Department of Computer Engineering
University of Kurdistan

Neural Networks (Graduate level)

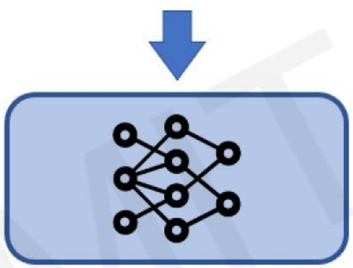
Deep Learning

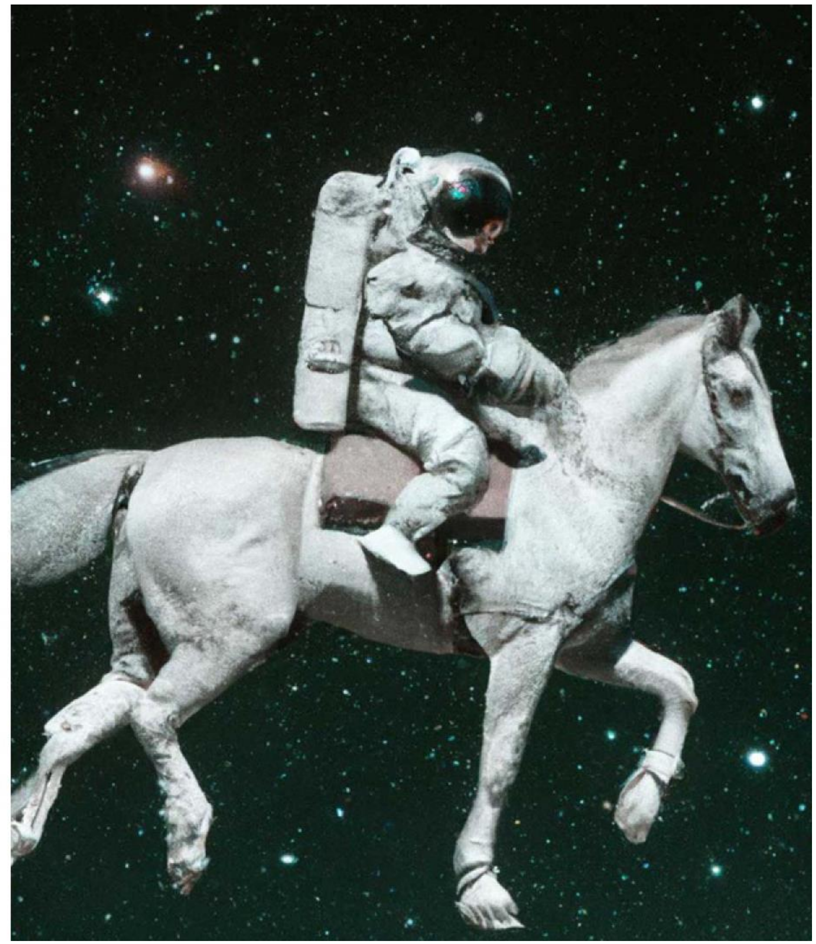
By: Dr. Alireza Abdollahpouri

Deep Learning

Generating Images from Natural Language

“A photo of an astronaut riding a horse.”

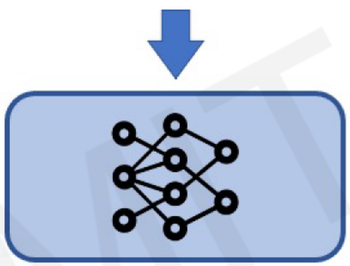




Deep Learning

Generating Language from Natural Language

“Write code in TensorFlow to train a neural network.”



Certainly! Here is an example of how you can use TensorFlow to train a neural network for binary classification:

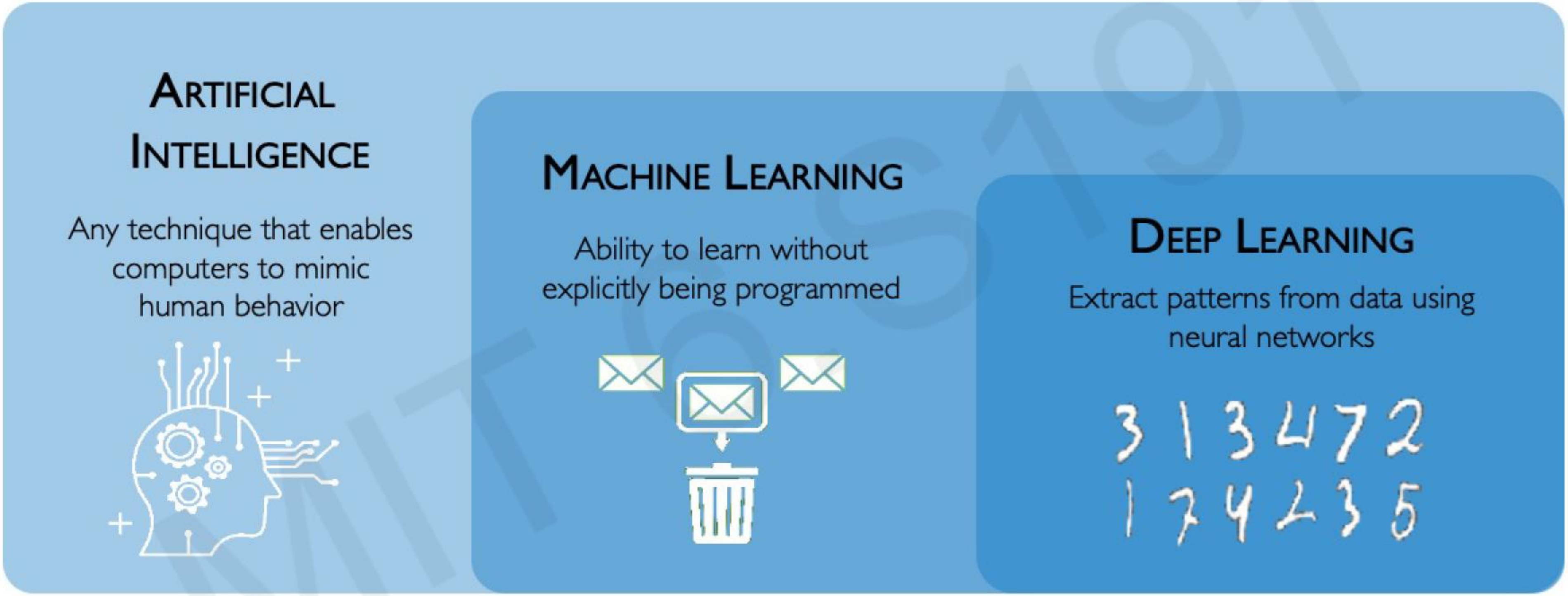
Copy code

```
import tensorflow as tf

# Load the data
(x_train, y_train), (x_test, y_test) = load_data()

# Define the model
model = tf.keras.Sequential([
    tf.keras.layers.Input(shape=x_train.shape[1:]),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=
['accuracy'])

# Train the model
model.fit(x_train, y_train, epochs=10, batch_size=32, validation_data=
(x_test, y_test))
```

This code assumes that you have a function `load_data()` that returns the training and test datasets as tuples of NumPy arrays. The input shape of the model is determined by the shape of the training data, and the model has two dense layers with 16 and 1 units, respectively, with ReLU and sigmoid activations. The model is then compiled with an Adam optimizer and a binary cross-entropy loss function, and is trained using the `fit()` method.

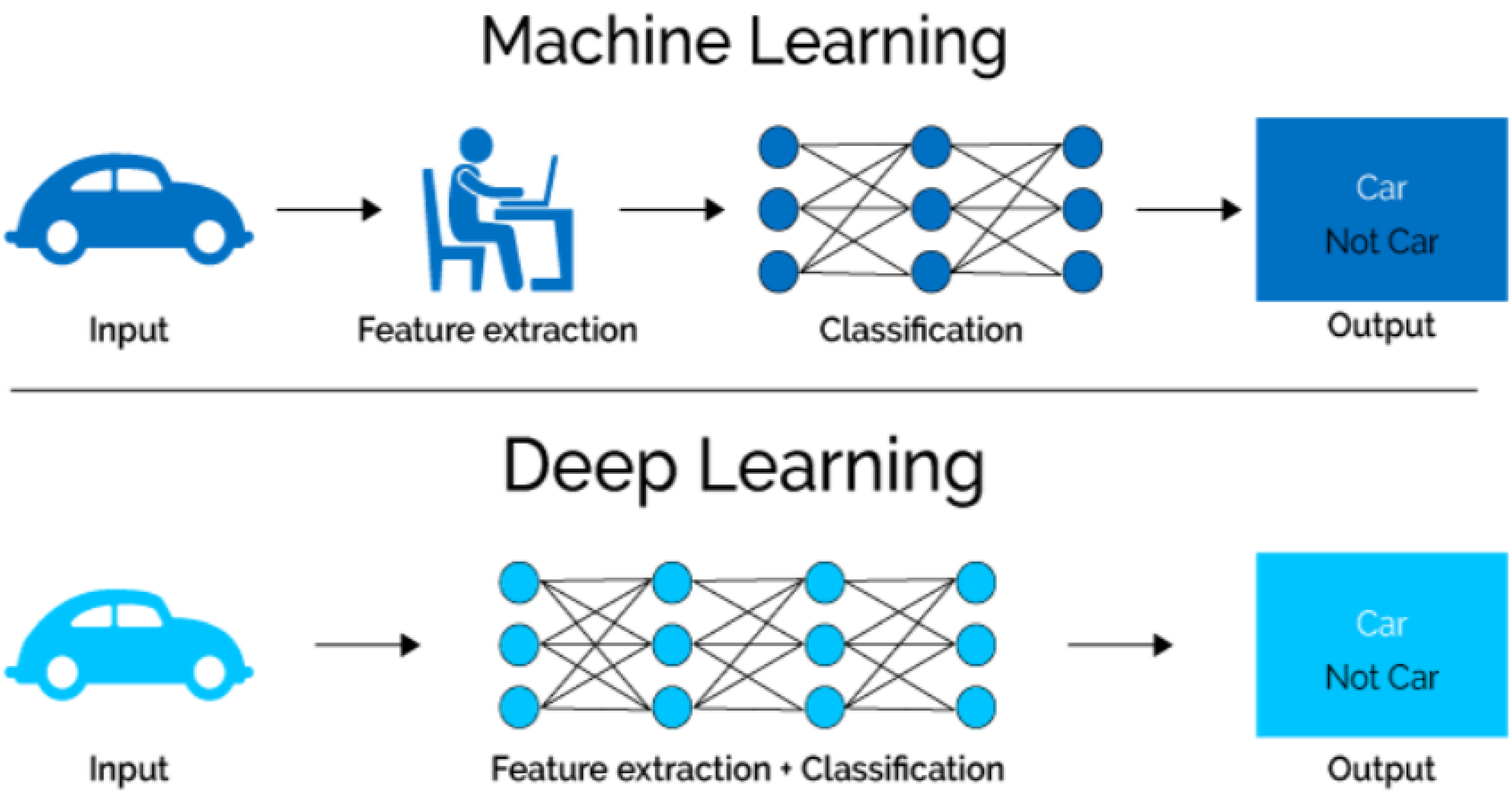
I hope this helps! Let me know if you have any questions.

What is Deep Learning?



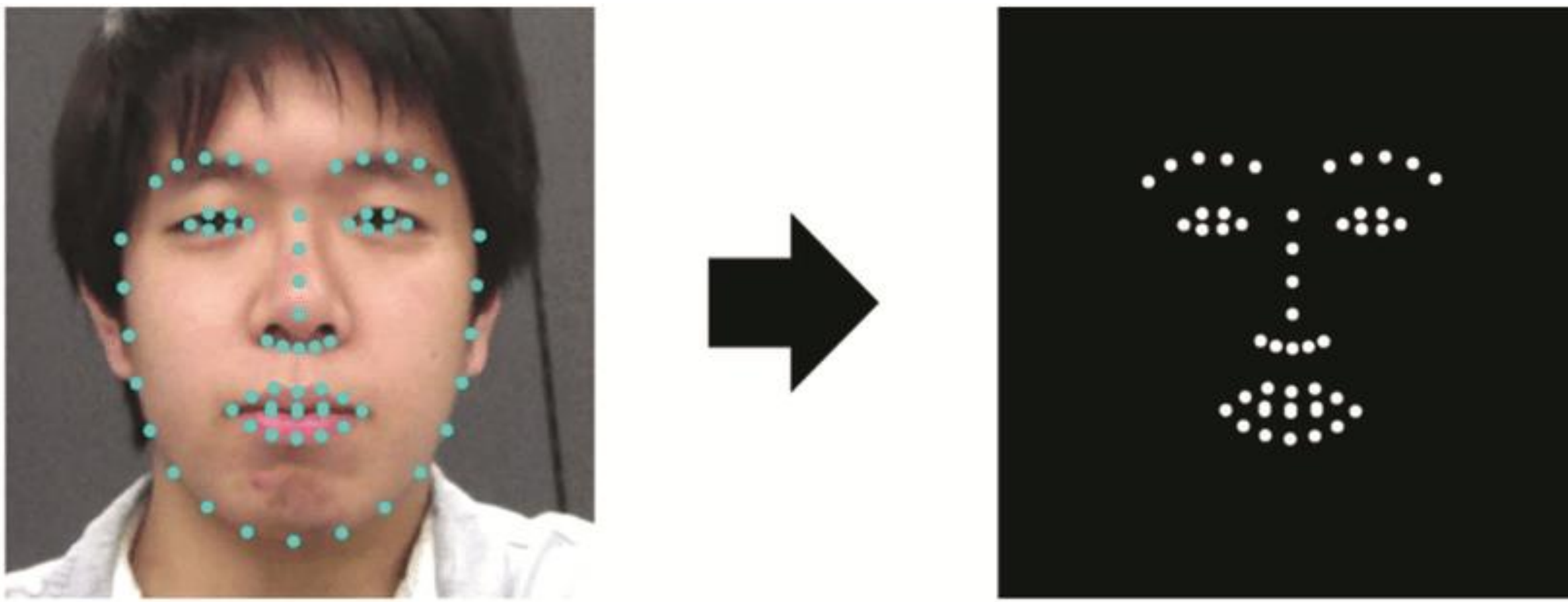
Teaching computers how to **learn a task** directly from **raw data**

Machine learning vs Deep Learning



Traditional Approach

Use hand-engineered features



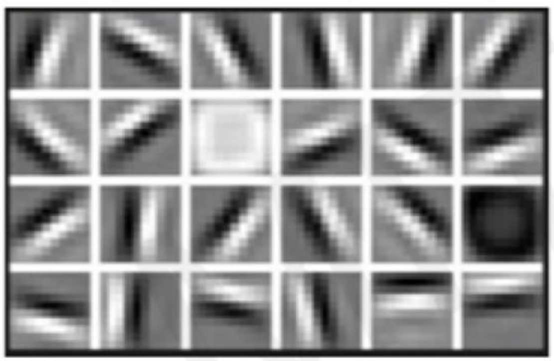
(a) Detected facial keypoints (b) Facial organ keypoint

Why Deep Learning?

Hand engineered features are time consuming, brittle, and not scalable in practice

Can we learn the **underlying features** directly from data?

Low Level Features



Lines & Edges

Mid Level Features

Eyes & Nose & Ears

High Level Features

Facial Structure

Why Now?

Neural Networks date back decades, so why the resurgence?

1952 Stochastic Gradient Descent

1958 Perceptron
- Learnable Weights

⋮

1986 Backpropagation
- Multi-Layer Perceptron

1995 Deep Convolutional NN
- Digit Recognition

⋮

1. Big Data
- Larger Datasets
- Easier Collection & Storage

IMAGENET

WIKIPEDIA
The Free Encyclopedia

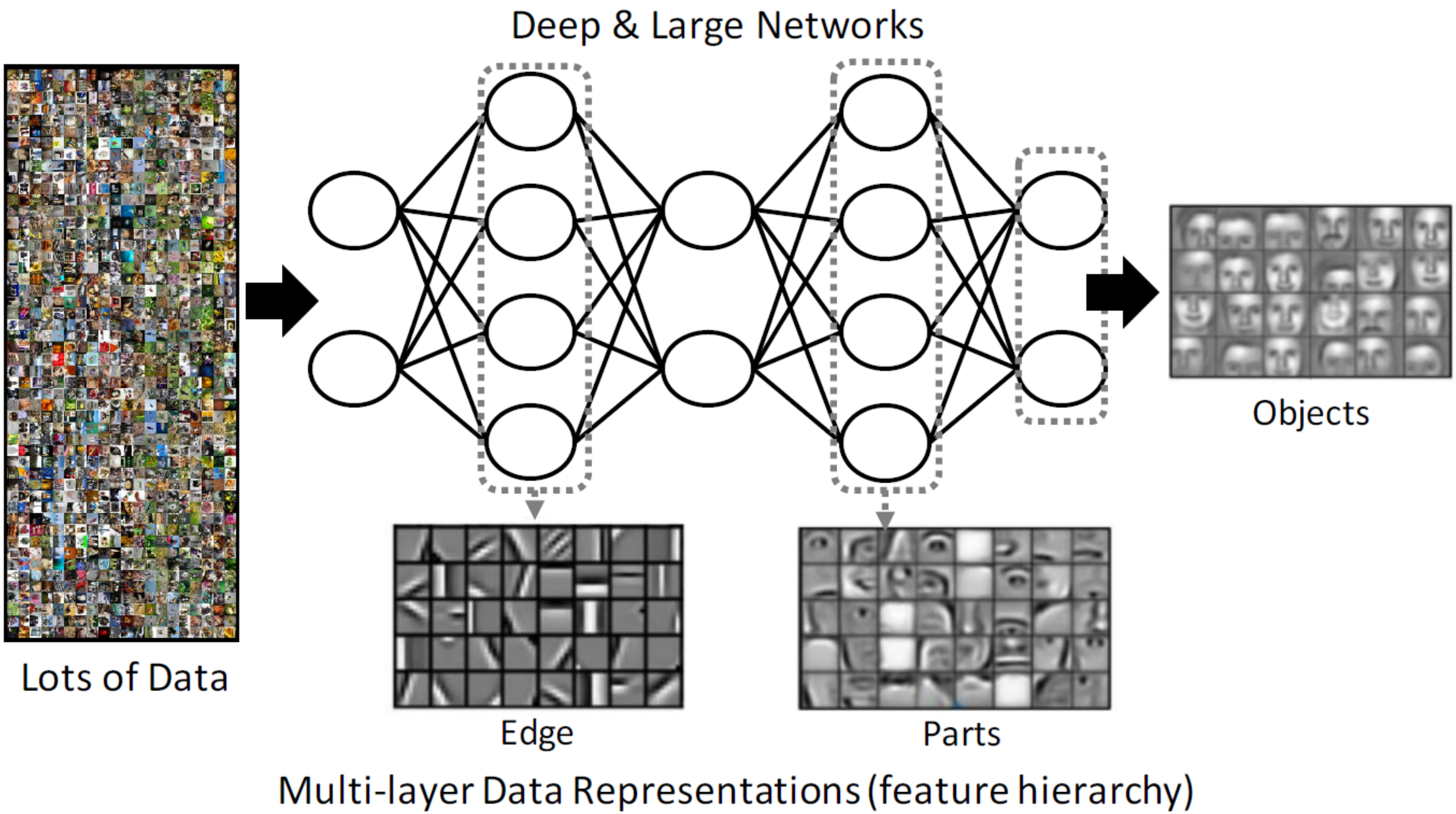
2. Hardware
- Graphics Processing Units (GPUs)
- Massively Parallelizable

3. Software
- Improved Techniques
- New Models
- Toolboxes

TensorFlow

Definition of Deep Learning

- An algorithm that learns **multiple levels** of abstractions in data



Deep Computer Vision

Our visual system is trained on images seen in 540 mln of years!

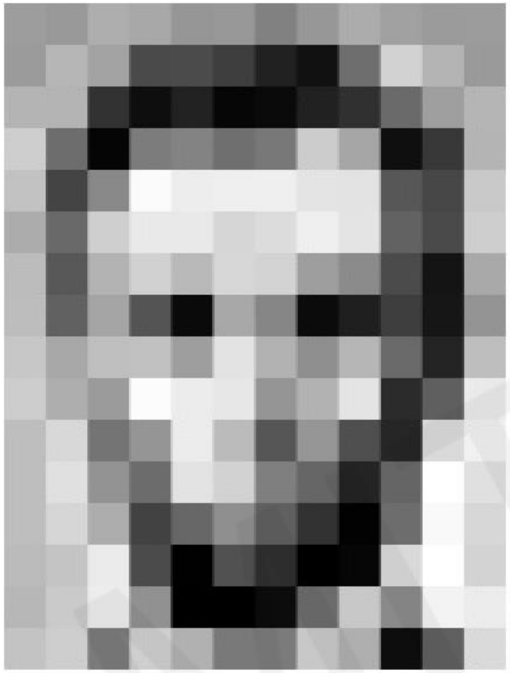
Images are Numbers

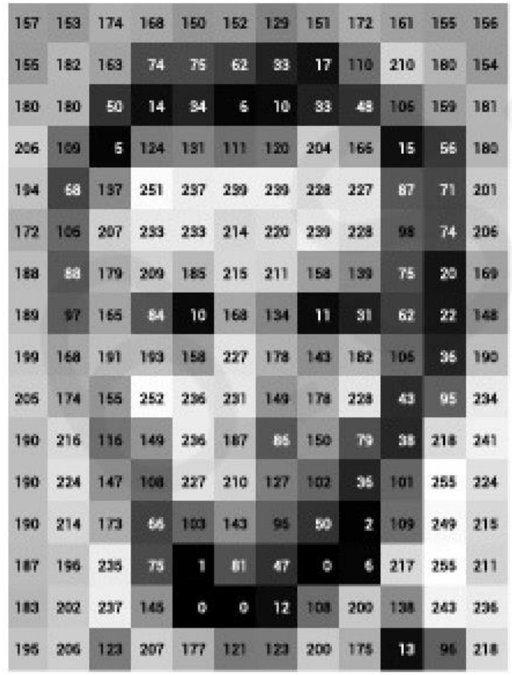
What I see

What a computer sees

08	02	22	97	38	15	00	40	00	75	04	05	07	78	52	12	50	77	91	08
49	49	99	40	17	81	18	57	60	87	17	40	98	43	69	48	04	56	62	00
81	49	31	73	55	79	14	29	93	71	40	67	53	88	30	03	49	13	36	65
52	70	95	23	04	60	11	42	69	24	68	56	01	32	56	71	37	02	36	91
22	31	16	71	51	67	63	89	41	92	36	54	22	40	40	28	66	33	13	80
24	47	32	60	99	03	45	02	44	75	33	53	78	36	84	20	35	17	12	50
32	98	81	28	64	23	67	10	26	38	40	67	59	54	70	66	18	38	64	70
67	26	20	68	02	62	12	20	95	63	94	39	63	08	40	91	66	49	94	21
24	55	58	05	66	73	99	26	97	17	78	78	96	83	14	88	34	89	63	72
21	36	23	09	75	00	76	44	20	45	35	14	00	61	33	97	34	31	33	95
78	17	53	28	22	75	31	67	15	94	03	80	04	62	16	14	09	53	56	92
16	39	05	42	96	35	31	47	55	58	88	24	00	17	54	24	36	29	85	57
86	56	00	48	35	71	89	07	05	44	44	37	44	60	21	58	51	54	17	58
19	80	81	68	05	94	47	69	28	73	92	13	86	52	17	77	04	89	55	40
04	52	08	83	97	35	99	16	07	97	57	32	16	26	26	79	33	27	98	66
88	36	68	87	57	62	20	72	03	46	33	67	46	55	12	32	63	93	53	69
04	42	16	73	38	25	39	11	24	94	72	18	08	46	29	32	40	62	76	36
20	69	36	41	72	30	23	88	34	62	99	69	82	67	59	85	74	04	36	16
20	73	35	29	78	31	90	01	74	31	49	71	48	86	81	16	23	57	05	54
01	70	54	71	83	51	54	69	16	92	33	48	61	43	52	01	89	19	67	48

