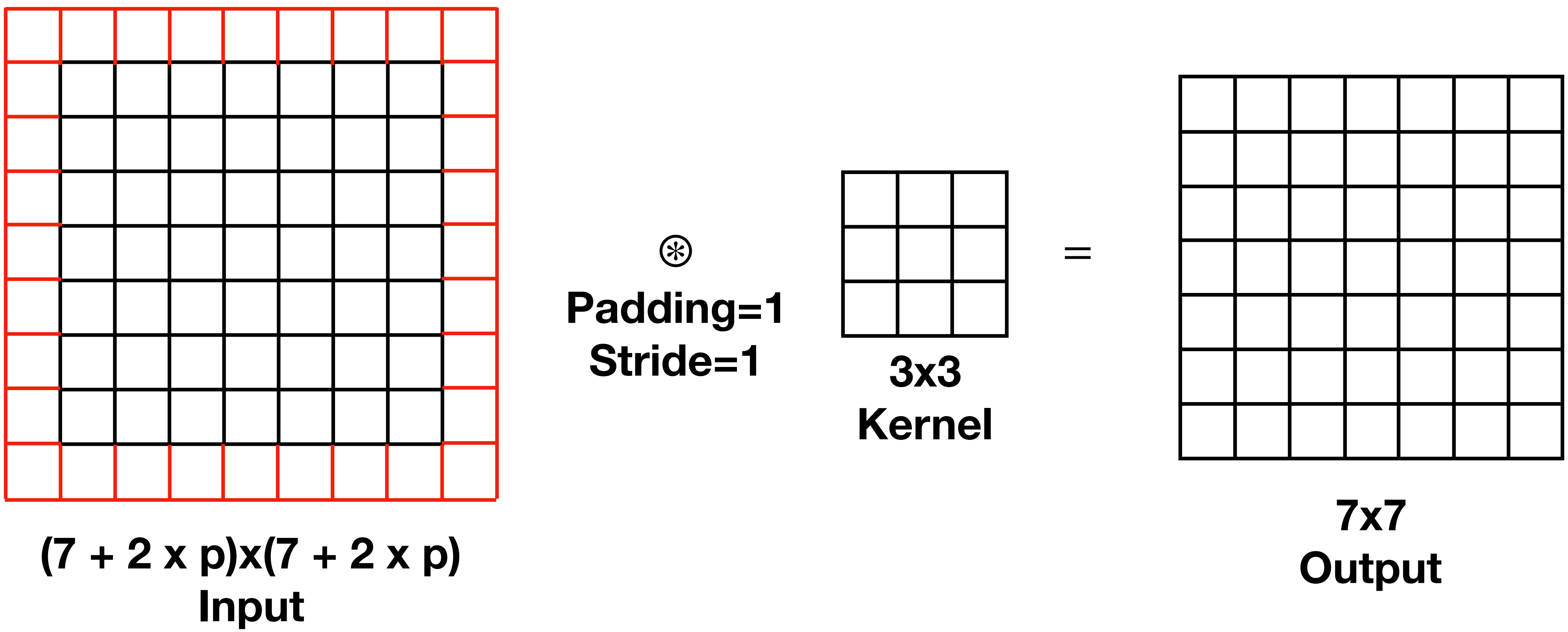
- Binary cross entropy loss (BCE) : $-\log \hat{p}(y_c|x) - \sum_{k=1,\ k\neq c}^{C} \log[1 - \hat{p}(y_k|x)]$
- Cross entropy loss (CE) : $\mathbb{E}_{p(y|x)}[-\log \hat{p}(y|x)] = -\sum_{c=1}^{C} p(y_c|x) \log \hat{p}(y_c|x) = -\log \hat{p}(y_c|x)$
- Mean squared error loss (MSE) : $\frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(x_{i,\ j} - \hat{x}_{i,\ j})^2$
- Mean absolute error loss (MAE) : $\frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}|x_{i,\ j} - \hat{x}_{i,\ j}|$

Convolution



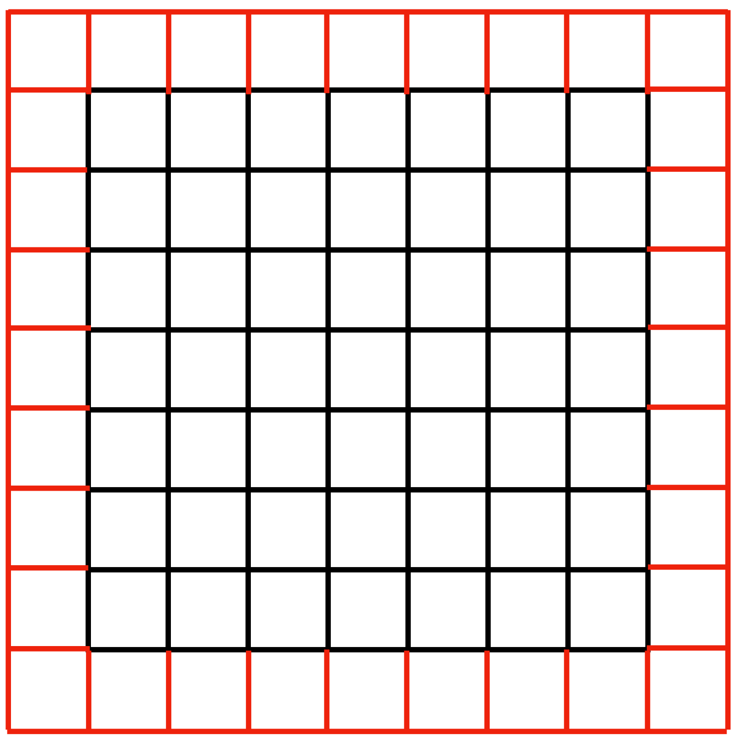
$$o = i - k + 1$$

Convolution



$$o = i - k + 2p + 1$$

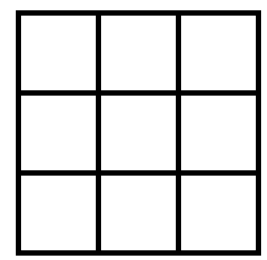
Convolution



(7 + 2 x p)x(7 + 2 x p)
Input

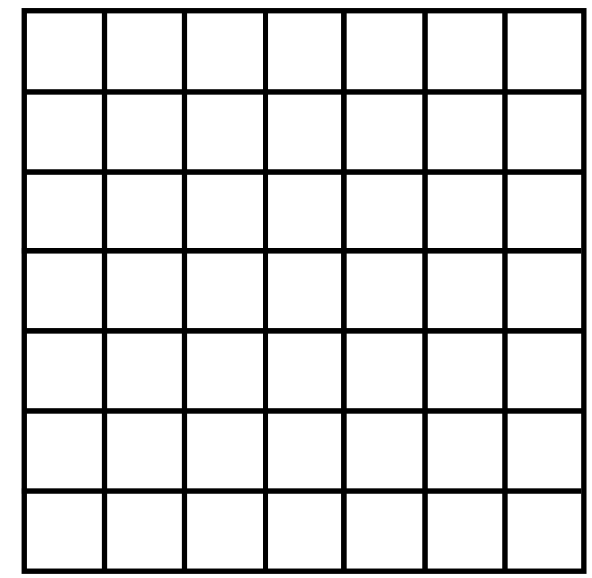
$\circledast$

Padding=1
Stride=2



3x3
Kernel

$=$



4x4
Output

$$o = \left\lfloor \frac{i - k + 2p}{s} \right\rfloor + 1$$