Images are Numbers





What the computer sees

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

An image is just a matrix of numbers [0,255]!
i.e., 1080x1080x3 for an RGB image

Color image: RGB 3 channels

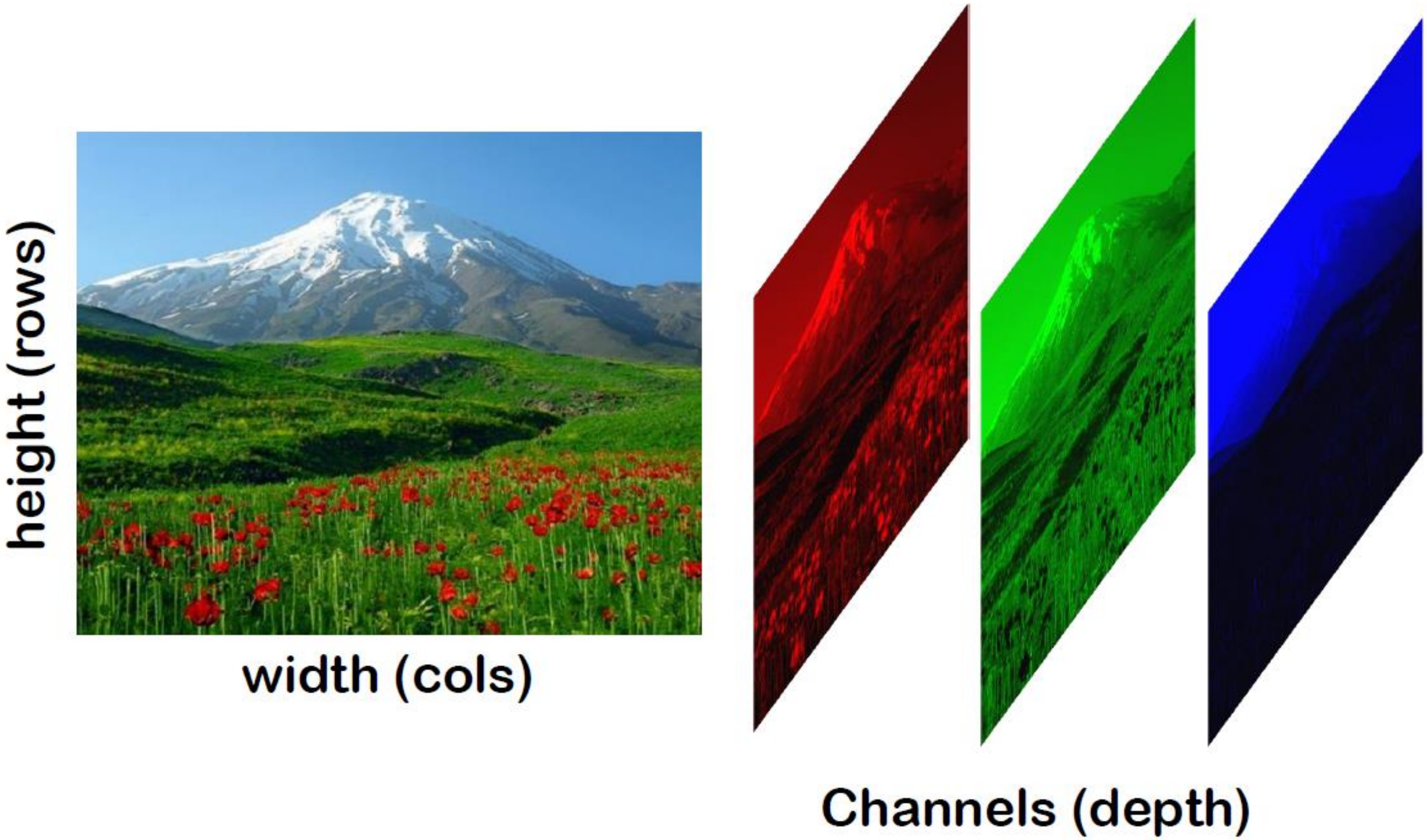
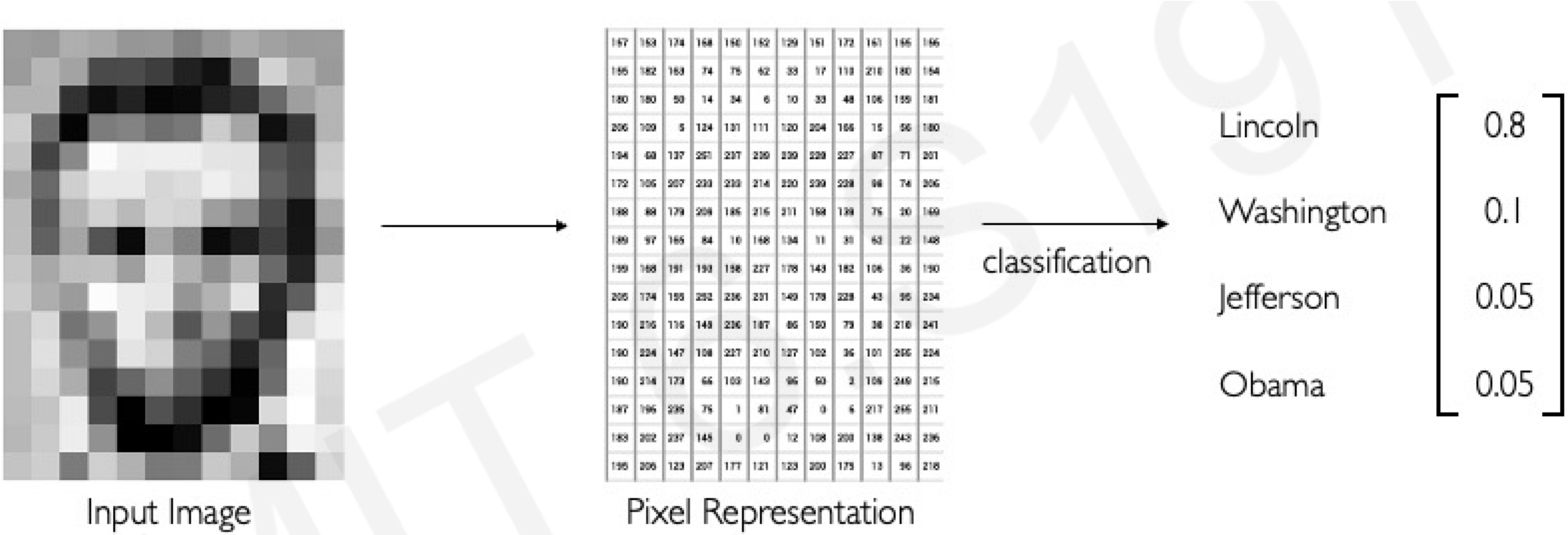


Image Classification task

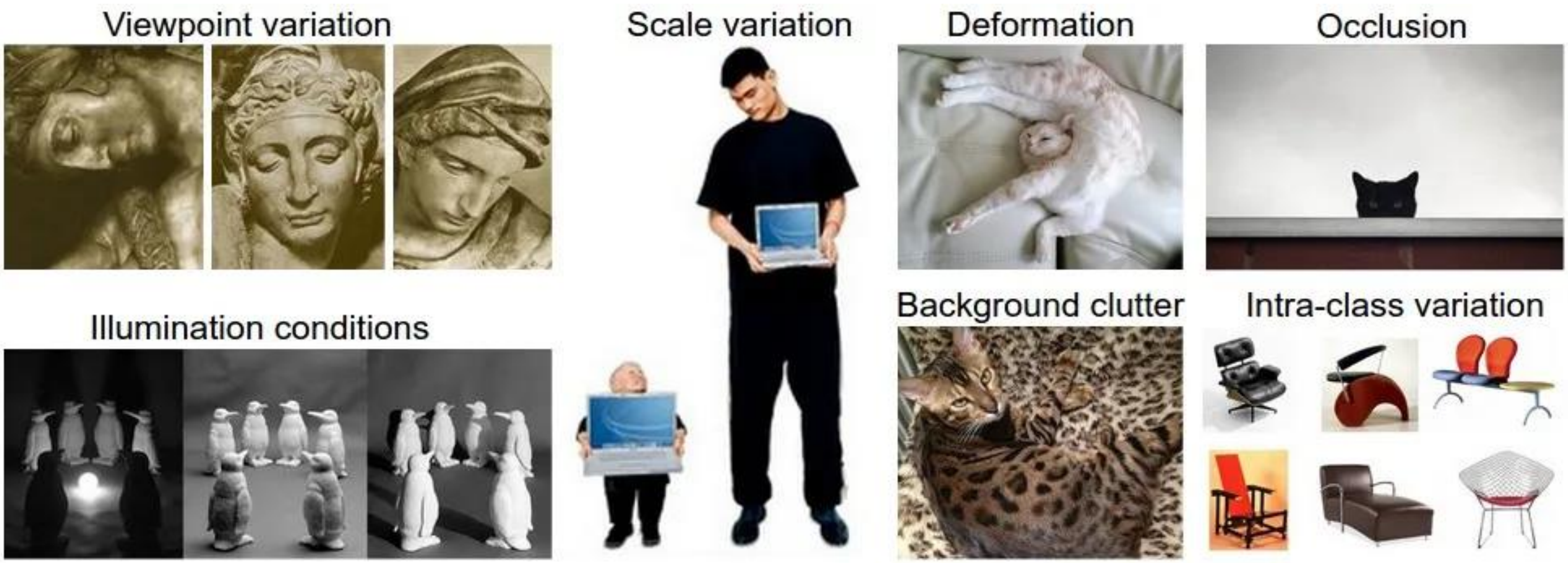


High-level Feature Detection

Let's identify key features in each image category

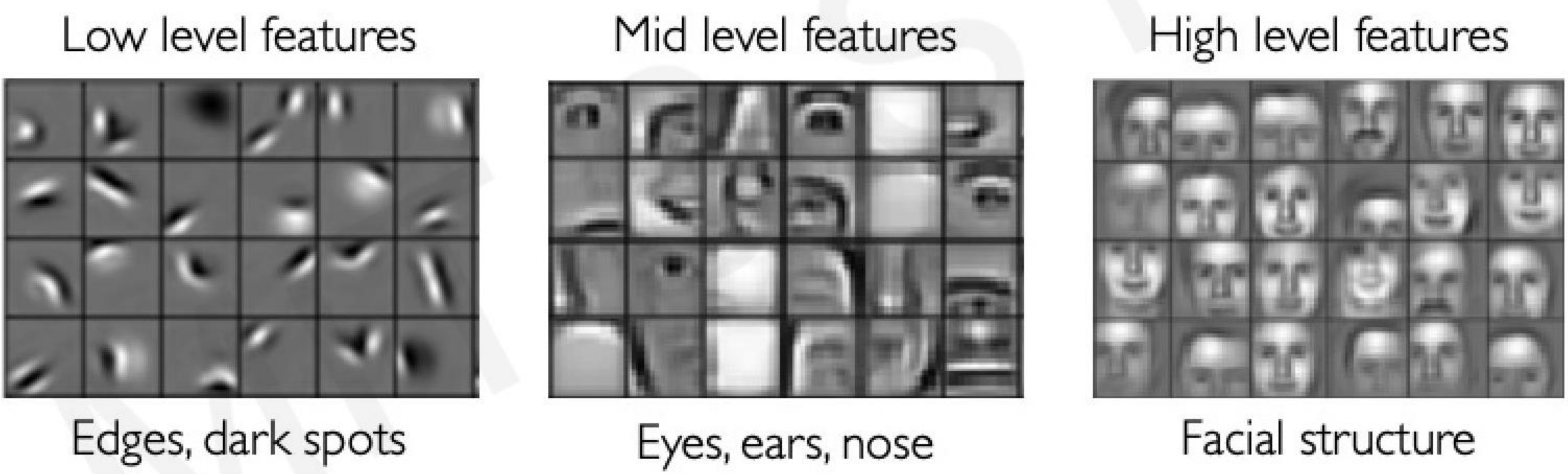
Nose, Eyes, Mouth	Wheels, License Plate, Headlights	Door, Windows, Steps

Manual Feature Extraction (challenges)

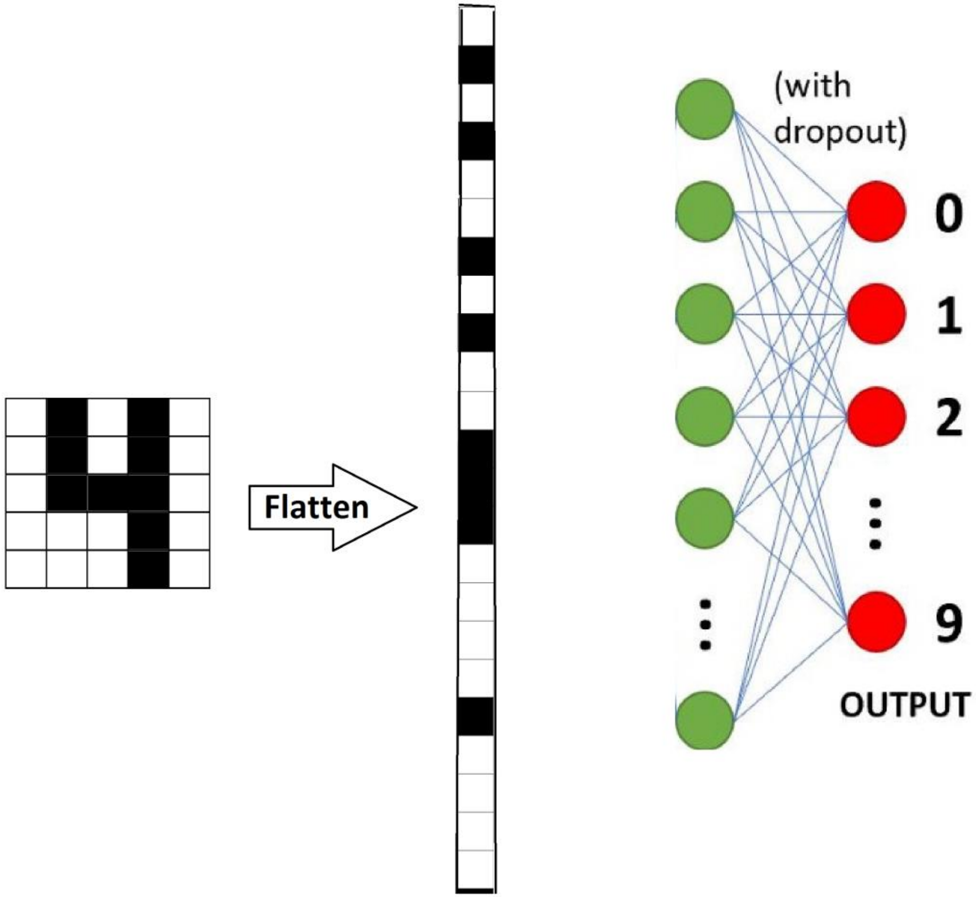


Learning Feature Representations

Can we learn a **hierarchy of features** directly from data instead of hand engineering?



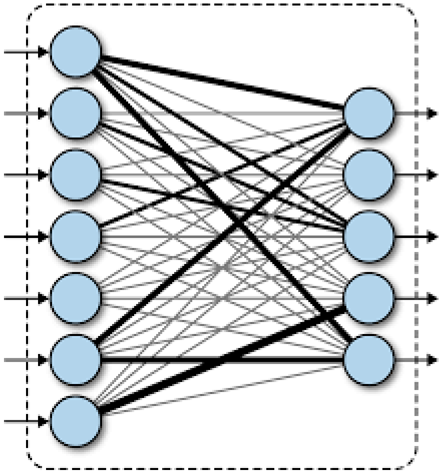
Traditional Method



Traditional Method

Previous DNNs use fully-connected layers

Connect **all** the neurons between the layers



Drawbacks:

- (-) **Large number of parameters**
- Easy to be over-fitted
- Large memory consumption

- (-) Does not enforce any structure, e.g., **No Spatial information**
- In many applications, local features are important, e.g., images

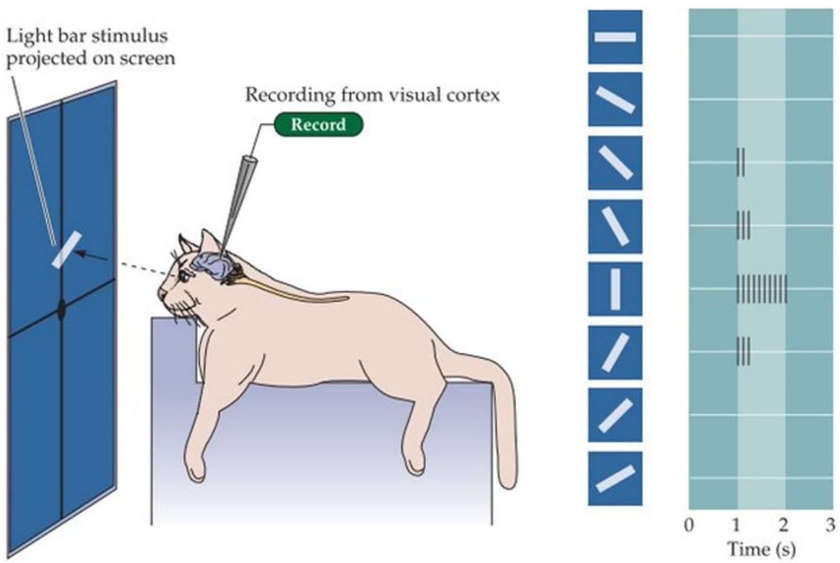
Convolutional Neural Networks (CNN)

Hubel and Wiesel's experiment

Hubel and Wiesel's experiments on cats' visual cortex influenced the intuition behind CNN models.

• They discovered that certain neurons in the visual cortex were sensitive to edges and lines.

• Different neurons responded to specific orientations of edges, regardless of their position in the visual field.

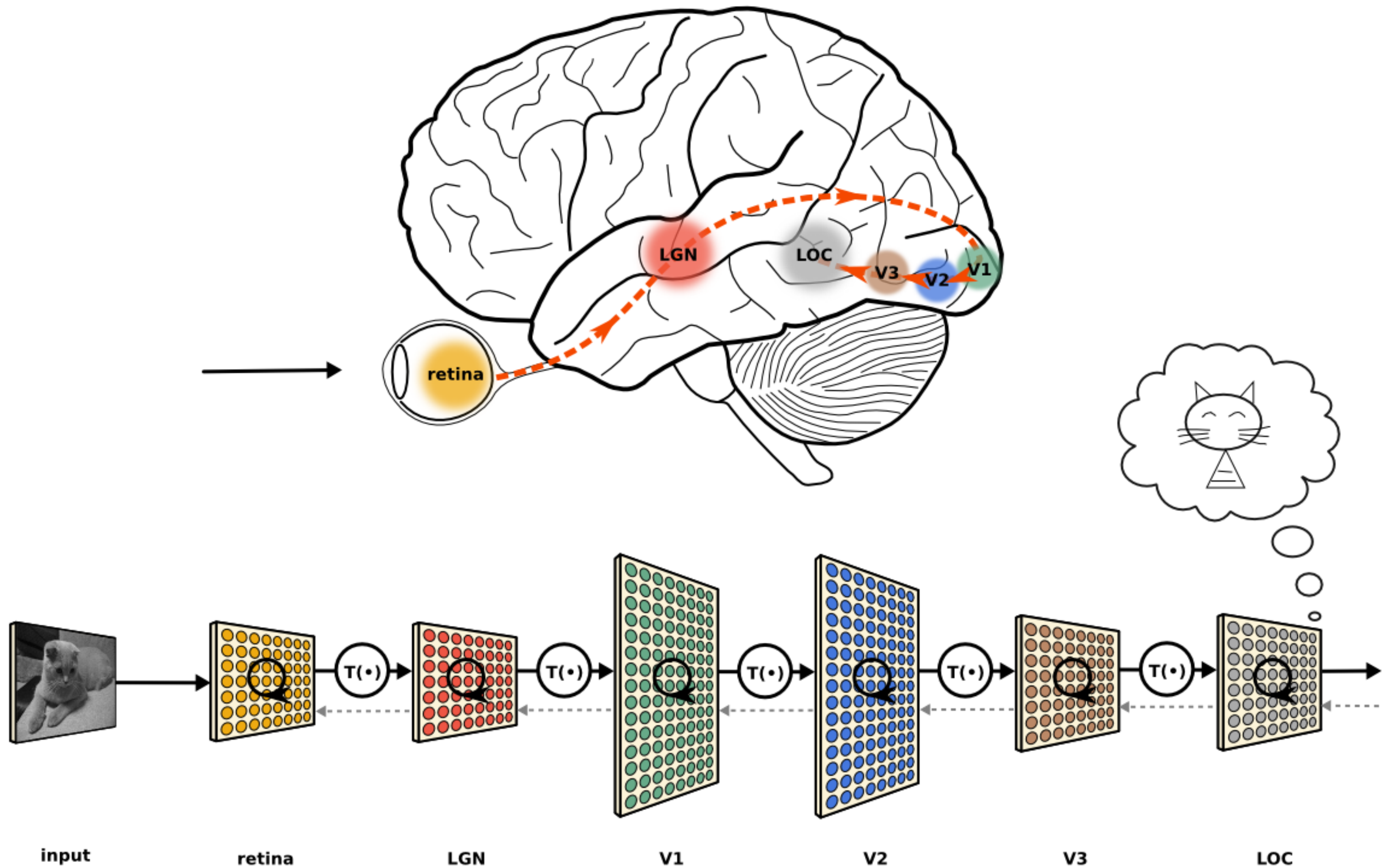


Hubel and Wiesel's experiment

- Their ground-breaking research led to the discovery of specialized cells in the visual cortex called "**simple cells**" and "**complex cells**."
- Simple cells responded selectively to specific orientations of lines or edges.
- Complex cells responded to more complex visual stimuli, such as moving lines or gratings

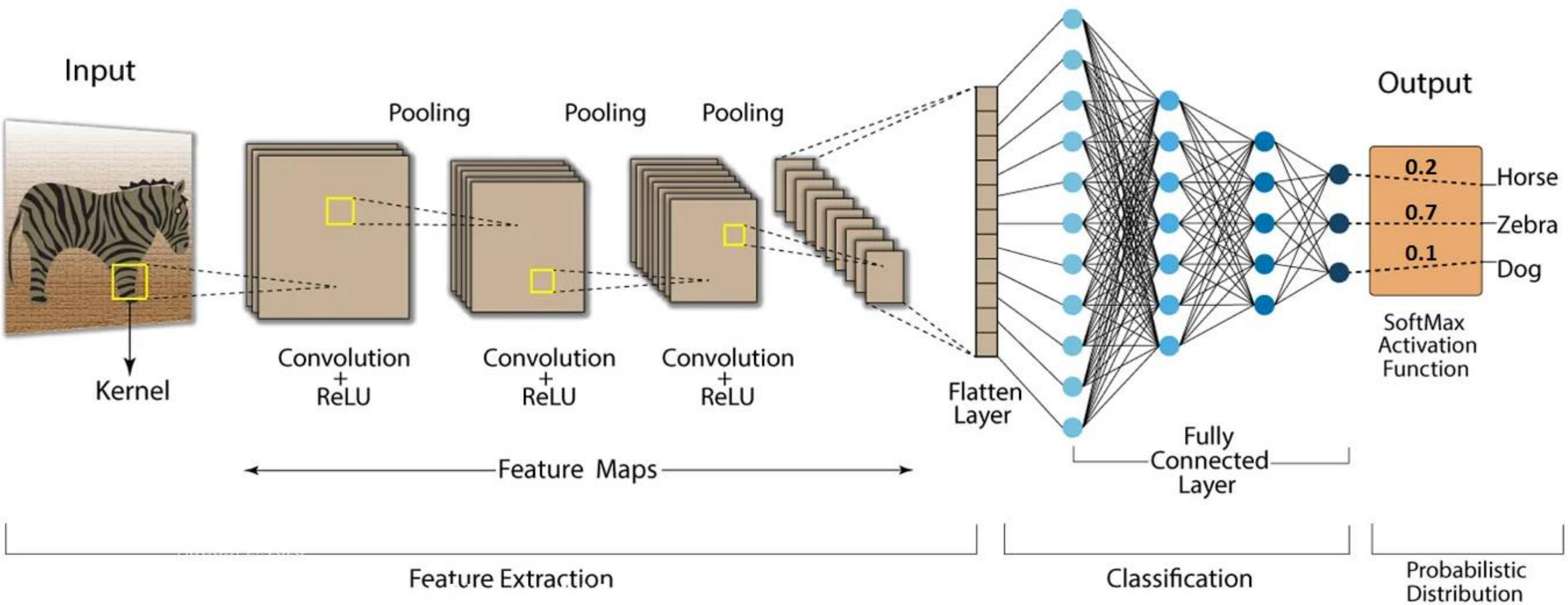
These findings led to the development of CNNs, which mimic the hierarchical processing of visual information in the brain.

Hierarchical visual processing in the brain



Convolutional Neural Networks (CNN)

- Convolutional layer
- Pooling layer
- Fully-connected (FC) layer

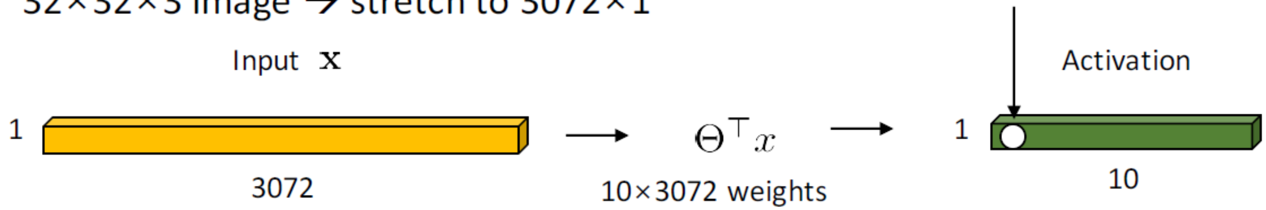


Convolutional layer

Fully-connected layer

- 32×32×3 image → stretch to 3072×1

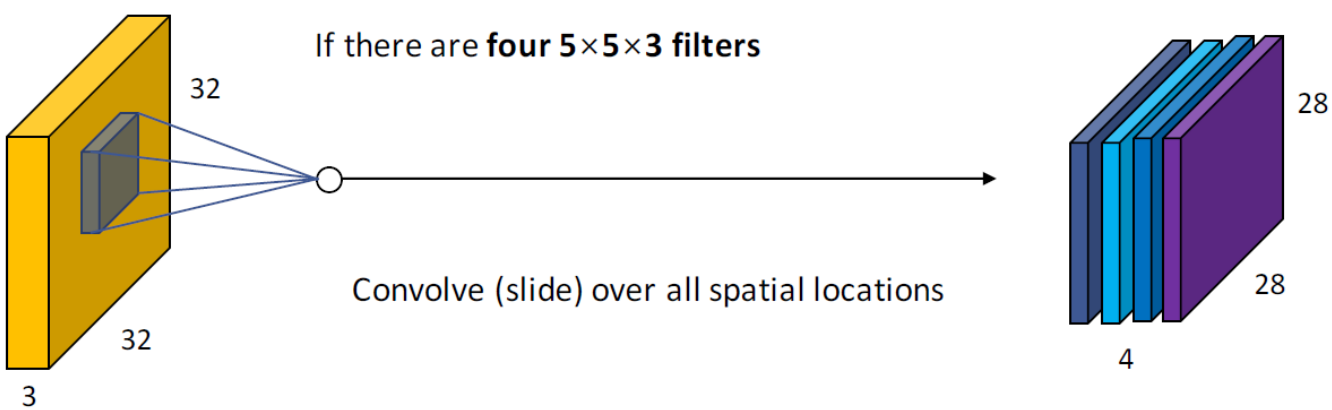
The result of taking a dot product between a row of $\Theta^\top$ and the input



Convolution layer

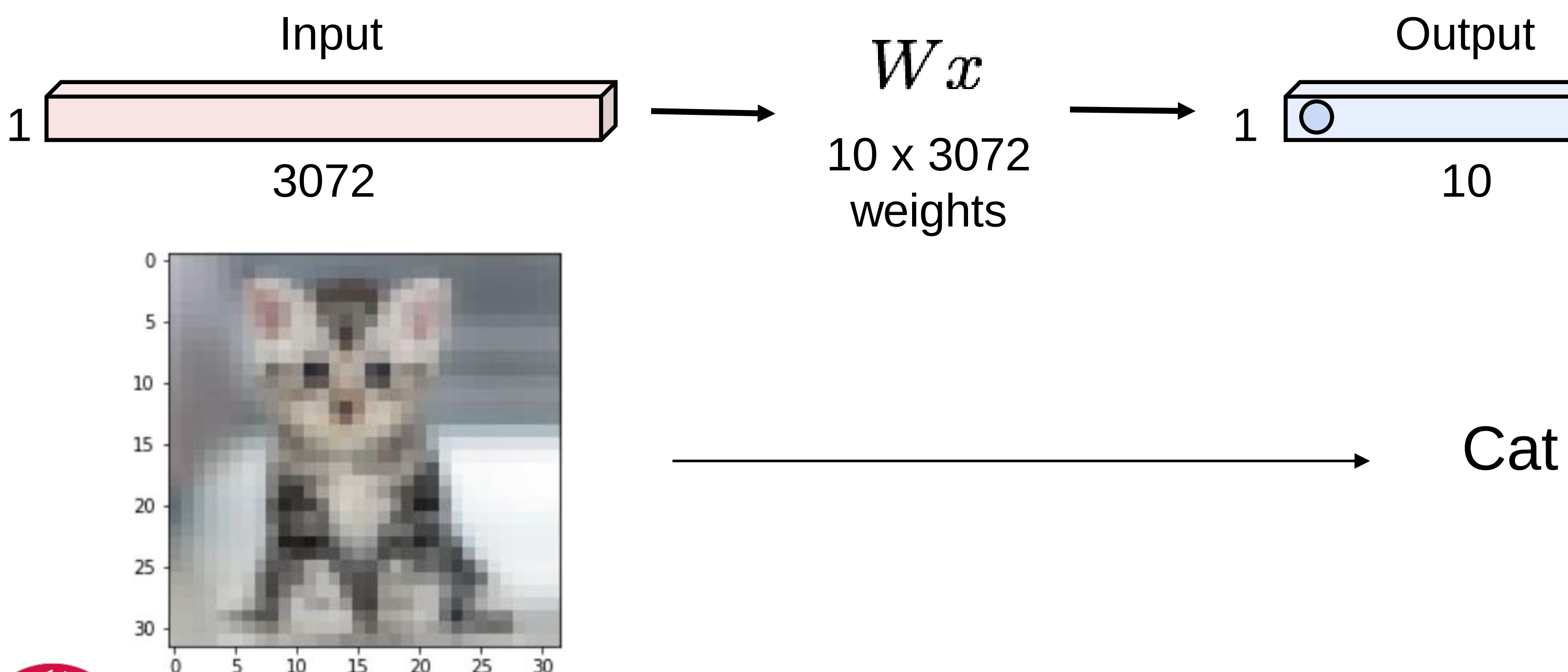
32×32×3 image

4 separate activation maps

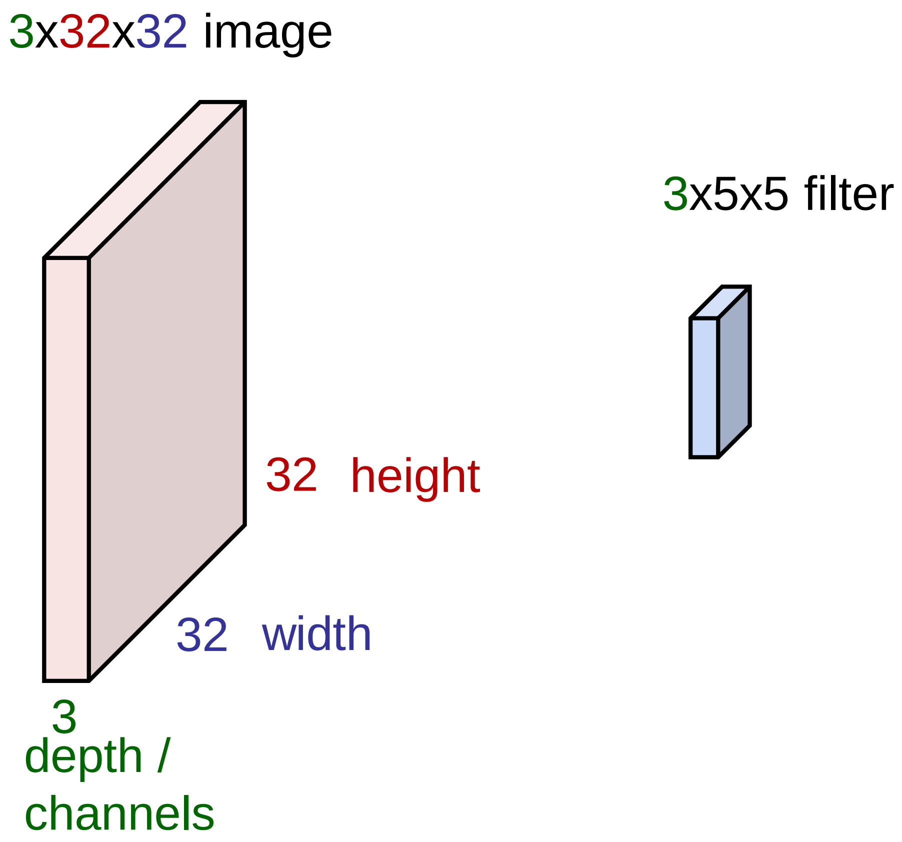


Traditional Method

32x32x3 image -> stretch to 3072 x 1



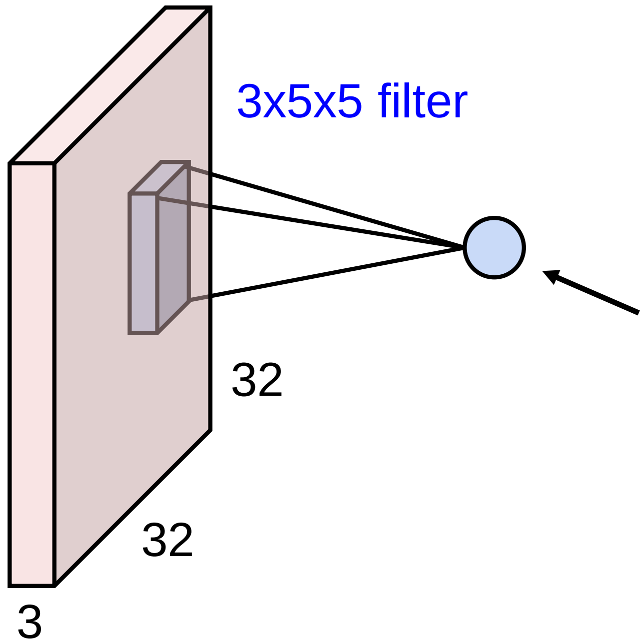
Convolutional layer



Convolve the filter with the image i.e. "slide over the image spatially, computing dot products"

Convolutional layer

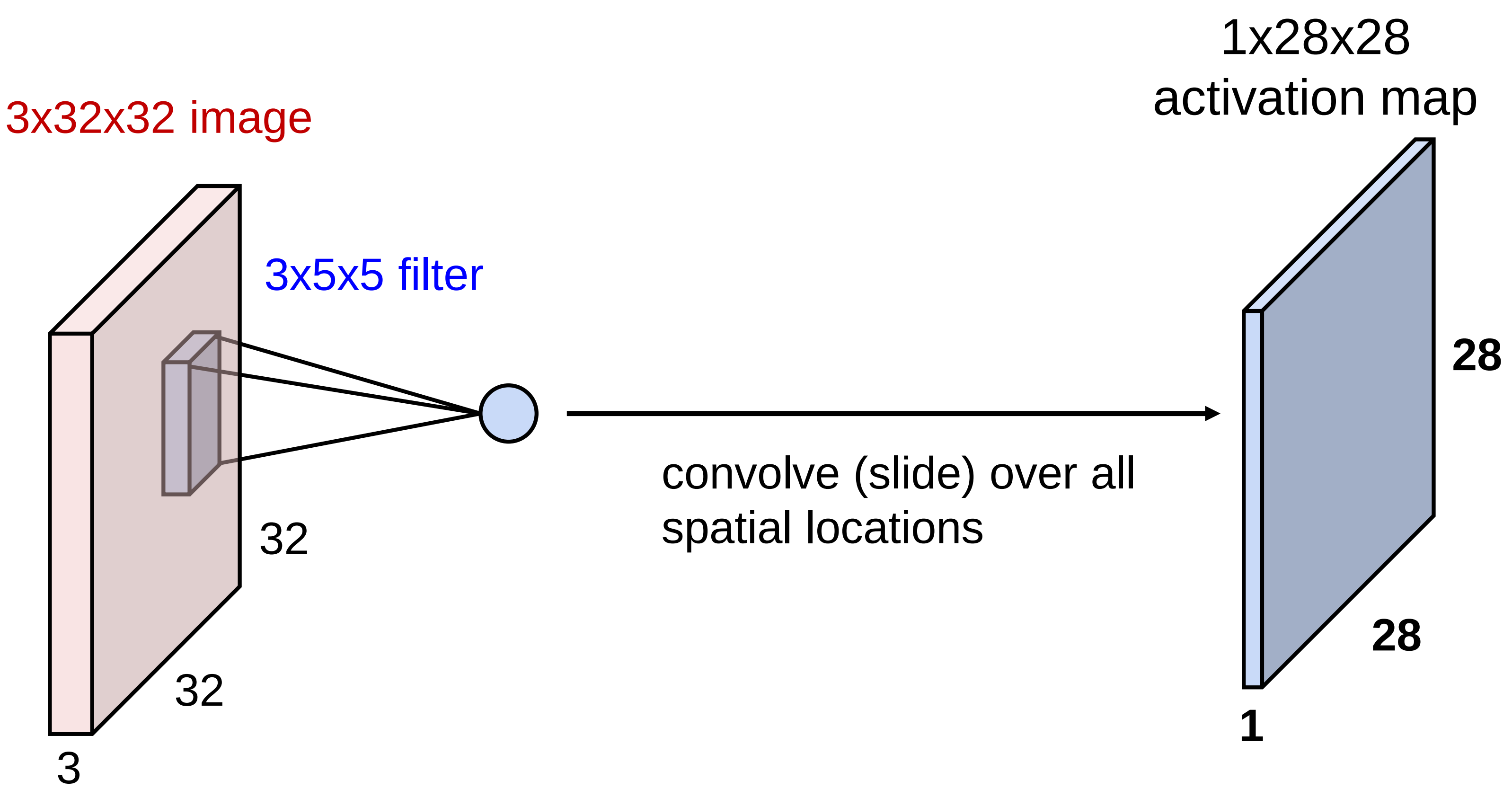
3x32x32 image



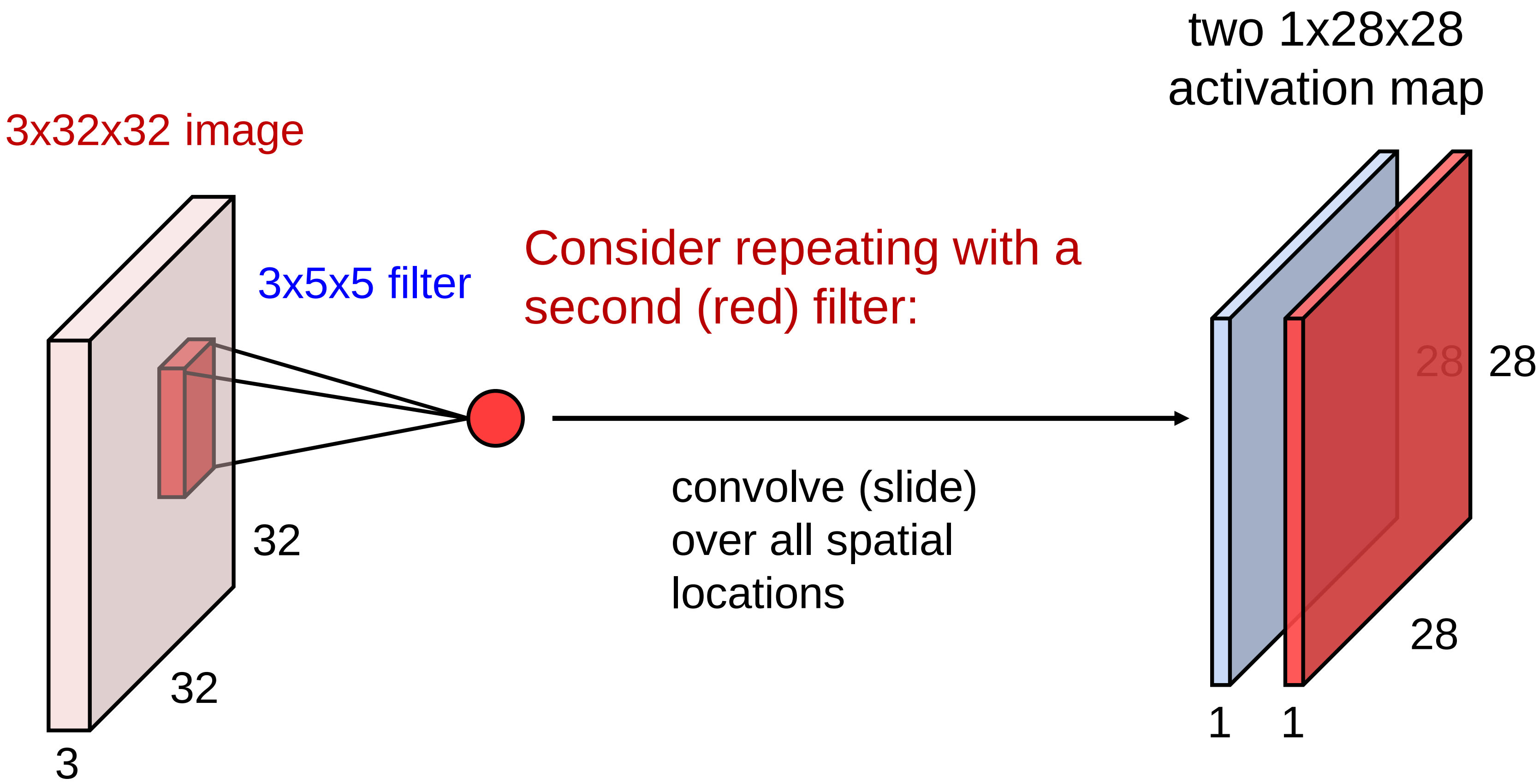
1 number :
the result of taking a dot product between the filter and a small 3x5x5 chunk of the image

$$w^T x + b$$

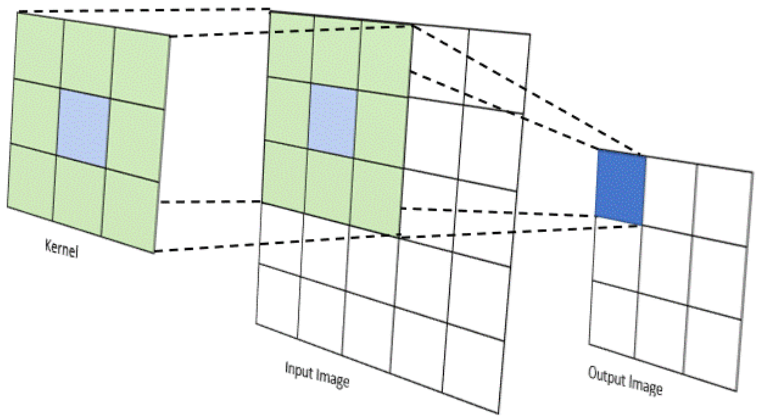
Convolution Layer



Convolutional layer



Convolutional layer



Input

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

x

Filter / Kernel

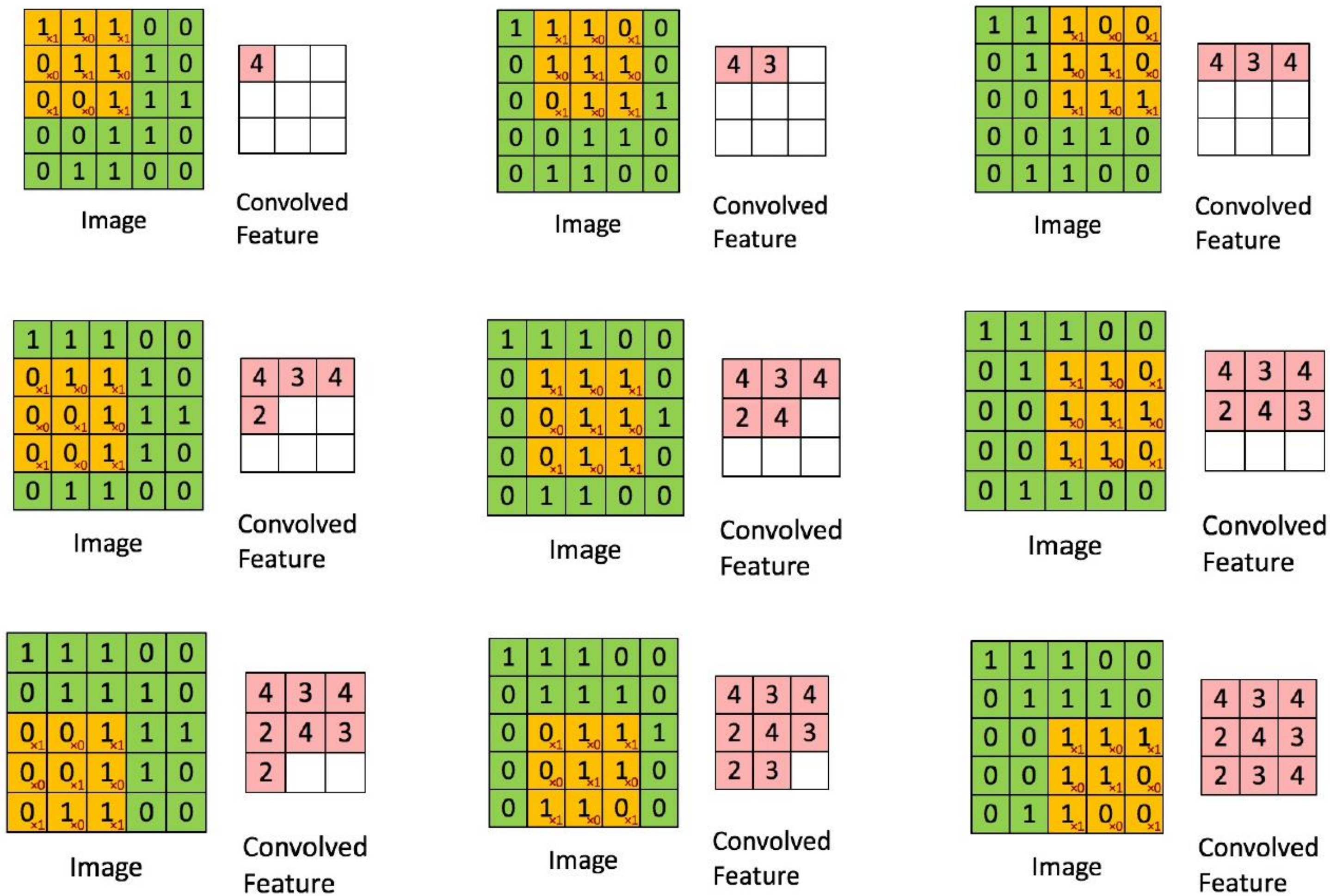
1	0	1
0	1	0
1	0	1

=

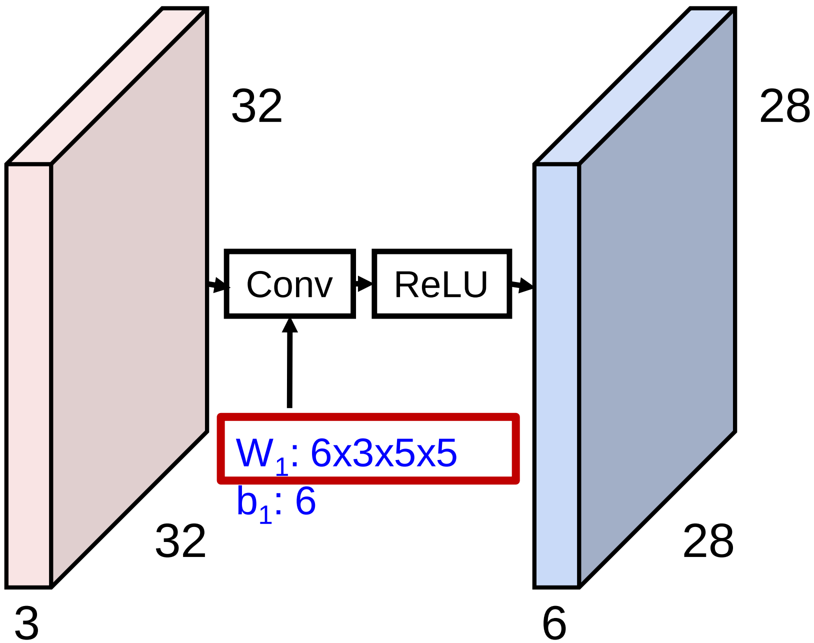
Feature map

4		

Convolution Operation



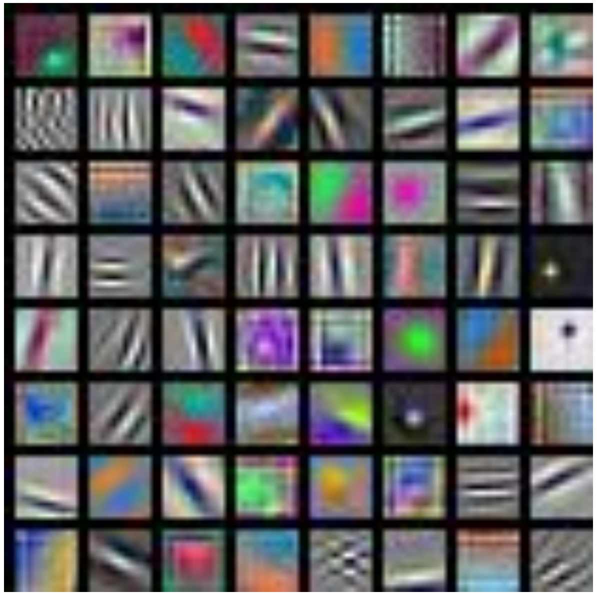
What do convolutional filters learn?



Input:
N x 3 x 32 x 32

First hidden layer:
N x 6 x 28 x 28

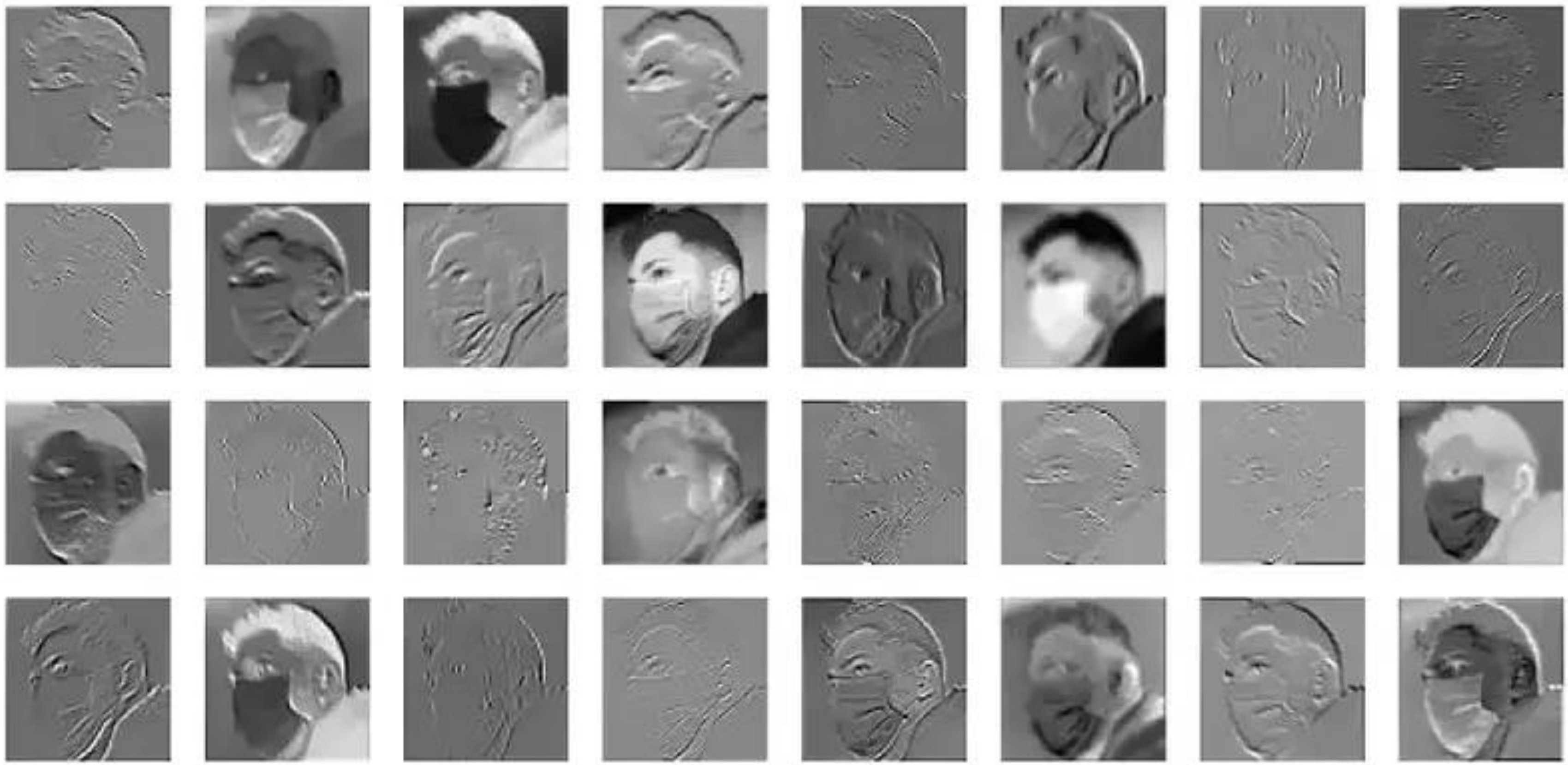
First-layer conv filters: local image templates
(Often learns oriented edges, opposing colors)



AlexNet: 64 filters, each 3x11x11

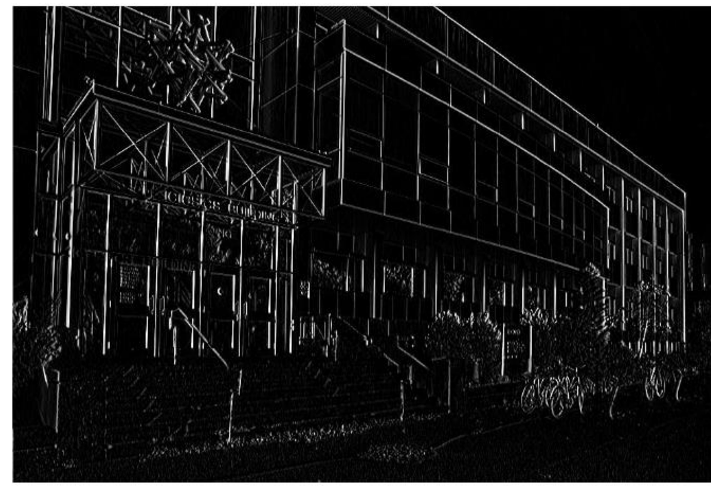
Convolution filters

Convolution of an image with different filters can perform operations such as edge detection, blur and sharpen by applying filters.



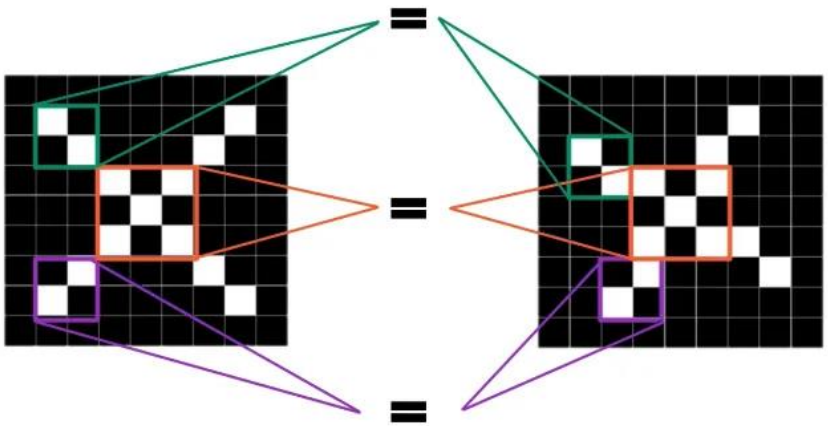
Convolution filters

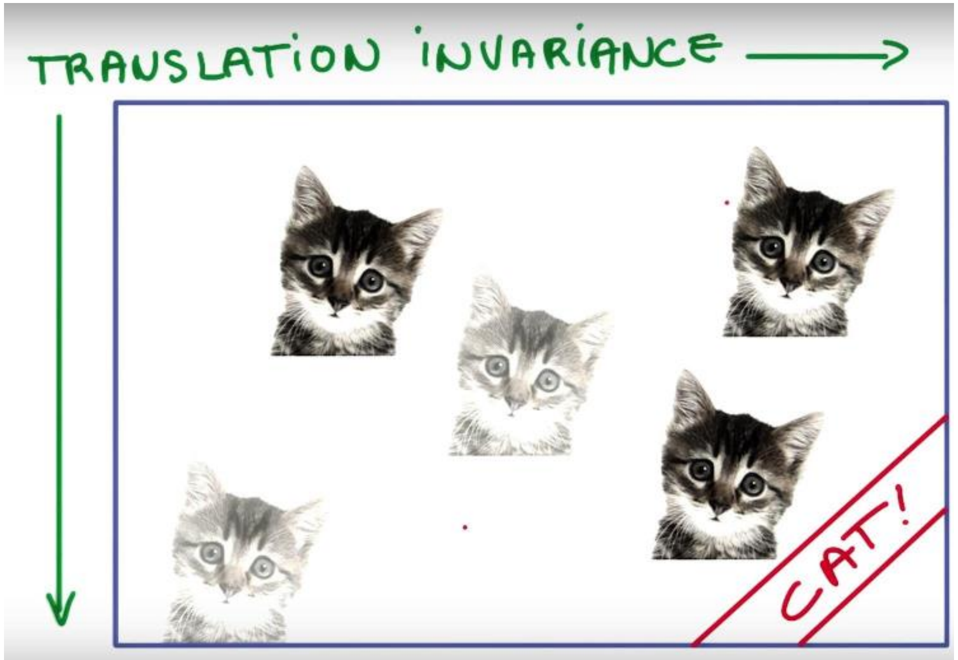
$$\star \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \longrightarrow$$



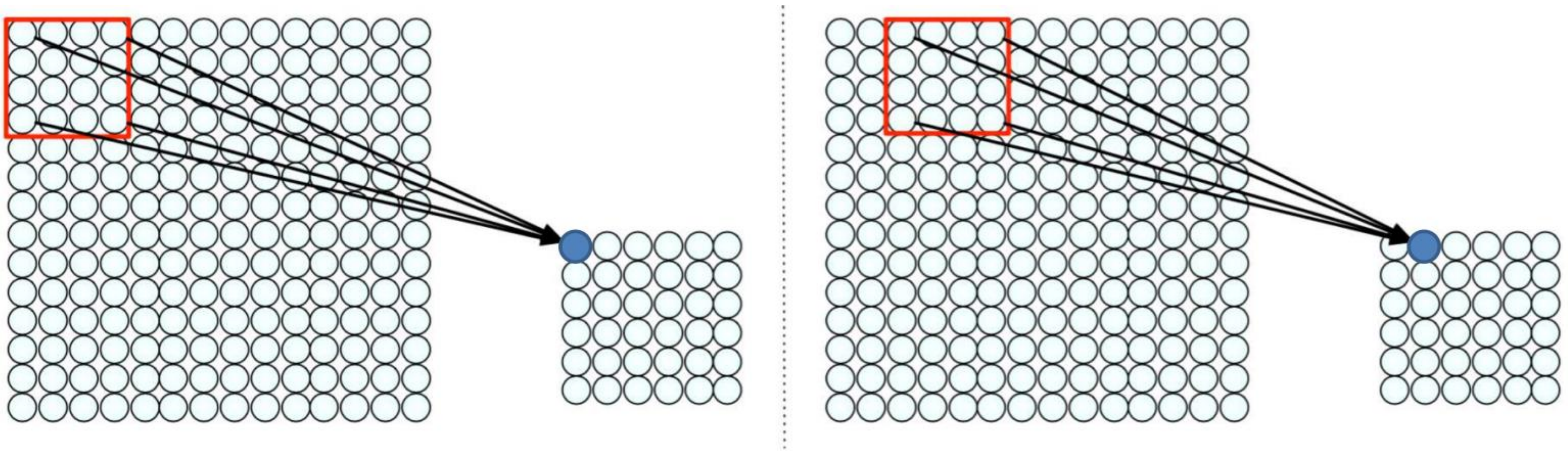
Translation invariance

When input is changed spatially (translated or shifted), the corresponding output to recognize the object should not be changed





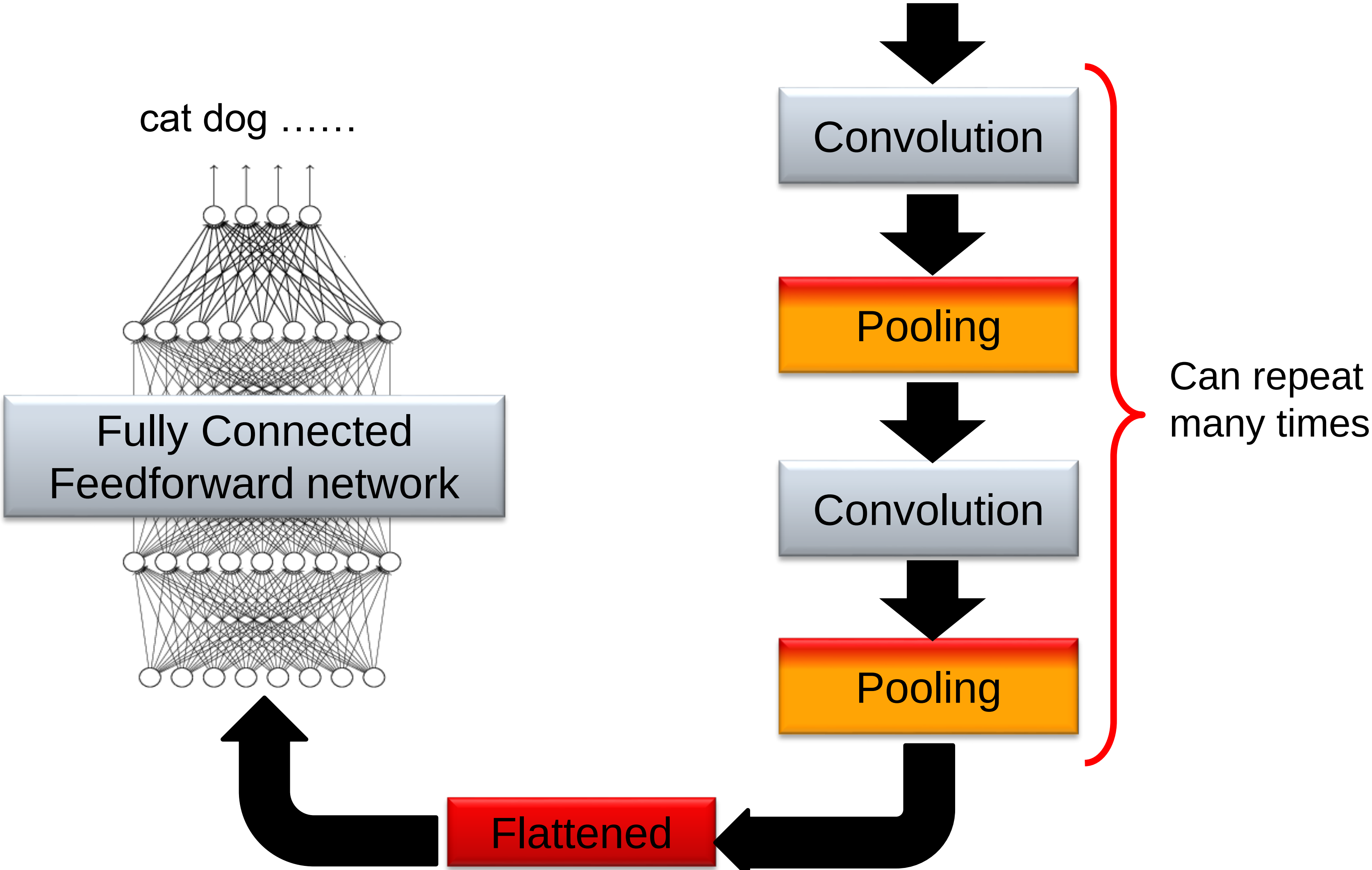
Using Spatial Structure/Information



2) Slide the patch window across the image.

Different **weights (filters)** *detect different features*

The whole CNN



CNN – Main components

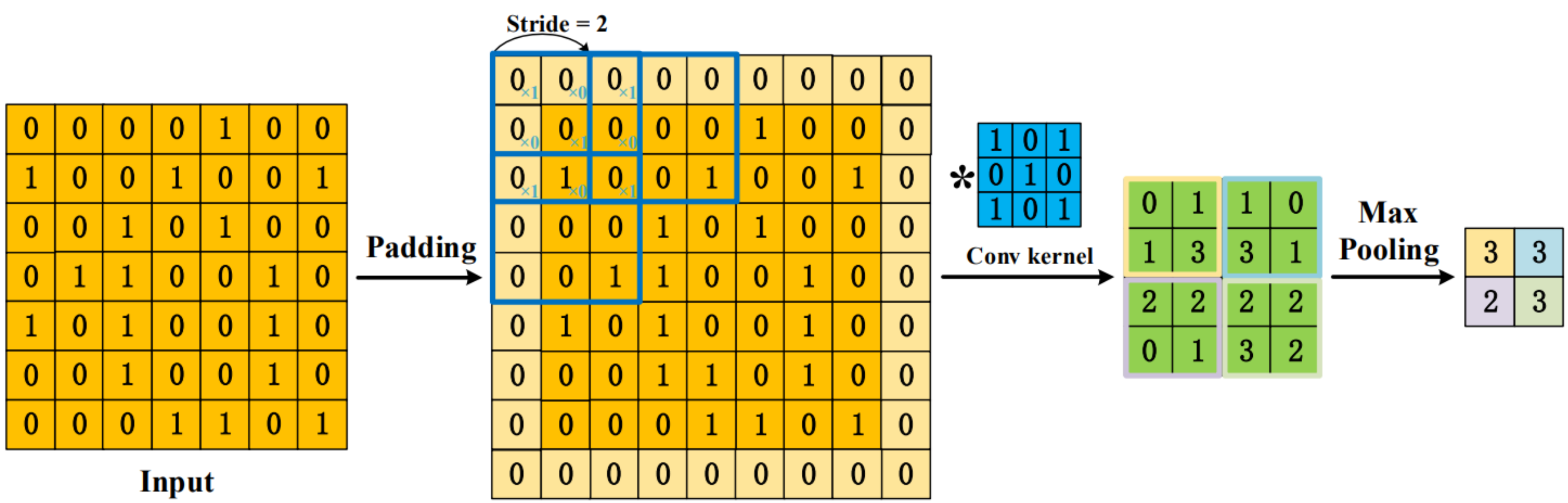
1. To build a CNN model, four components are typically needed (Li et al. 2020).

   Convolution The outputs of convolution can be called feature maps.

   Padding Padding enlarges the input with zero value.

   Stride For controlling the density of convolving, stride used.

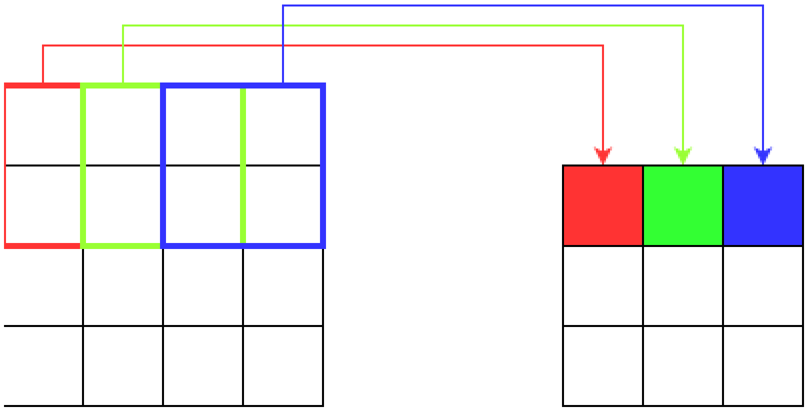
   Pooling As a result, Pooling (down-sampling) such as max pooling and average pooling obviates large number of features in feature map.



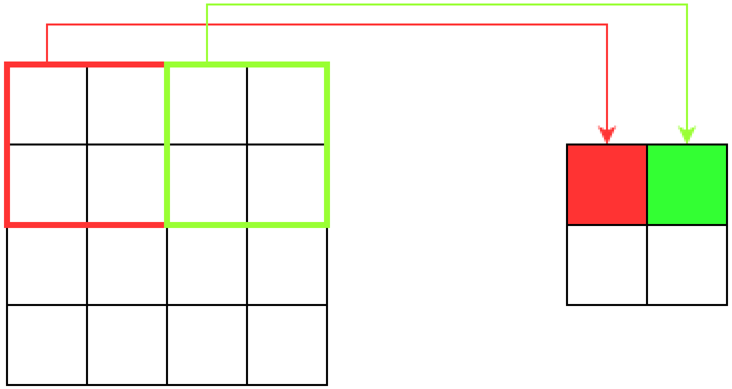
Strides

- Stride is the number of pixels shifts over the input matrix.
- When the stride is 1 then we move the filters to 1 pixel at a time.
- When the stride is 2 then we move the filters to 2 pixels at a time and so on.

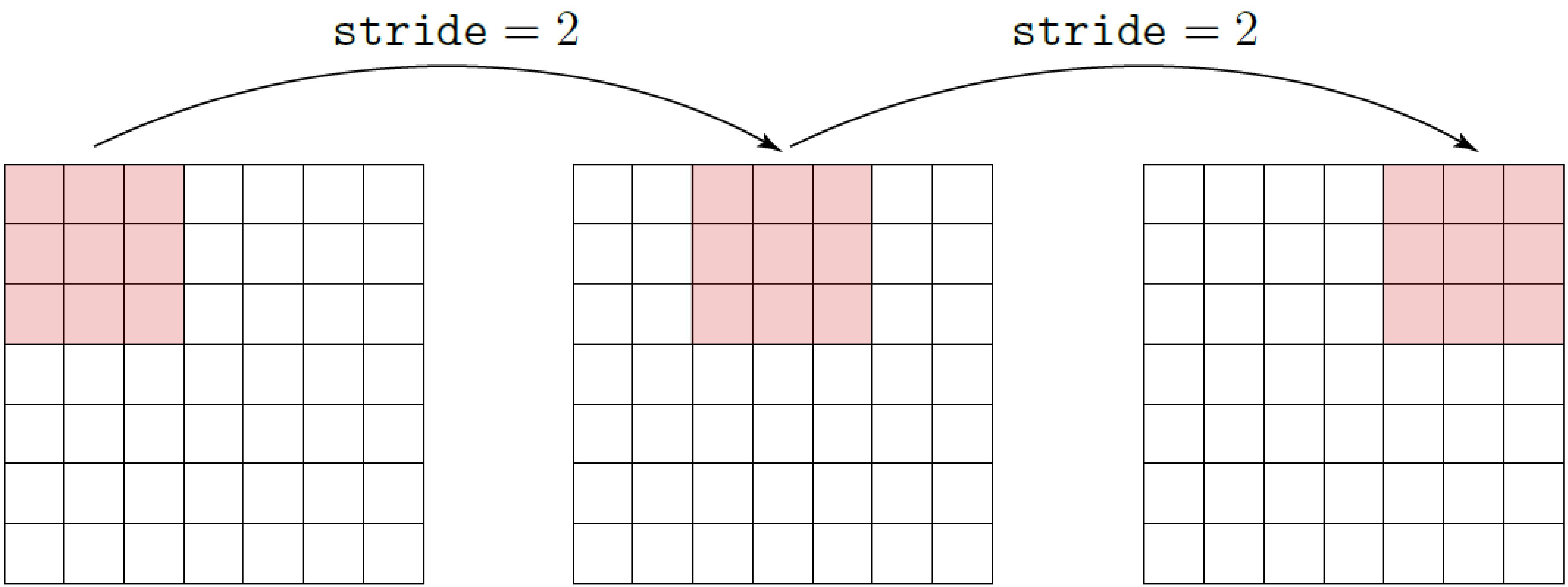
Stride = 1



Stride = 2



Strides



Padding

- Sometimes filter does not fit perfectly fit the input image. We have two options:

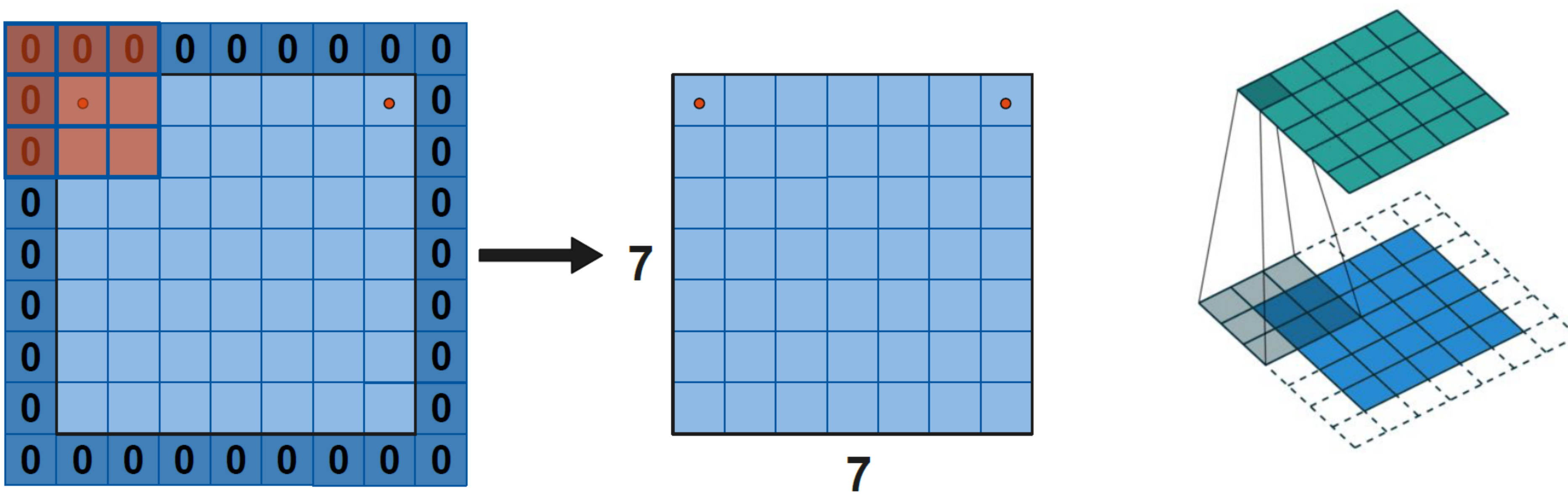
- Pad the picture with zeros (zero-padding) so that it fits

- Drop the part of the image where the filter did not fit. This is called valid padding which keeps only valid part of the image.

Padding

Add zeros around image borders to conserve the spatial extent of the input.
Prevents fast shrinking of the input data (image)

- **Example:** Convolution with 3 x 3 filter and padding



Padding

- If you have a stride of 1 and if you set the size of zero padding to
- $$Zero\ Padding = \frac{(K-1)}{2}$$
- where K is the filter size, then the input and output volume will always have the same spatial dimensions.
- The formula for calculating the output size for any given conv layer is
- $$O = \frac{(W - K + 2P)}{S} + 1$$
- where O is the output height/length, W is the input height/length, K is the filter size, P is the padding, and S is the stride.

Padding

- In practice: Common to **zero pad** the border
  - Used to control the output filter size

9

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

9

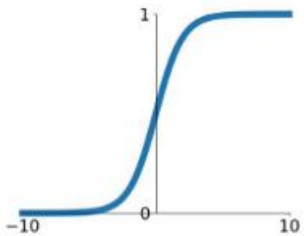
7×7 input (spatially)
Zero pad 1 pixel border
Assume 3×3 filter
Applied with **stride 3**

→ **3×3 output**

Non-linearity

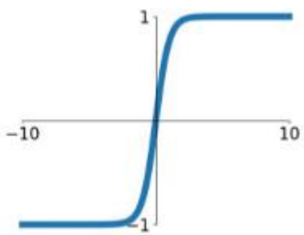
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



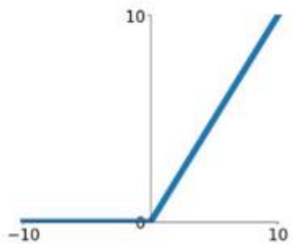
tanh

$\tanh(x)$



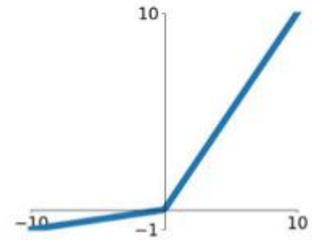
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

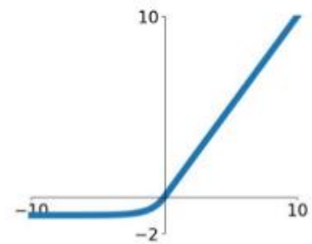


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Non Linearity (ReLU)

- ReLU stands for Rectified Linear Unit for a non-linear operation. The output is $f(x) = \max(0,x)$.
- Why ReLU is important?
- ReLU's purpose is to introduce non-linearity in our ConvNet. Since, the real world data would want our ConvNet to learn would be non-negative linear values.
- There are other non linear functions such as tanh or sigmoid can also be used instead of ReLU.
- Most of the data scientists uses ReLU since performance wise ReLU is better than other two.

Non Linearity (ReLU)

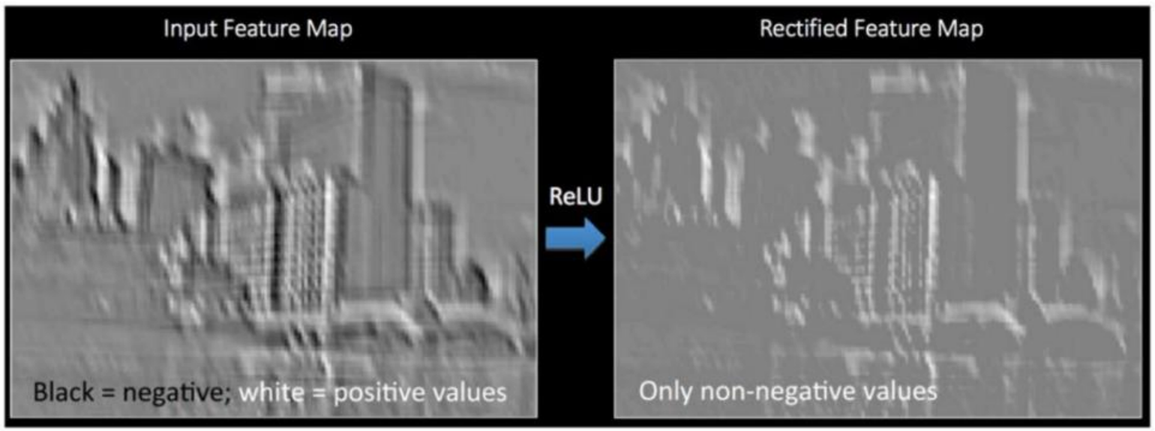
1	14	-9	4
-2	-20	10	6
-3	3	11	1
2	54	-2	80

ReLU →

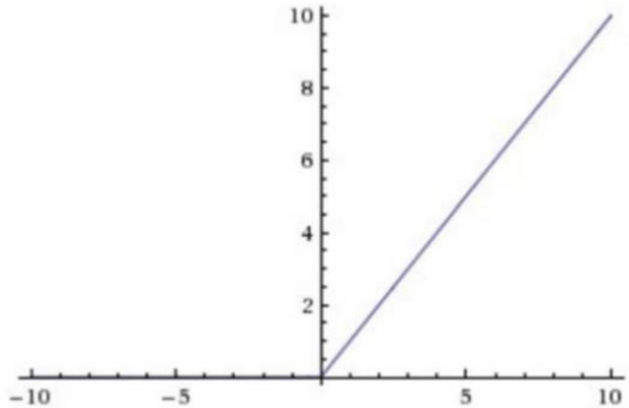
1	14	0	4
0	0	10	6
0	3	11	1
2	54	0	80

Non Linearity (ReLU)

- Apply after every convolution operation (i.e., after convolutional layers)
- ReLU: pixel-by-pixel operation that replaces all negative values by zero. **Non-linear operation!**



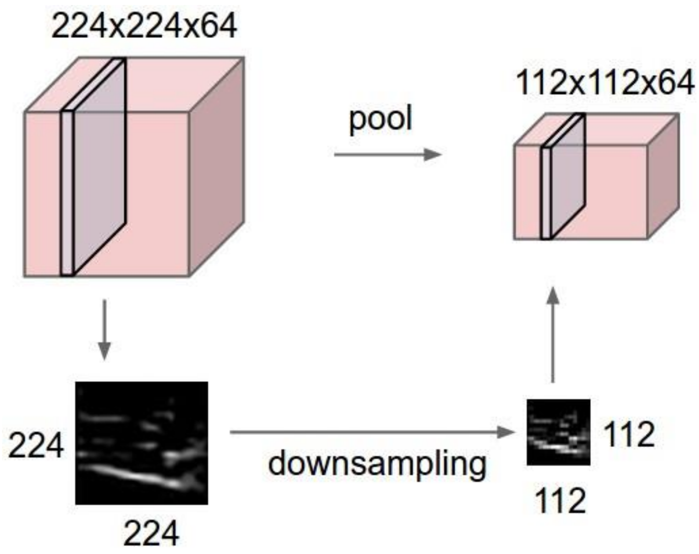
Rectified Linear Unit (ReLU)

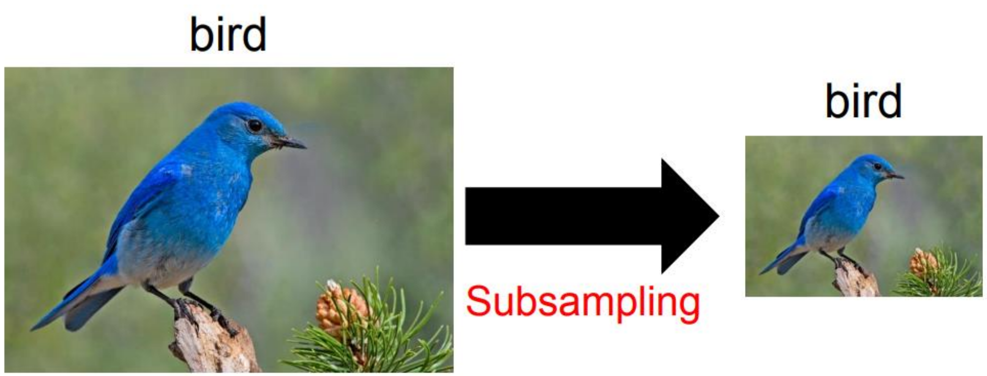


$$g(z) = \max(0, z)$$

Pooling Layers

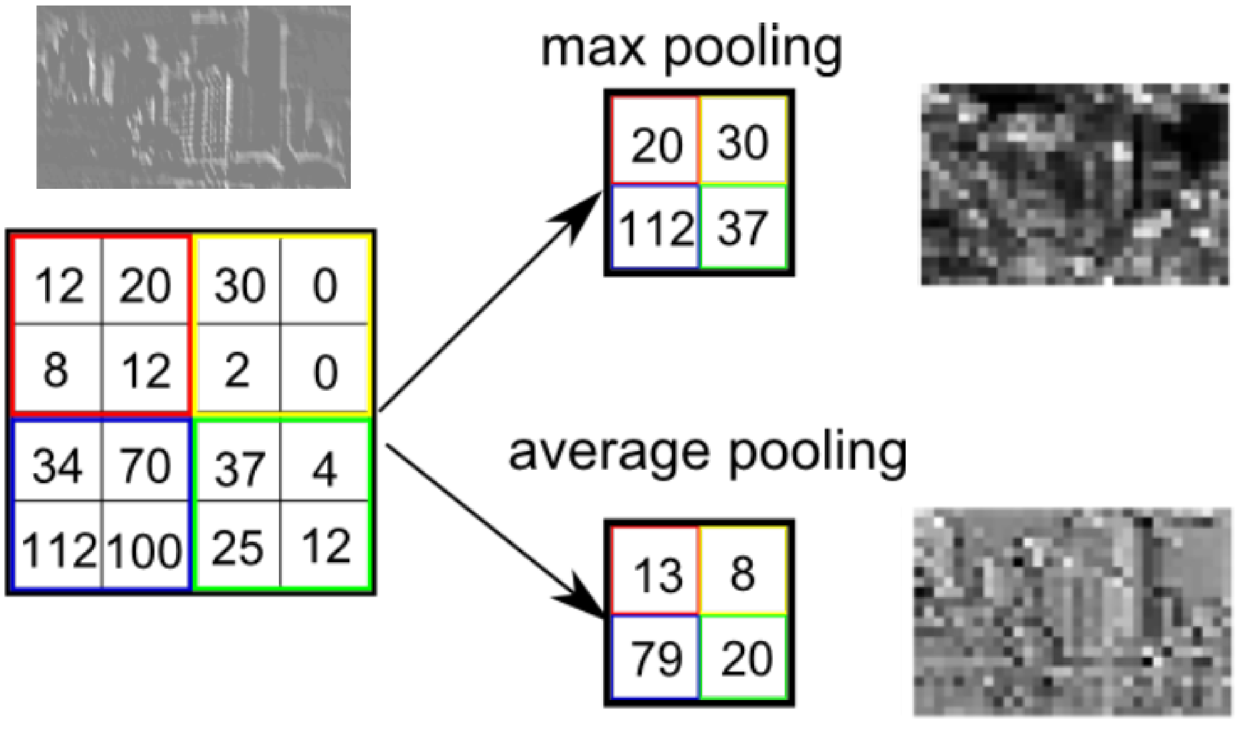
Pooling (or subsampling)
- Make the representations smaller (**will not change the object)**
- (+) Reduce number of parameters and computation



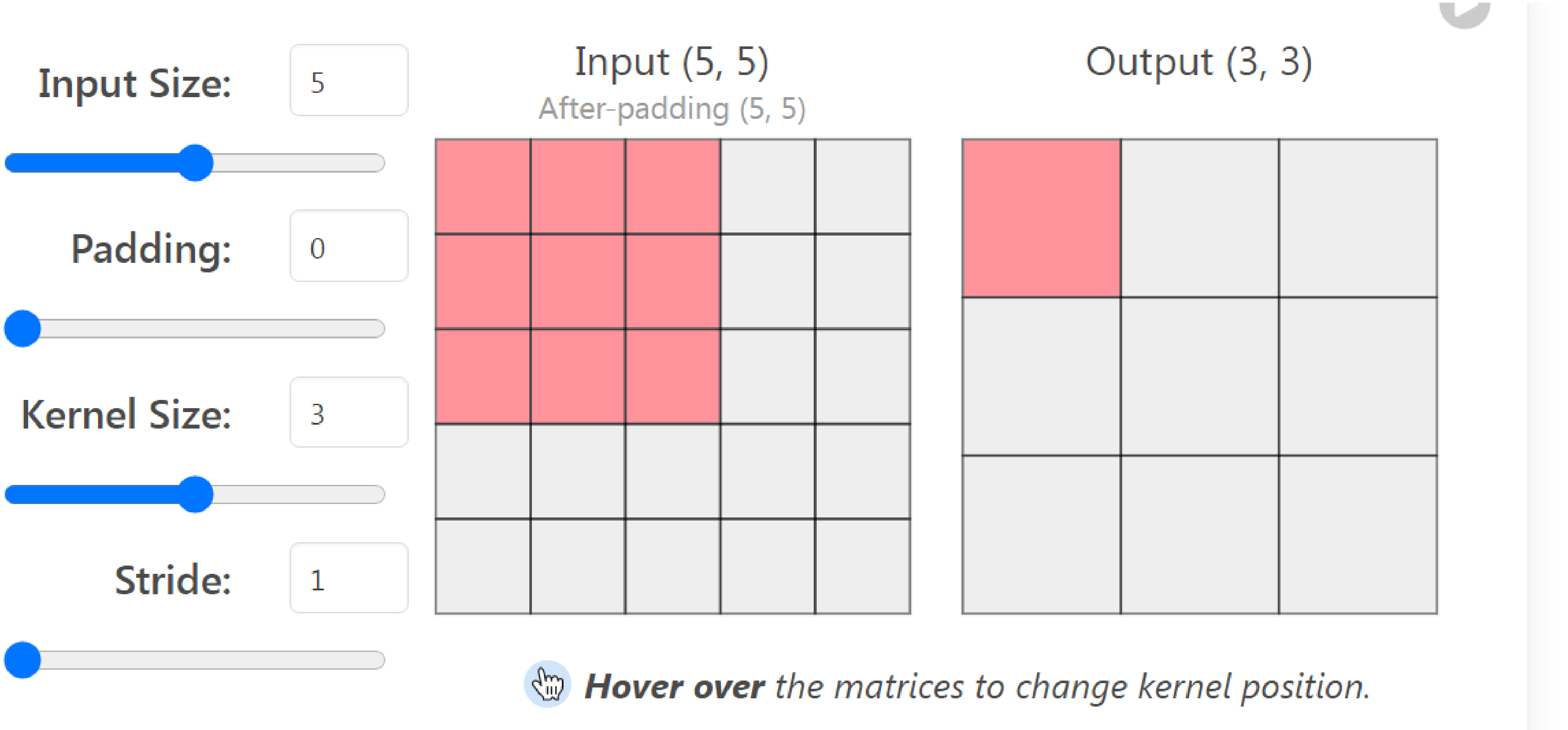


Pooling Layer

- Max pooling and average pooling
  - With 2×2 filters and stride 2

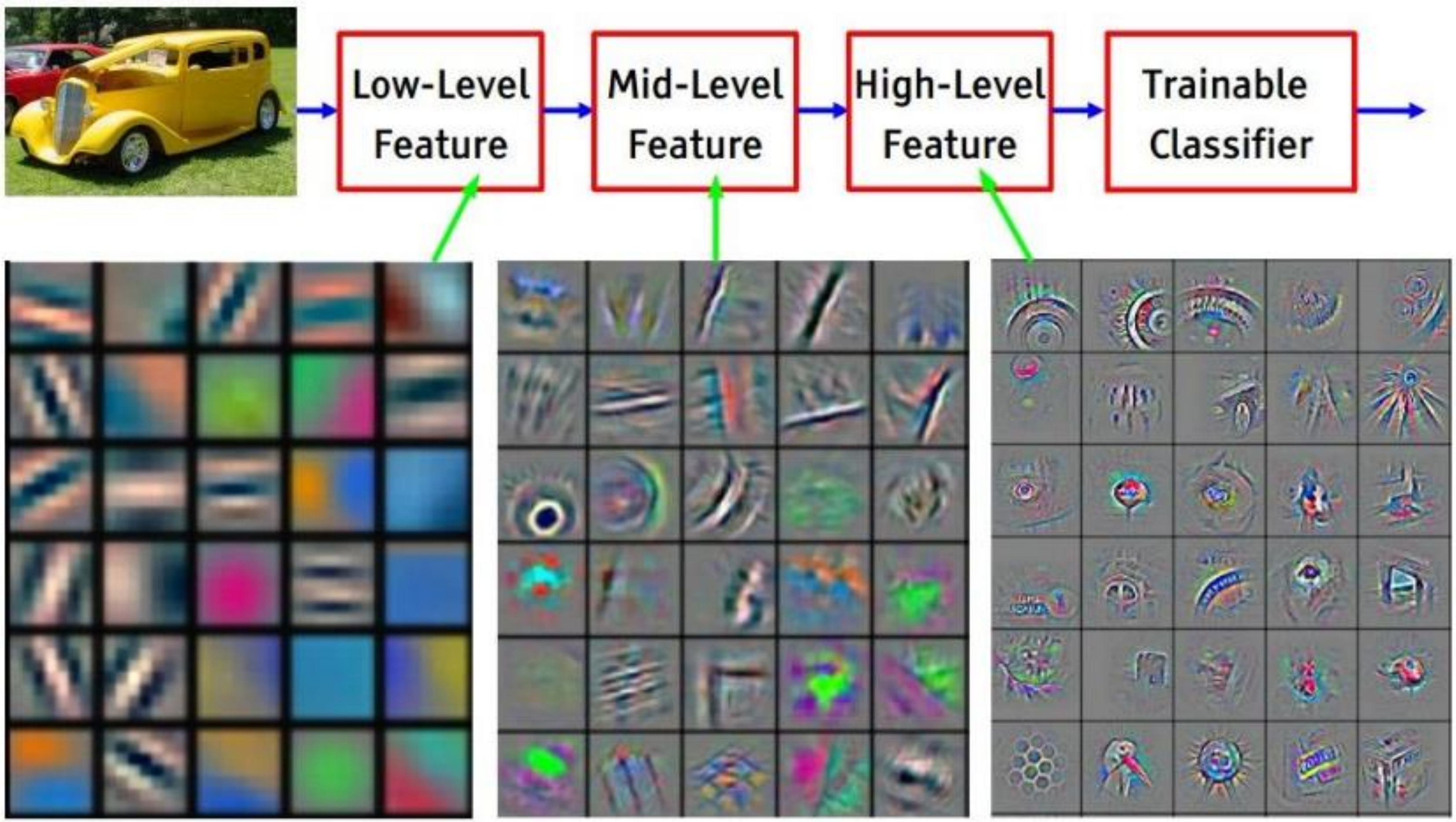


Understanding Hyperparameters



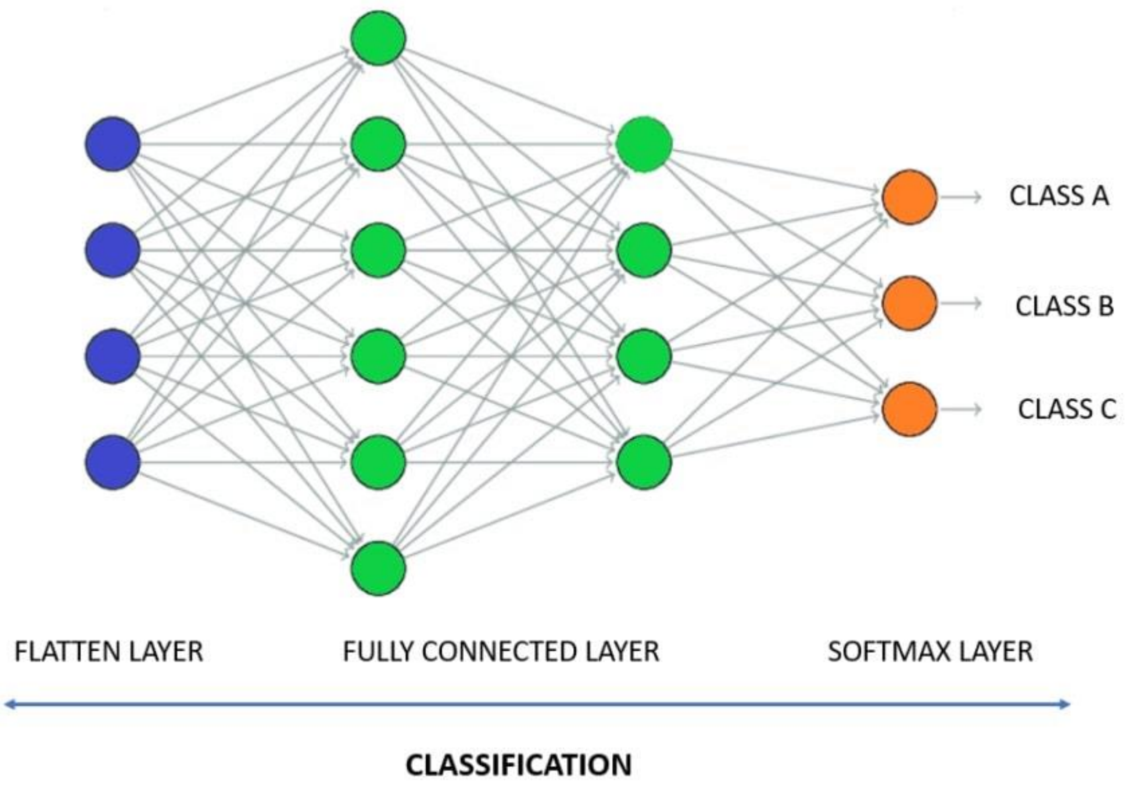
https://poloclub.github.io/cnn-explainer/

Visualization of CNNs layers



Fully Connected Layer

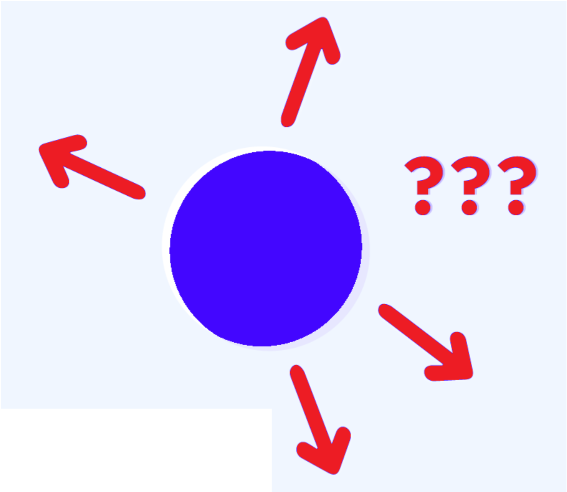
The layer we call as FC layer, we flattened our matrix into vector and feed it into a fully connected layer like neural network.



Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNN)

Given an image of a ball, can you predict where it will go next?



Recurrent Neural Networks (RNN)

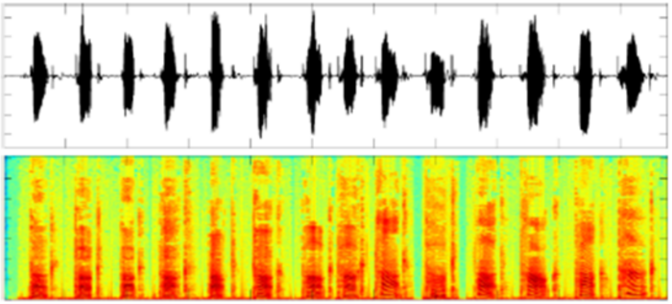
Given an image of a ball, can you predict where it will go next?

Recurrent Neural Networks (RNN)

- Models **temporal** information
- Hidden states as a function of inputs and previous time step information

$$\mathbf{h}_t = f(\mathbf{h}_{t-1}, \mathbf{x}_t; \Theta)$$

- Temporal information is important in many applications
  - Language
  - Speech
  - Video



Sequential Data

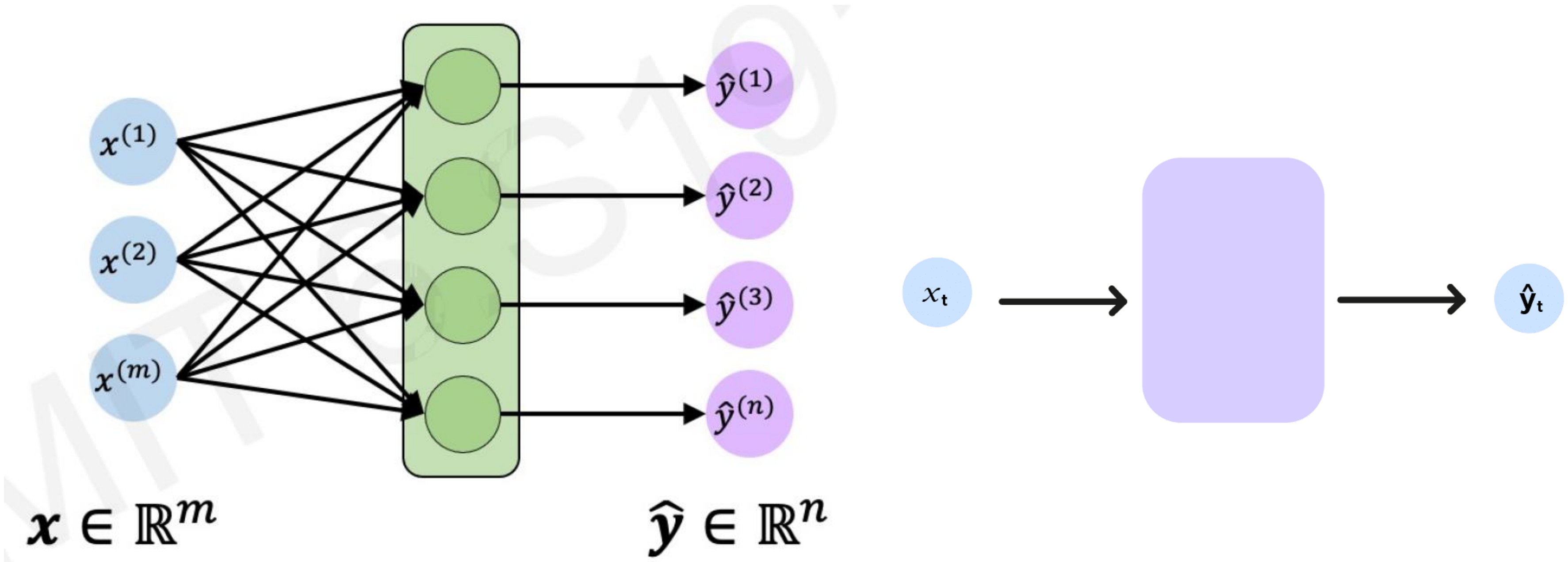
Sometimes the sequence of data matters.

- **Text generation**
- **Stock price prediction**

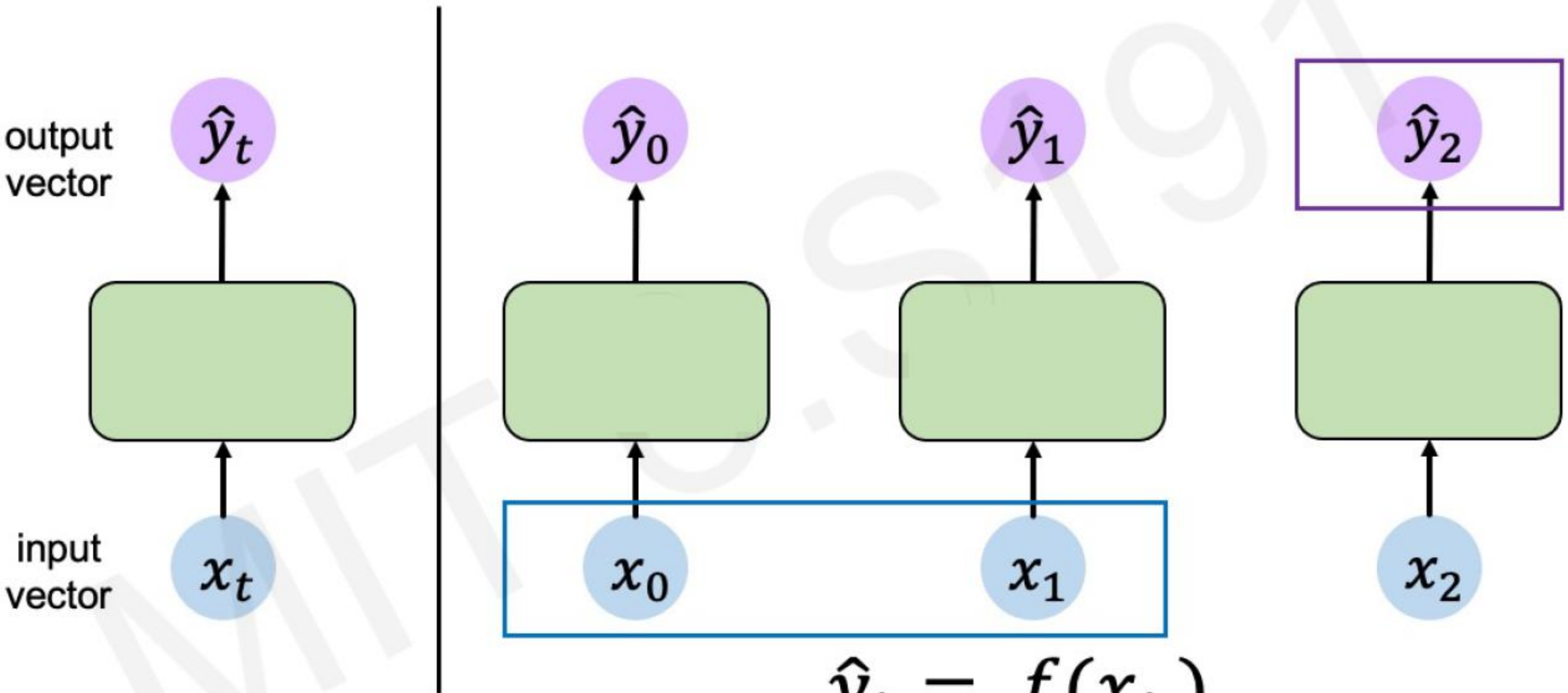
The clouds are in the the ....

sky

Feed-Forward Network

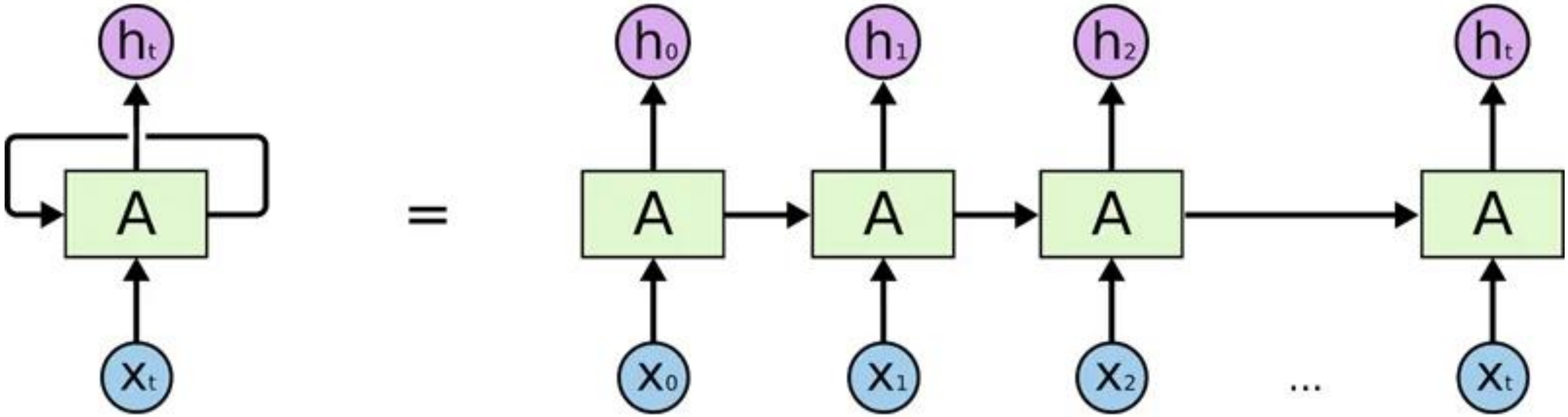


Handling Individual Time steps

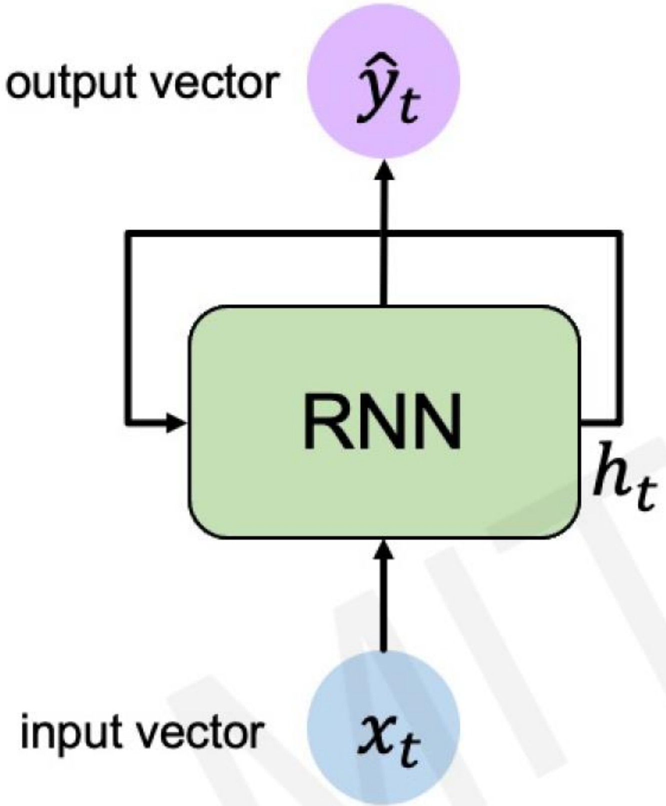


$$\hat{y}_t = f(x_t)$$

Recurrent Neural Network



Recurrent Neural Network

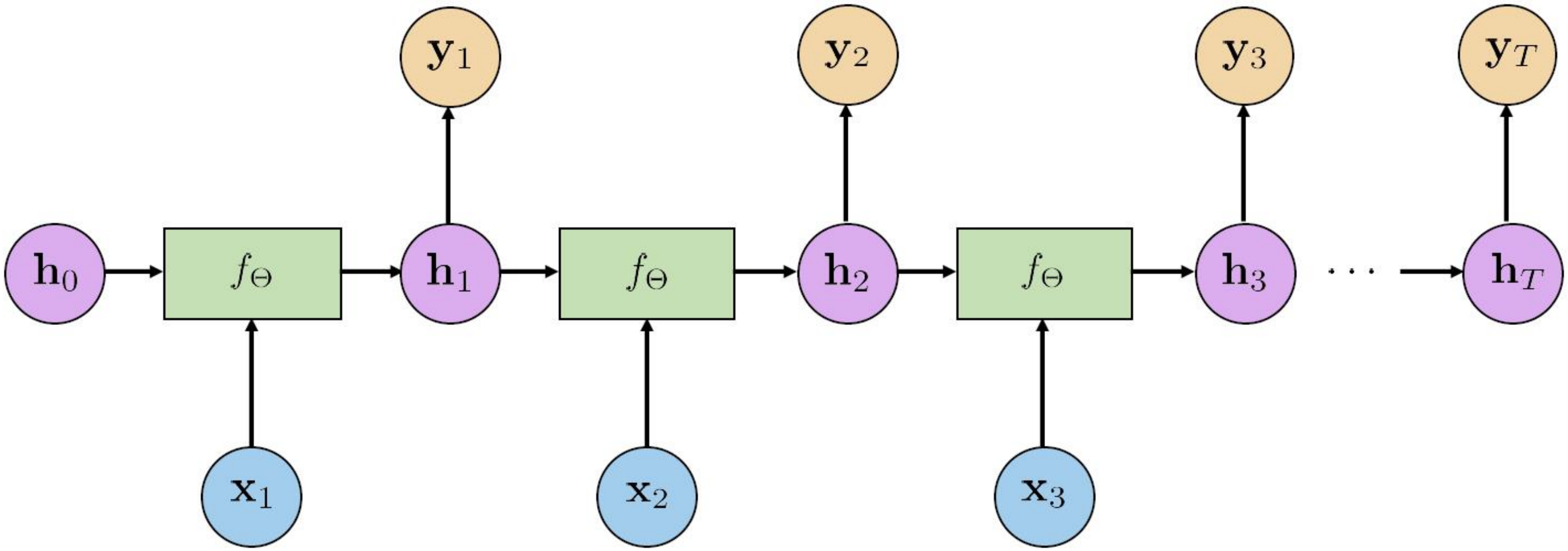


Apply a **recurrence relation** at every time step to process a sequence:

$$\underbrace{\boxed{h_t}}_{\text{cell state}} = \underbrace{\boxed{f_W}}_{\text{function with weights } W}(\underbrace{\boxed{x_t}}_{\text{input}}, \underbrace{\boxed{h_{t-1}}}_{\text{old state}})$$

Note: the same function and set of parameters are used at every time step

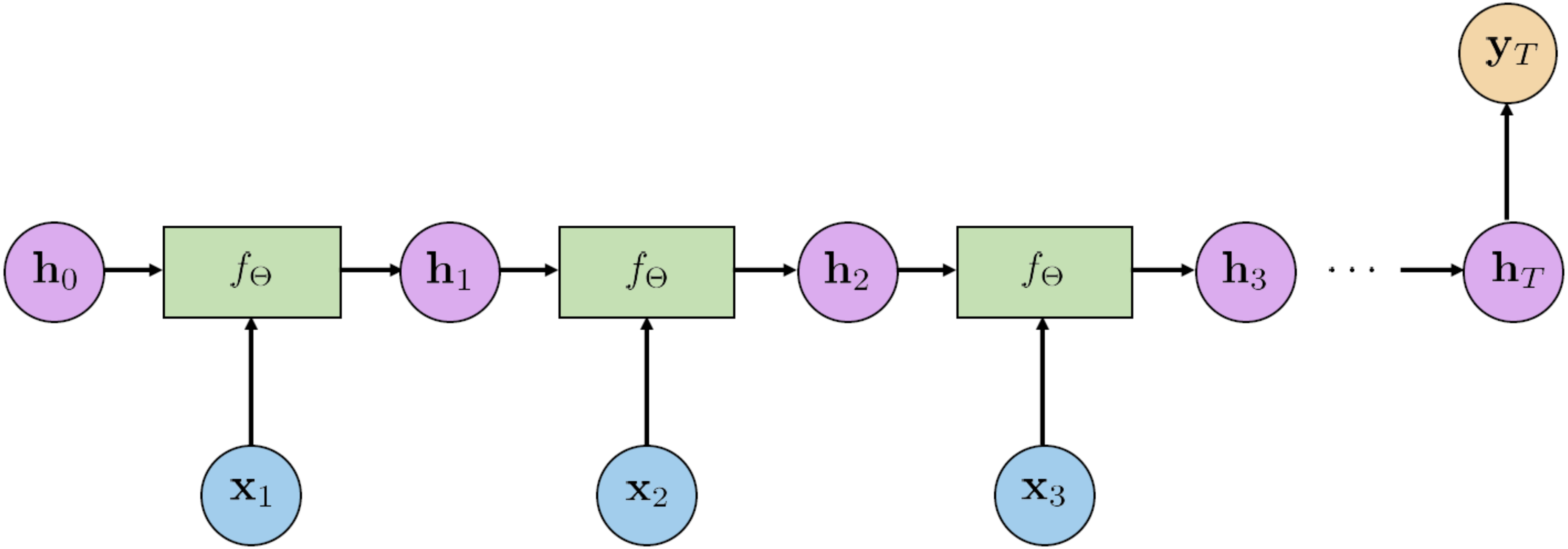
RNN: Computation Graph (Many to Many)



e.g., **Machine Translation**
(Sequence of words → Sequence of words)

Input sentence:	*Translation (PBMT):*	*Translation (GNMT):*	*Translation (human):*
李克強此行將啟動中加總理年度對話機制，與加拿大總理杜魯多舉行兩國總理首次年度對話。	Li Keqiang premier added this line to start the annual dialogue mechanism with the Canadian Prime Minister Trudeau two prime ministers held its first annual session.	Li Keqiang will start the annual dialogue mechanism with Prime Minister Trudeau of Canada and hold the first annual dialogue between the two premiers.	Li Keqiang will initiate the annual dialogue mechanism between premiers of China and Canada during this visit, and hold the first annual dialogue with Premier Trudeau of Canada.

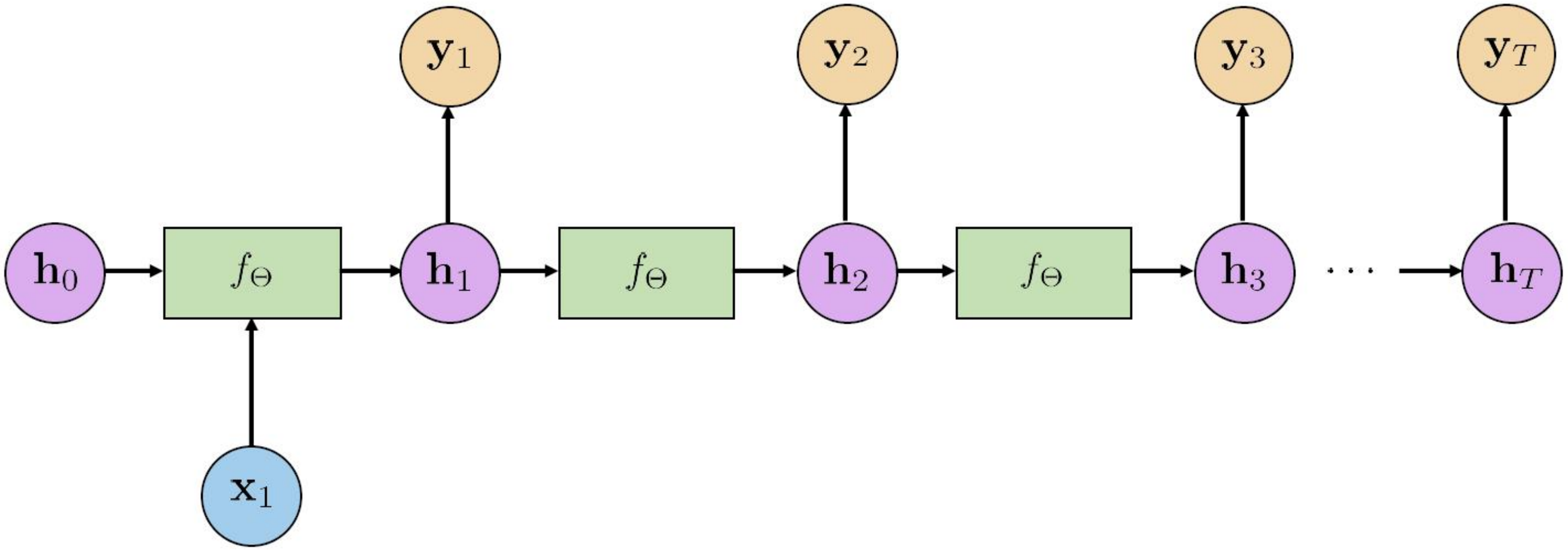
RNN: Computation Graph (Many to one)



NN Class is very interesting!

e.g., **Sentiment Classification**
(Sequence of words → sentiment)

RNN: Computation Graph (One to Many)



e.g., **Image Captioning**
(Image → sequence of words)

No errors

A white teddy bear sitting in the grass

Minor errors

A man in a baseball uniform throwing a ball

Somewhat related

A woman is holding a cat in her hand

RNNs- Image Captioning Examples

Describes without errors	Describes with minor errors	Somewhat related to the image

A person riding a motorcycle on a dirt road.

Two dogs play in the grass.

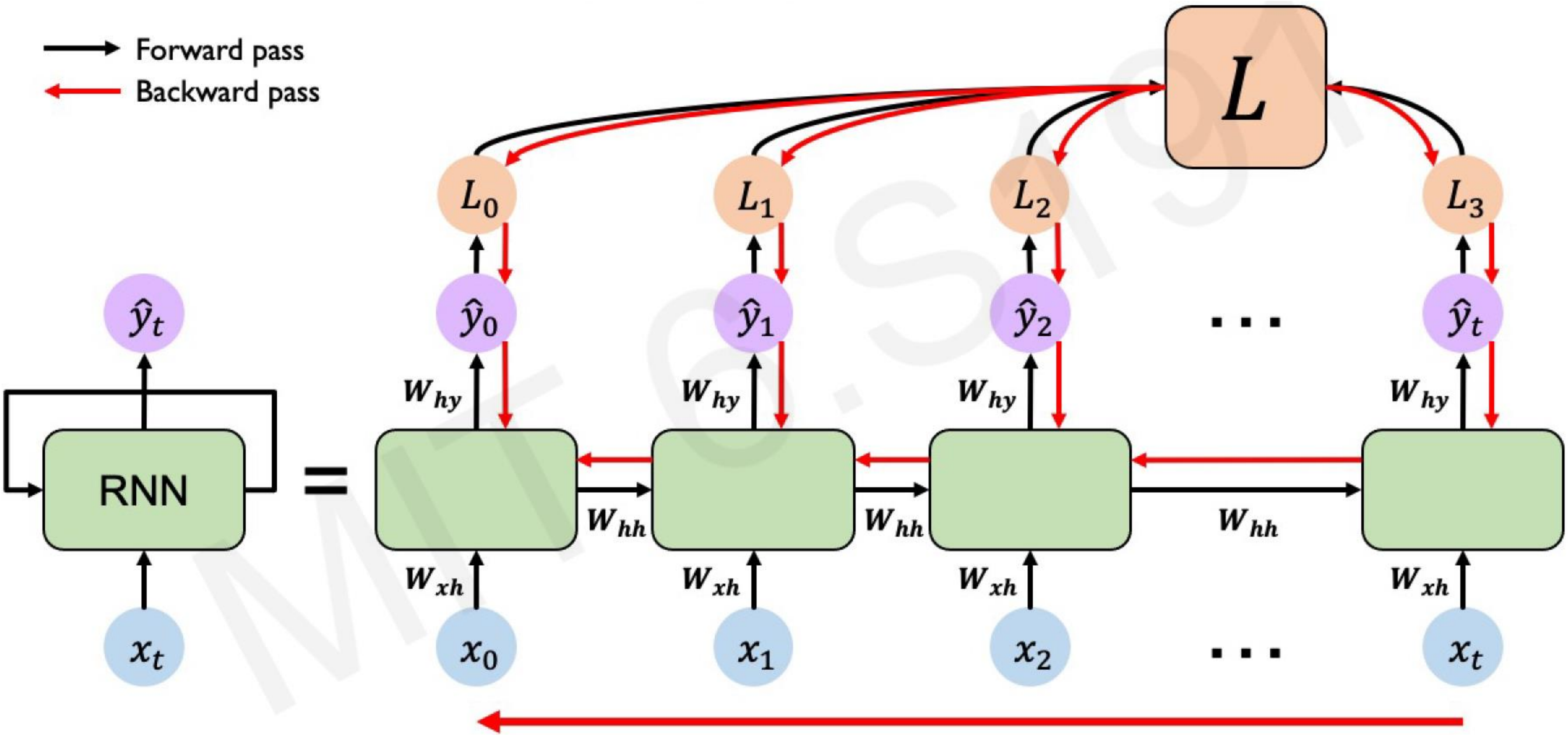
A skateboarder does a trick on a ramp.

A group of young people playing a game of frisbee.

Two hockey players are fighting over the puck.

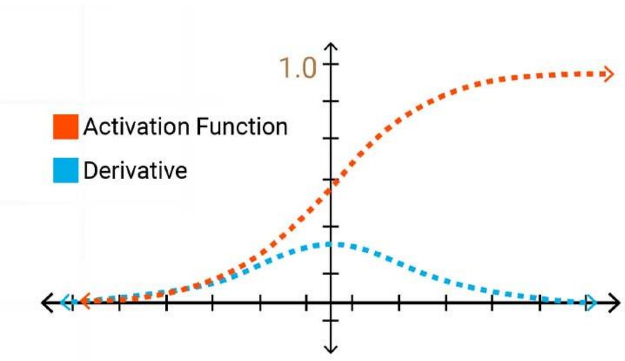
A little girl in a pink hat is blowing bubbles.

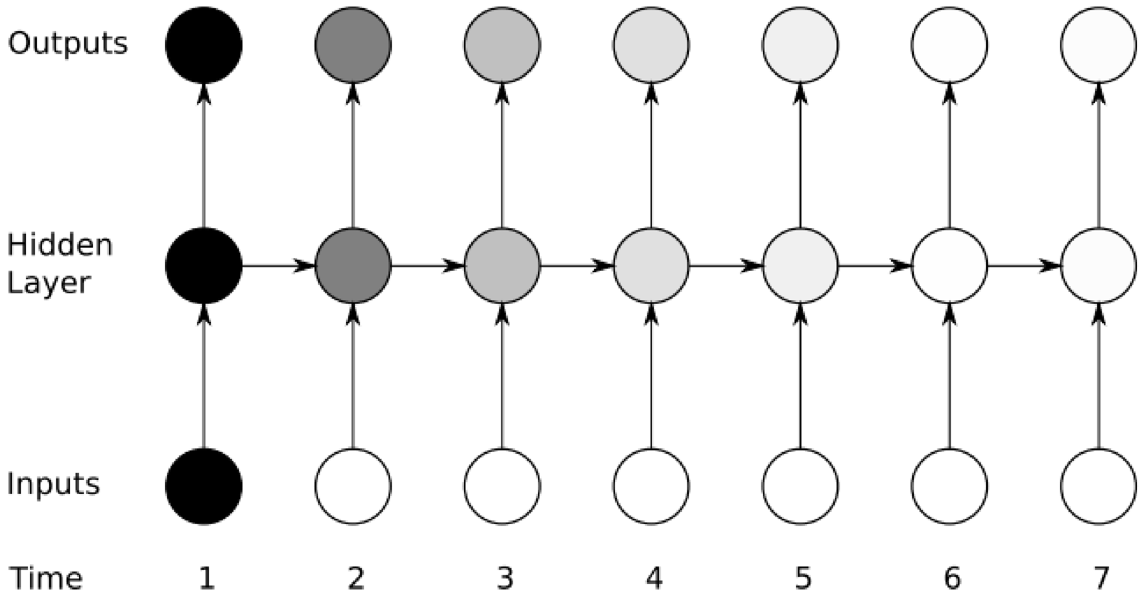
RNN Training -Backpropagation Through Time



Vanishing Gradient Over Time

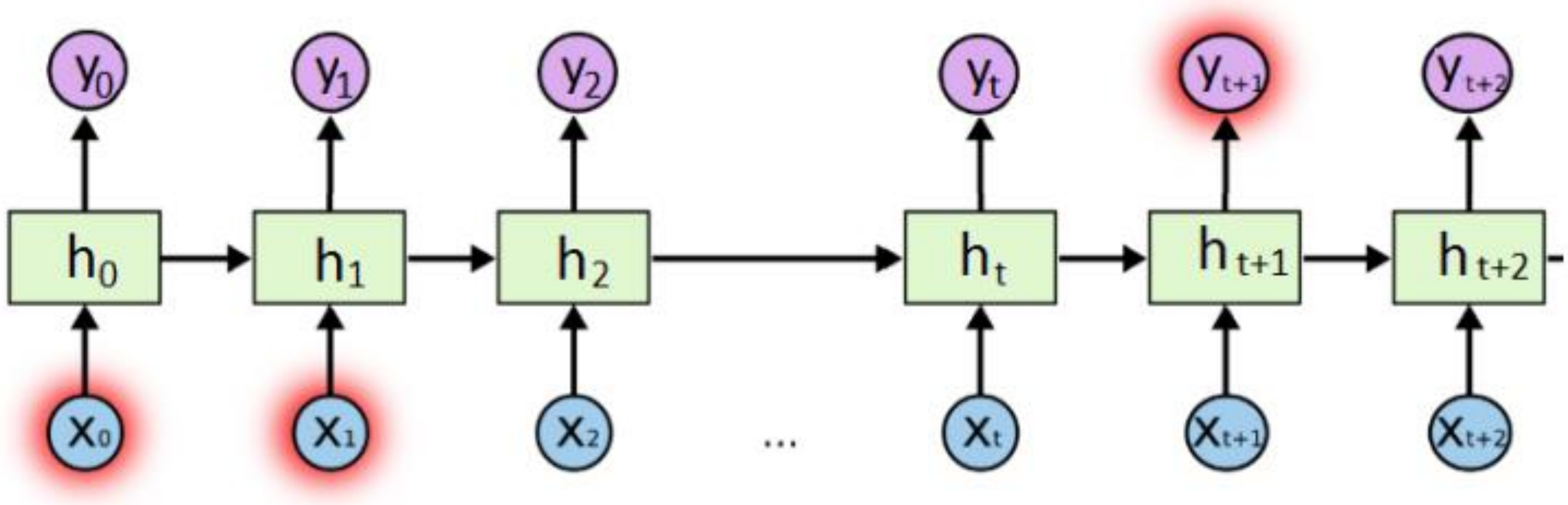
This is more problematic in vanilla RNN (with tanh/sigmoid activation)
- When trying to handle long temporal dependency
- The gradient **vanishes over time**





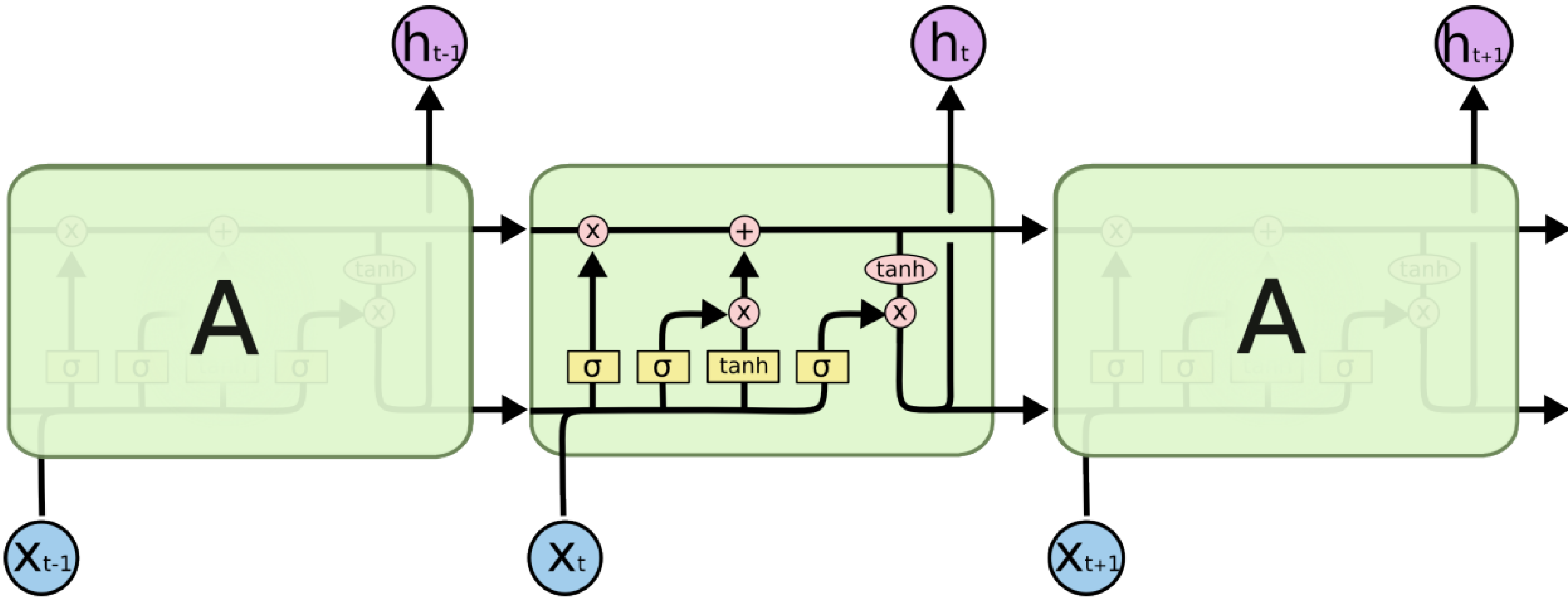
The Problem of Long-term Dependencies

Iran is my home country, and therefore, I can speak **…**

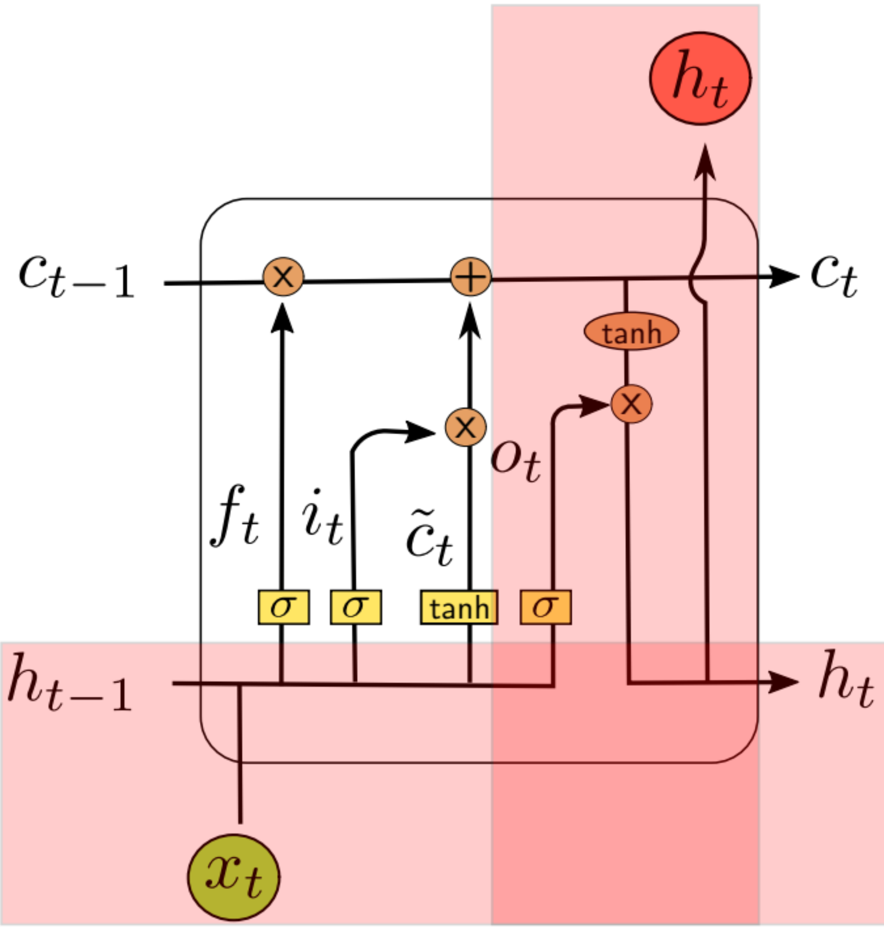


Long Short-Term Memory (LSTM)

LSTM networks are RNNs capable of learning long-term dependencies



LSTM gates



$f_t = \sigma(W_f \cdot [h_{t-1}, x_t])$ (Forget)

$i_t = \sigma(W_i \cdot [h_{t-1}, x_t])$ (Input)

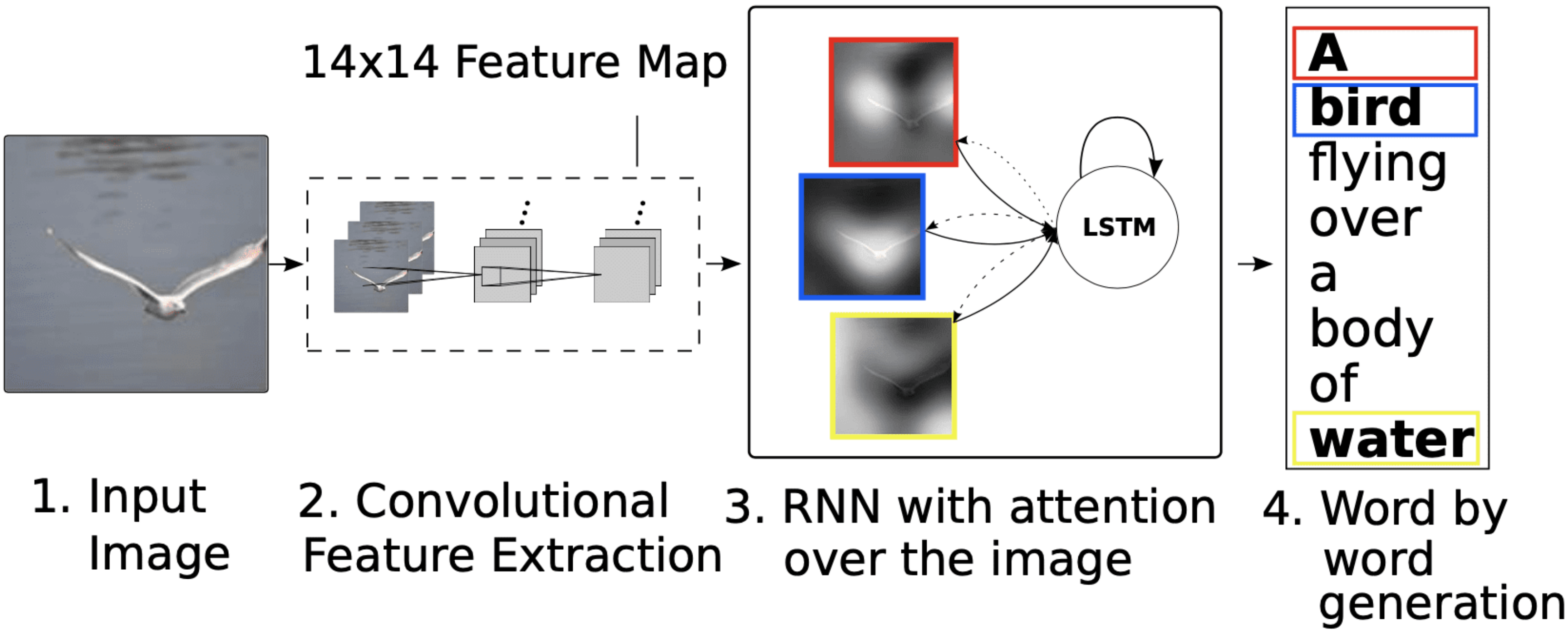
$\tilde{c}_t = tanh(W_c \cdot [h_{t-1}, x_t])$

$c_t = f_t \times c_{t-1} + i_t \times \tilde{c}_t$ (Update)

$o_t = \sigma(W_o \cdot [h_{t-1}, x_t])$

$h_t = o_t \times tanh(c_t)$ (Output)

RNNs - Attention Mechanism



RNNs - Attention Mechanism Examples

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

Generative Artificial Intelligence

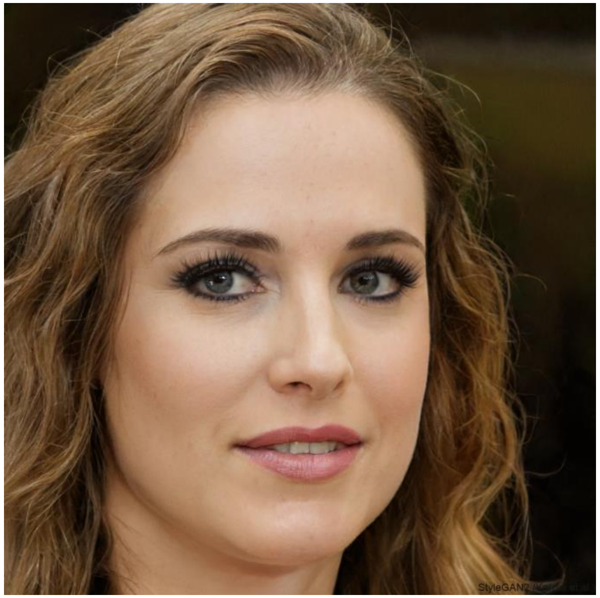
What is Generative AI?

- Generative AI is a type of artificial intelligence that uses neural networks to create text, images, and other content.
- It is based on the idea of a generative model, which is a statistical model that can generate new data samples that are similar to the data that it was trained on.
- Generative models are trained on large amounts of data, and they learn to identify patterns in the data.
- Once a generative model has been trained, it can be used to generate new data samples..

Generative AI Models

- **Generative Adversarial Networks (GANs):** Compete to create realistic content. Applications: Image generation, style transfer, data augmentation.
- **Variational Autoencoders (VAEs):**Learn latent representations of data. Applications: Image generation, anomaly detection, data compression.
- **Large Language Models (LLMs):**Process and generate text. Applications: Text generation, translation, summarization, code generation.
- **Diffusion Models:** Gradually add noise to an image and then denoise it. Applications: Image generation, image editing, text-to-image generation.

Which face is real?



A

B

C

Supervised vs Unsupervised Learning

Supervised Learning:

Given data x, predict output y

Goal: Learn a function to map x → y

Requires labeled data

Methods: Classification, Regression, Detection, Segmentation

Unsupervised Learning

Given data x

Goal: Learn the hidden or underlying structure of the data

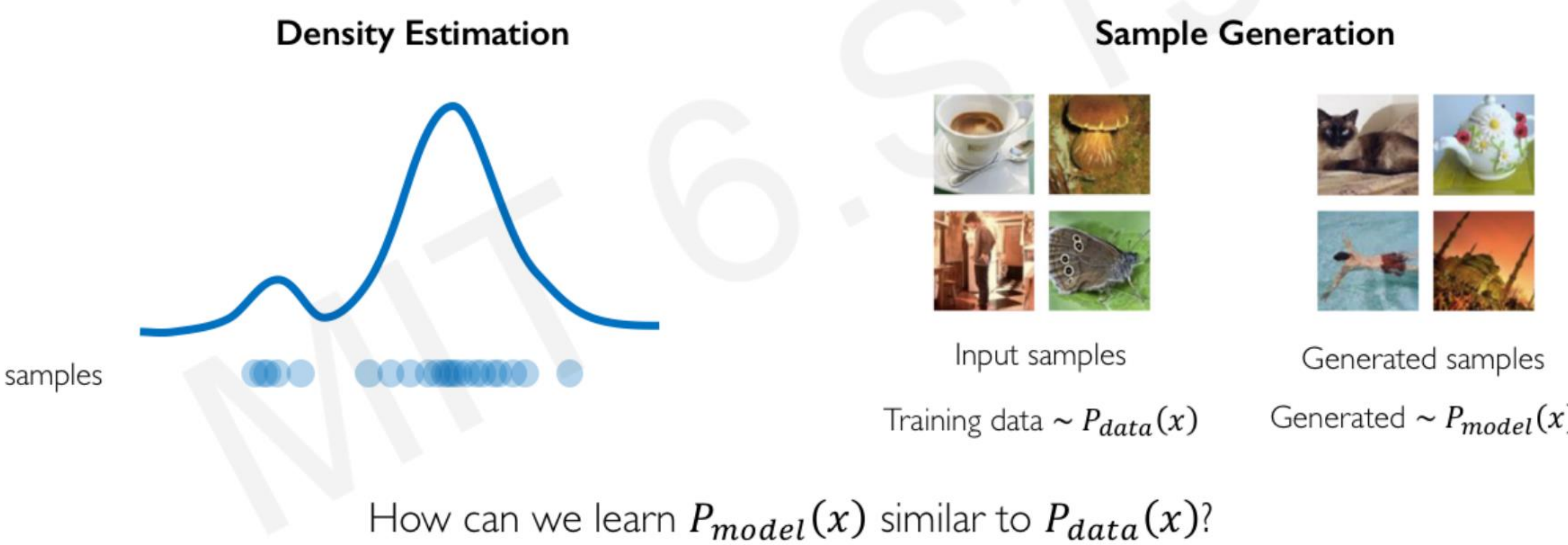
Requires data (no labels)

Methods: Clustering/Density, Compression

Generative Modeling

Goal: take as input training samples from some distribution and learn a model that represents the distribution.

Two operations:



Generative Modeling- Debiasing

Capable of uncovering **underlying features** in a dataset

vs

Homogeneous skin color, pose

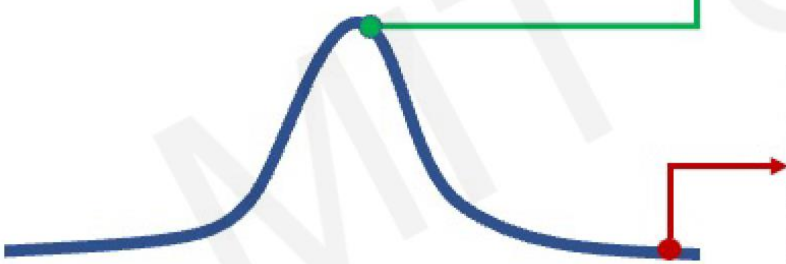
Diverse skin color, pose, illumination

How can we use this information to create fair and representative datasets?

Generative Modeling- Outlier detection

- **Problem:** How can we detect when we encounter something new or rare?
- **Strategy:** Leverage generative models, detect outliers in the distribution
- Use outliers during training to improve even more!

95% of Driving Data:
(1) sunny, (2) highway, (3) straight road



Detect outliers to avoid unpredictable behavior when training

Edge Cases

Harsh Weather

Pedestrians

Generative Adversarial Networks Introduction

• First introduced by Ian Goodfellow et al. in 2014
• GANs have been used to generate images, videos, poems, and some simple conversation

Generator

Generates candidates/images (from a probability distribution)
It's objective is to 'fool' the discriminator by producing novel synthesized instances that appear to come from the true data

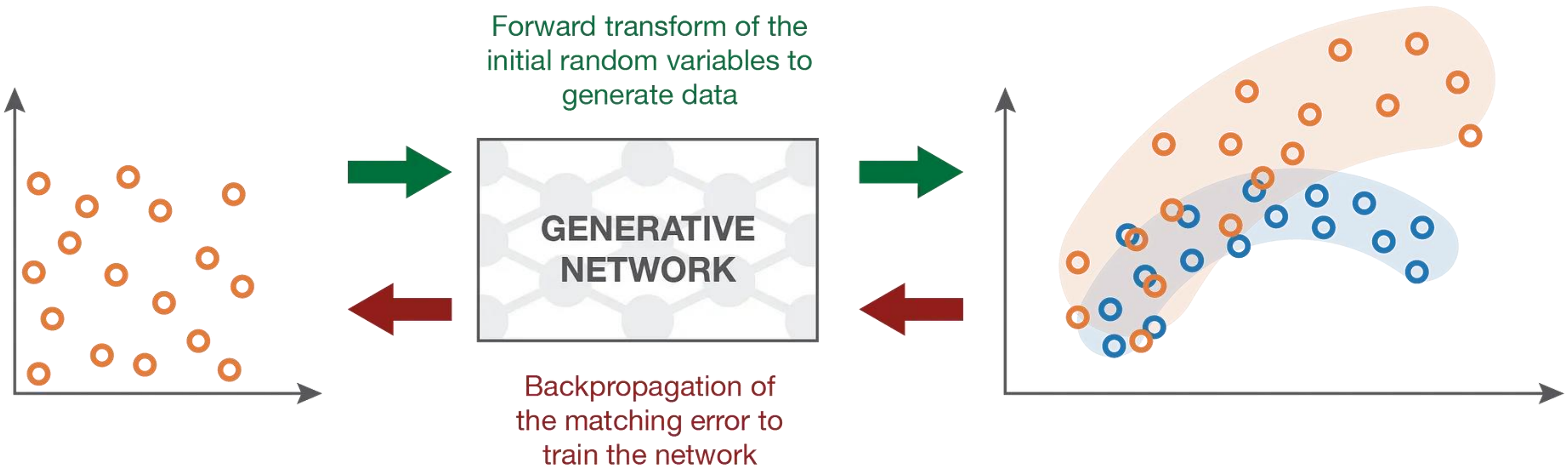
Discriminator

Evaluates the generated images to see if they come from the true data or not

Backpropagationapplied to both networks:
•Generator to produce better images
•Discriminator to be more skilled at evaluating generated images

Generative Adversarial Networks Introduction

- GANs are deep neural net architectures comprised of two neural networks, competing one against the other and playing an adversarial game against each other.
- Gangs are neural networks that trained in an adversarial manner to generate data mimicking some distribution.

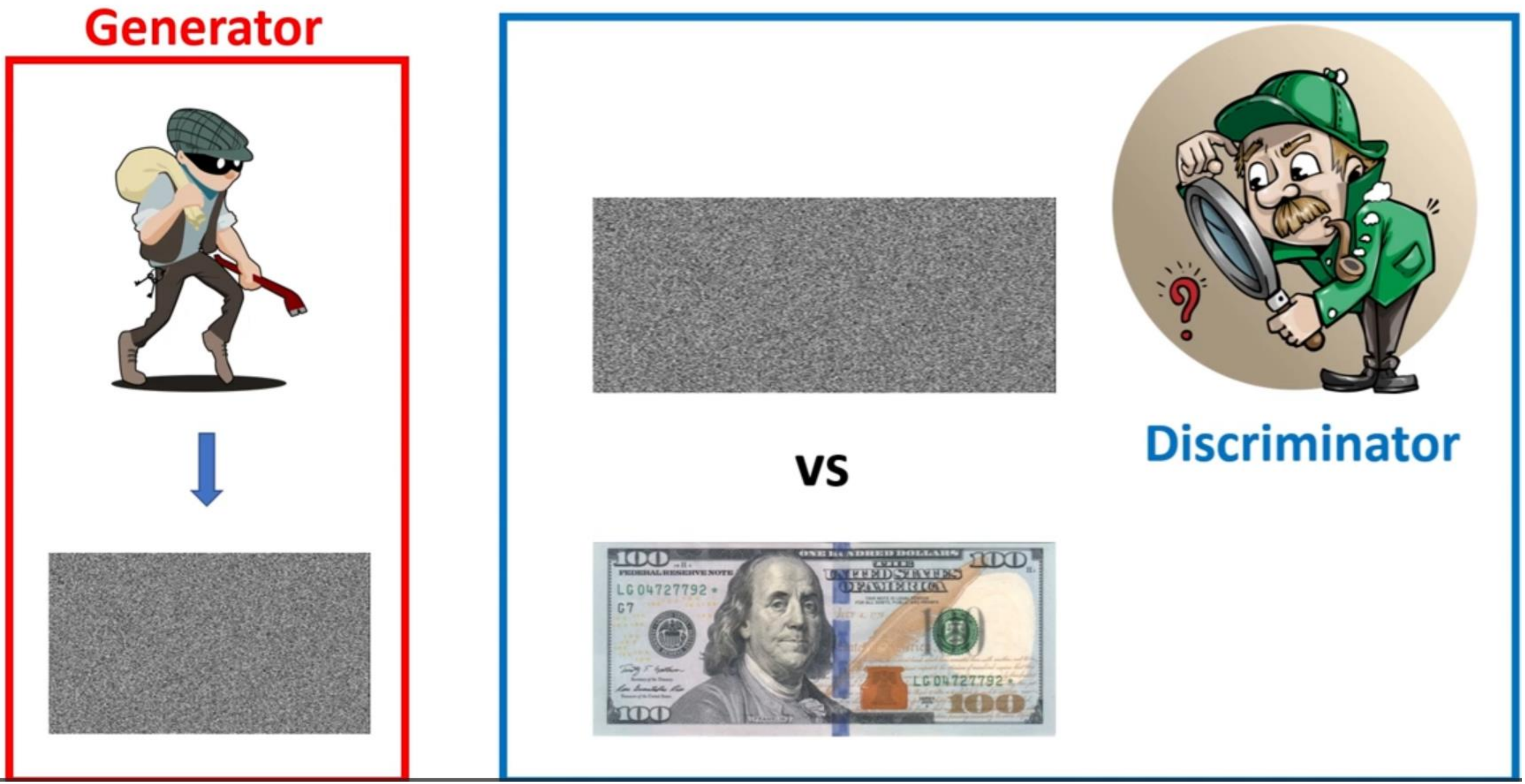


Input random variables (drawn from a uniform).

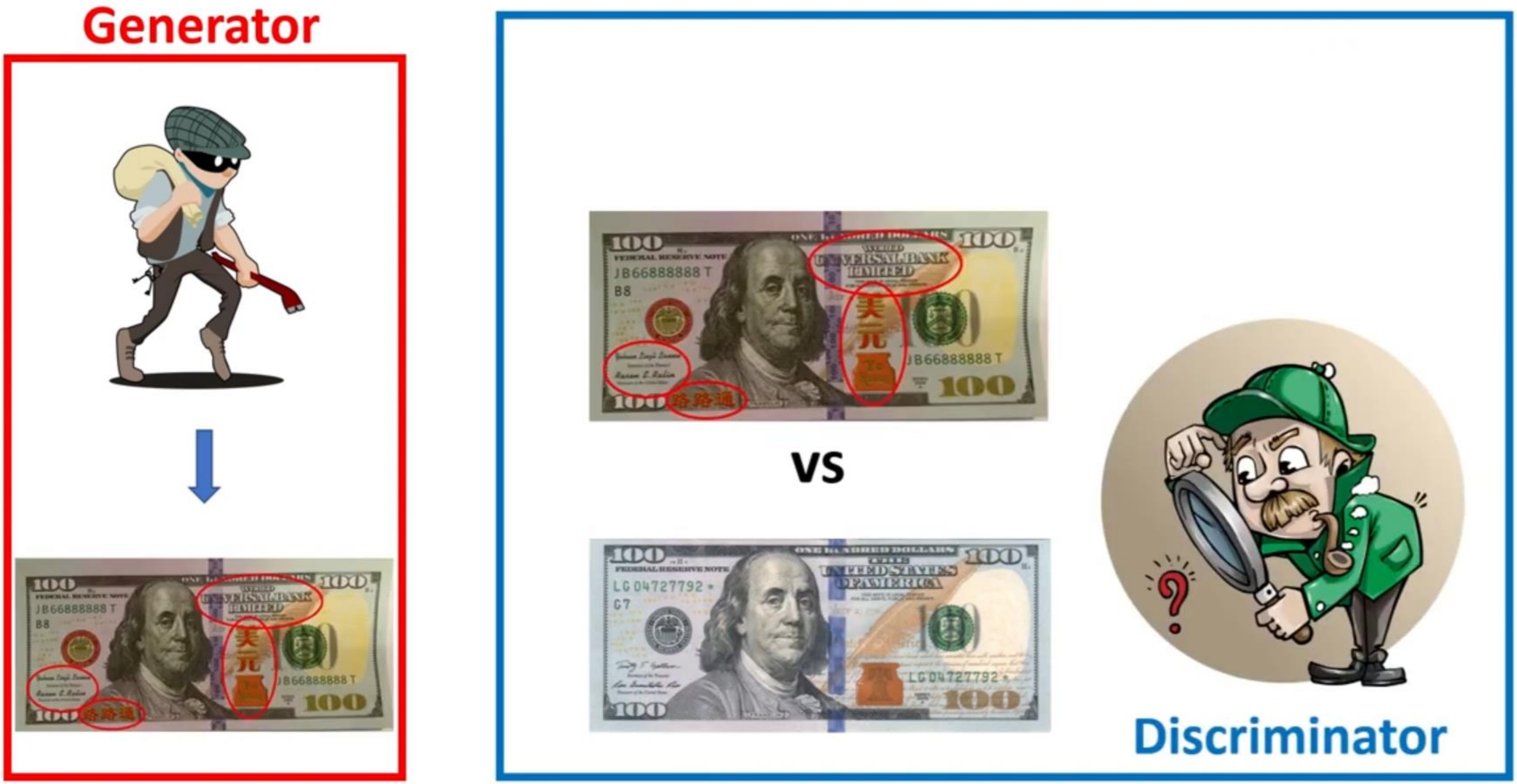
Generative network to be trained.

The generated distribution is compared to the true distribution and the "matching error" is backpropagated to train the network.

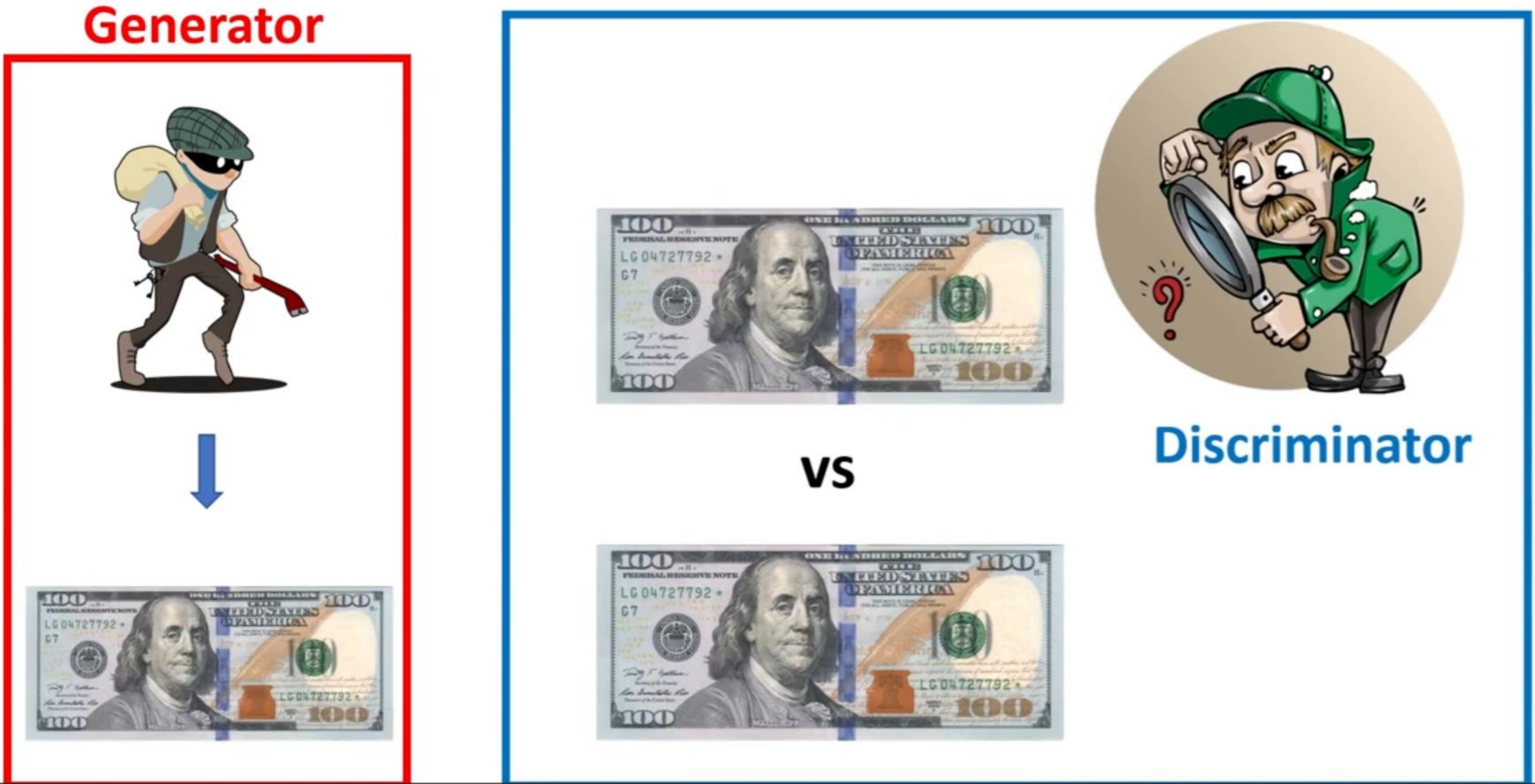
Generator & Discriminator



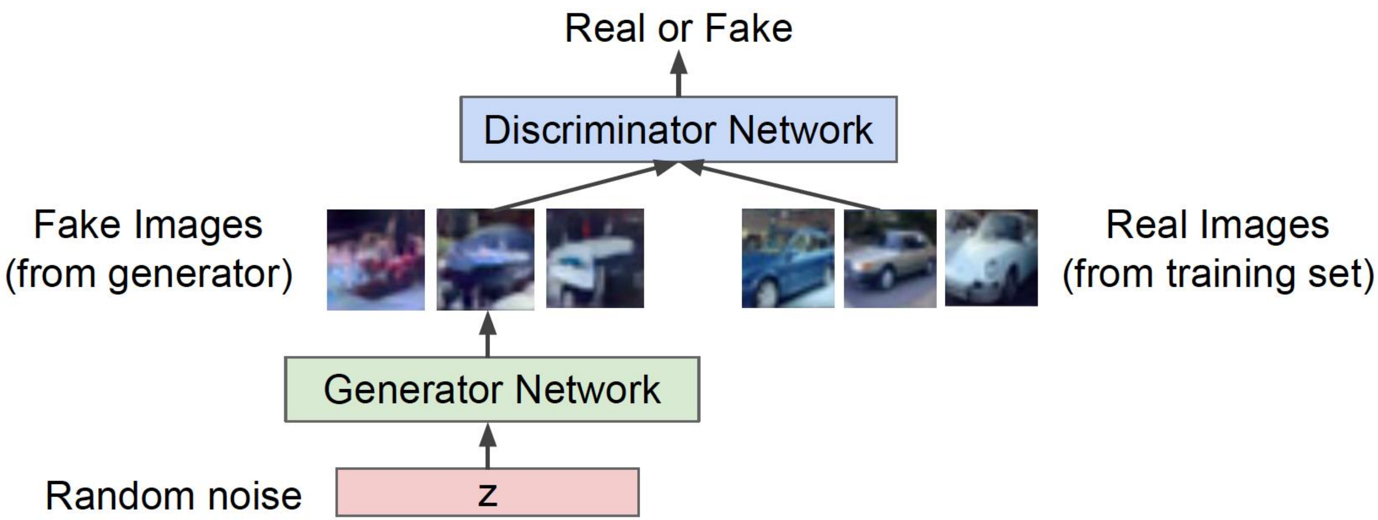
Generator & Discriminator



Generator & Discriminator



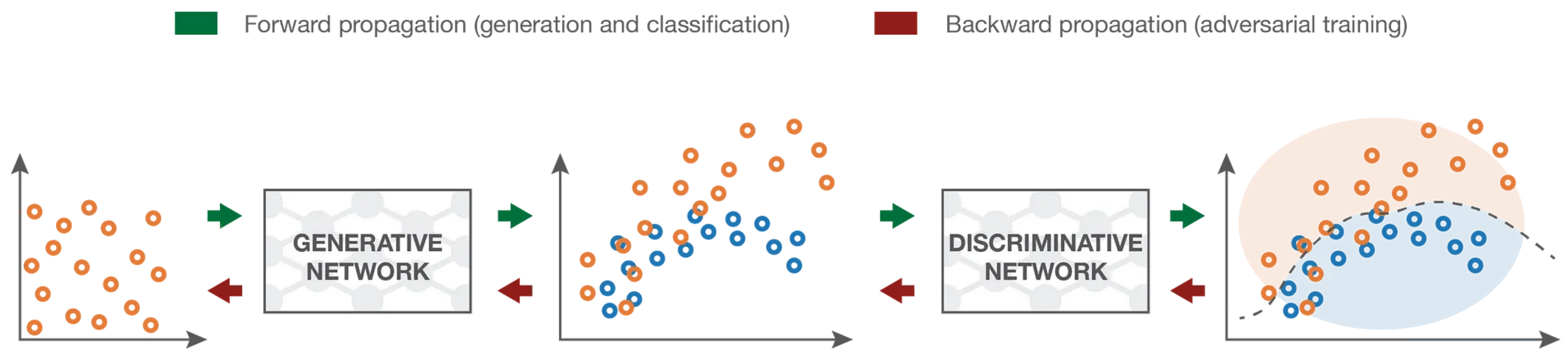
GANs - Training Objective



Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log \underbrace{D_{\theta_d}(x)}_{\text{Discriminator output for real data x}} + \mathbb{E}_{z \sim p(z)} \log(1 - \underbrace{D_{\theta_d}(G_{\theta_g}(z))}_{\text{Discriminator output for generated fake data G(z)}}) \right]$$

GANs - Training Objective

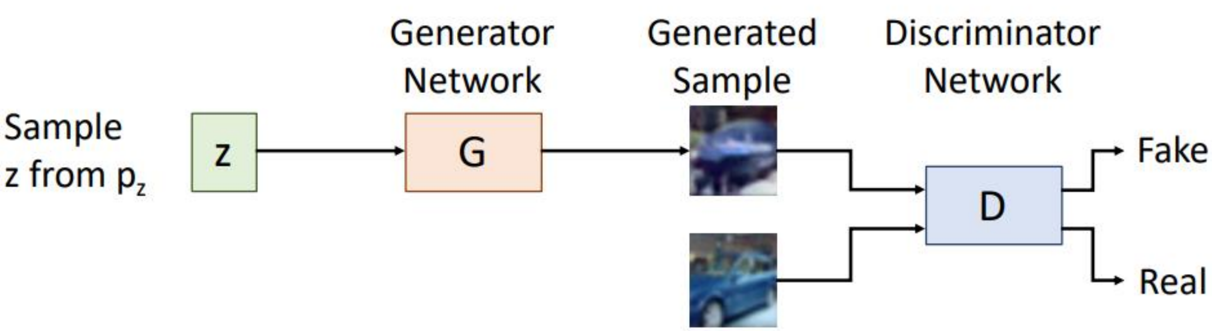


Input random variables.	The generative network is trained to **maximise** the final classification error.	The generated distribution and the true distribution are not compared directly.	The discriminative network is trained to **minimise** the final classification error.	The classification error is the basis metric for the training of both networks.

GANs - Training Objective

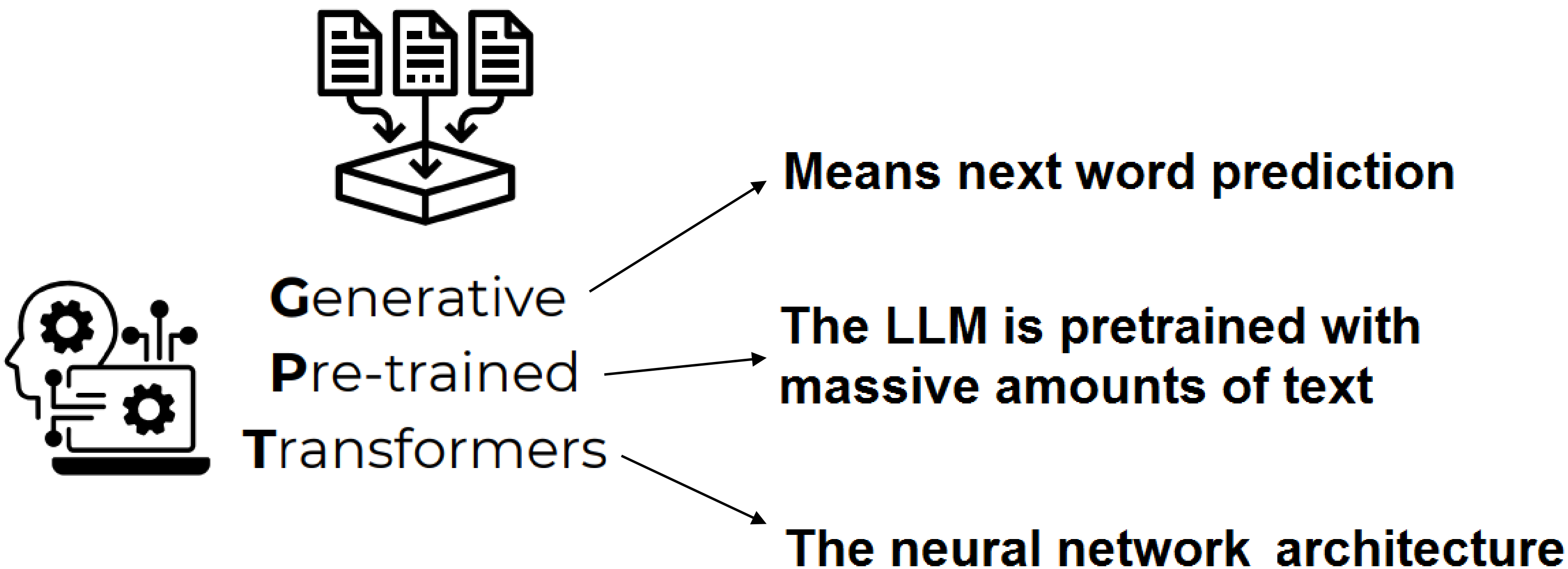
Jointly train generator G and discriminator D with a minimax game

$$\min_{G} \max_{D} \left(\overbrace{E_{x\sim p_{data}}[\log D(x)]}^{\text{Discriminator wants } D(x) = 1 \text{ for real data}} + E_{z\sim p(z)} \underbrace{\overbrace{\left[\log\left(1 - D(G(z))\right)\right]}^{\text{Discriminator wants } D(x) = 0 \text{ for fake data}}}_{\text{Generator wants } D(x) = 1 \text{ for fake data}} \right)$$



Large language models (LLM)

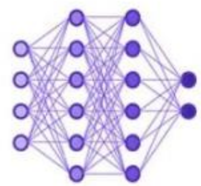
Large language models, like ChatGPT, are designed to generate human-like text based on the patterns they learn from vast amounts of data.



Large language models (LLM)

The cat likes to sleep in the

Input



Neural Network
(LLM)

Word	Probability
ability	0.002
bag	0.071
box	**0.085**
...	...
zebra	0.001

Output

Language modeling is learning to predict the next word.

Transformer Architecture

Two main building blocks an **encoder** and a **decoder** block

- Attention modules
- Position-wise feed-forward networks
- Residual Connection and Normalization
- Positional encoding

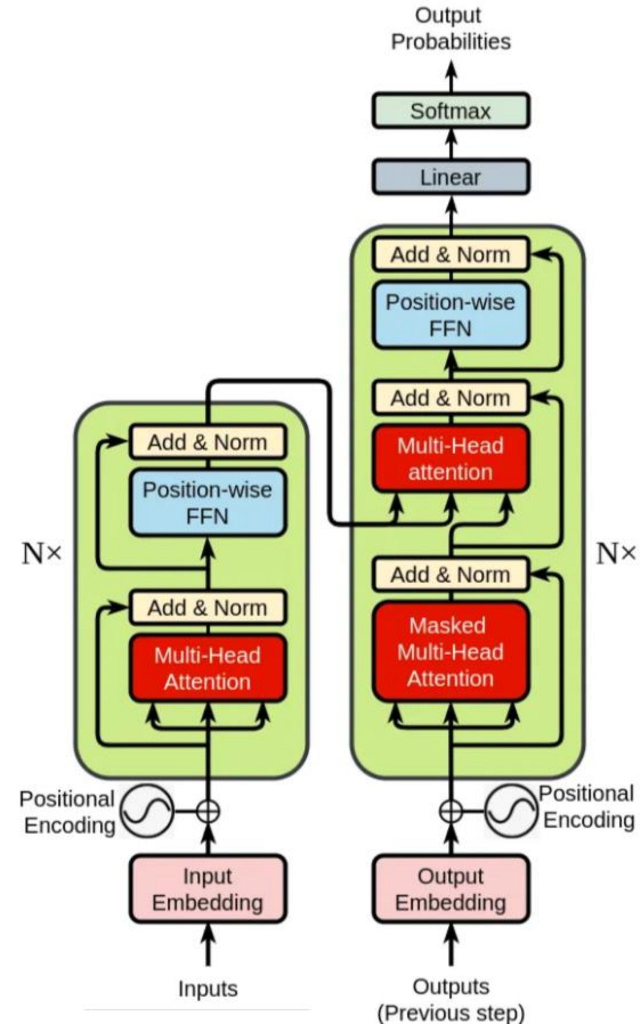


Image Inpainting

Conditional Image

Inpainting with L2 loss

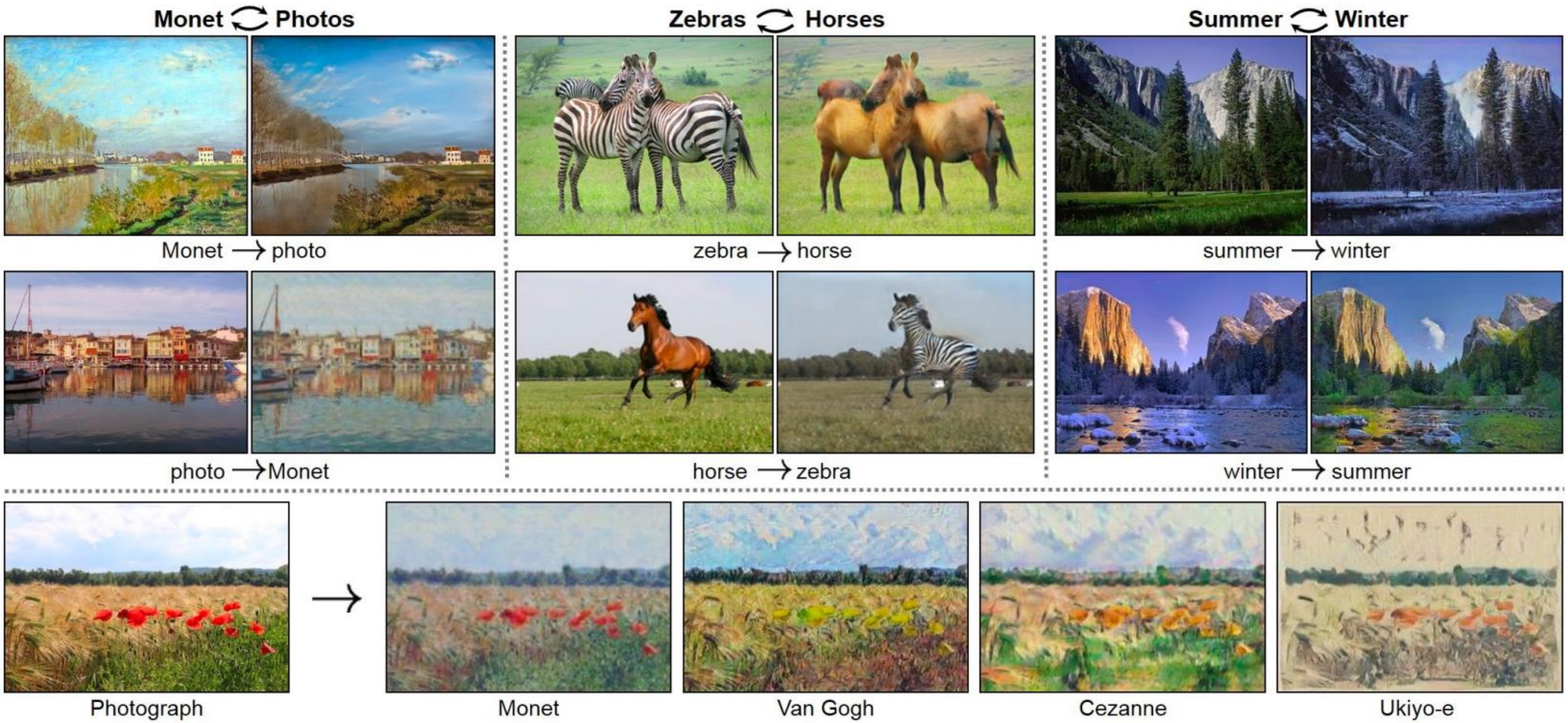
Inpainting with CGAN

Context Encoders: Feature Learning by Inpainting, D.Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, A. Efros,2016

Mixing styles from two source images

Image-to-Image Translation with GANs

CycleGAN (Image-to-Image Translation)

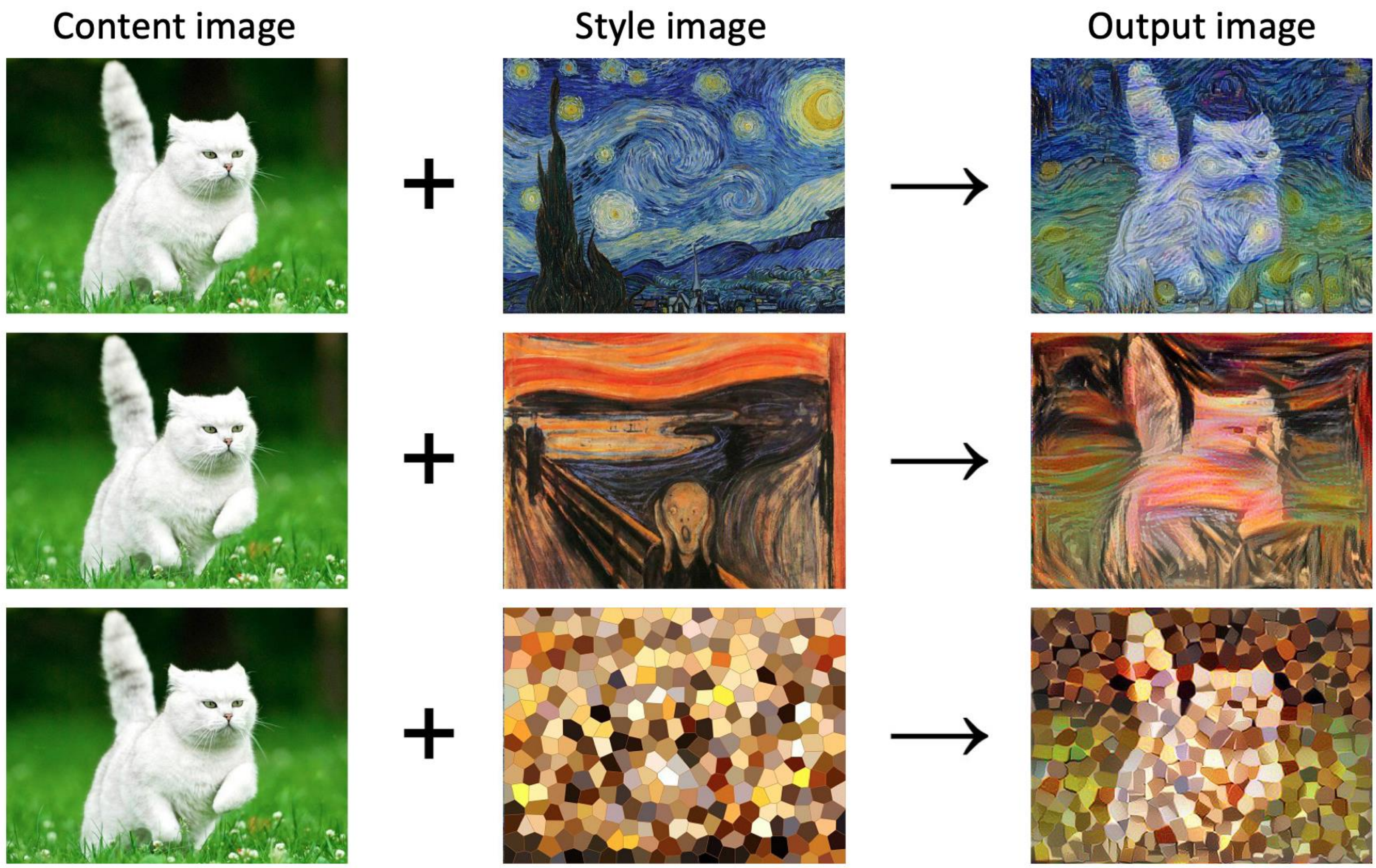


https://github.com/junyanz/CycleGAN

CycleGAN (Video-to-Video Translation)

https://github.com/junyanz/CycleGAN

Neural Style transfer



OpenAI-DALL E-2: Text-to-Image

A photo of a quaint flower shop storefront with a pastel green and clean white facade and open door and big window

Cat sipping tea and posting to twitter while sitting on a couch

A rabbit detective sitting on a park bench and reading a newspaper in a victorian setting

A lion in a hoodie hacking on a laptop

Teddy bears shopping for groceries in ancient Egypt

Teddy bears working on new AI research on the moon in the 1980s

OpenAI-Sora: Text-to-Video

Sora (2024)

Create real high quality videos from a text description

https://openai.com/sora

Prompt: Several giant wooly mammoths approach treading through a snowy meadow, their long wooly fur lightly blows in the wind as they walk, snow covered trees and dramatic snow capped mountains in the distance, mid afternoon light with wispy clouds and a sun high in the distance creates a warm glow, the low camera view is stunning capturing the large furry mammal with beautiful photography, depth of field.


Questions