Unit 8
AI for Multimedia

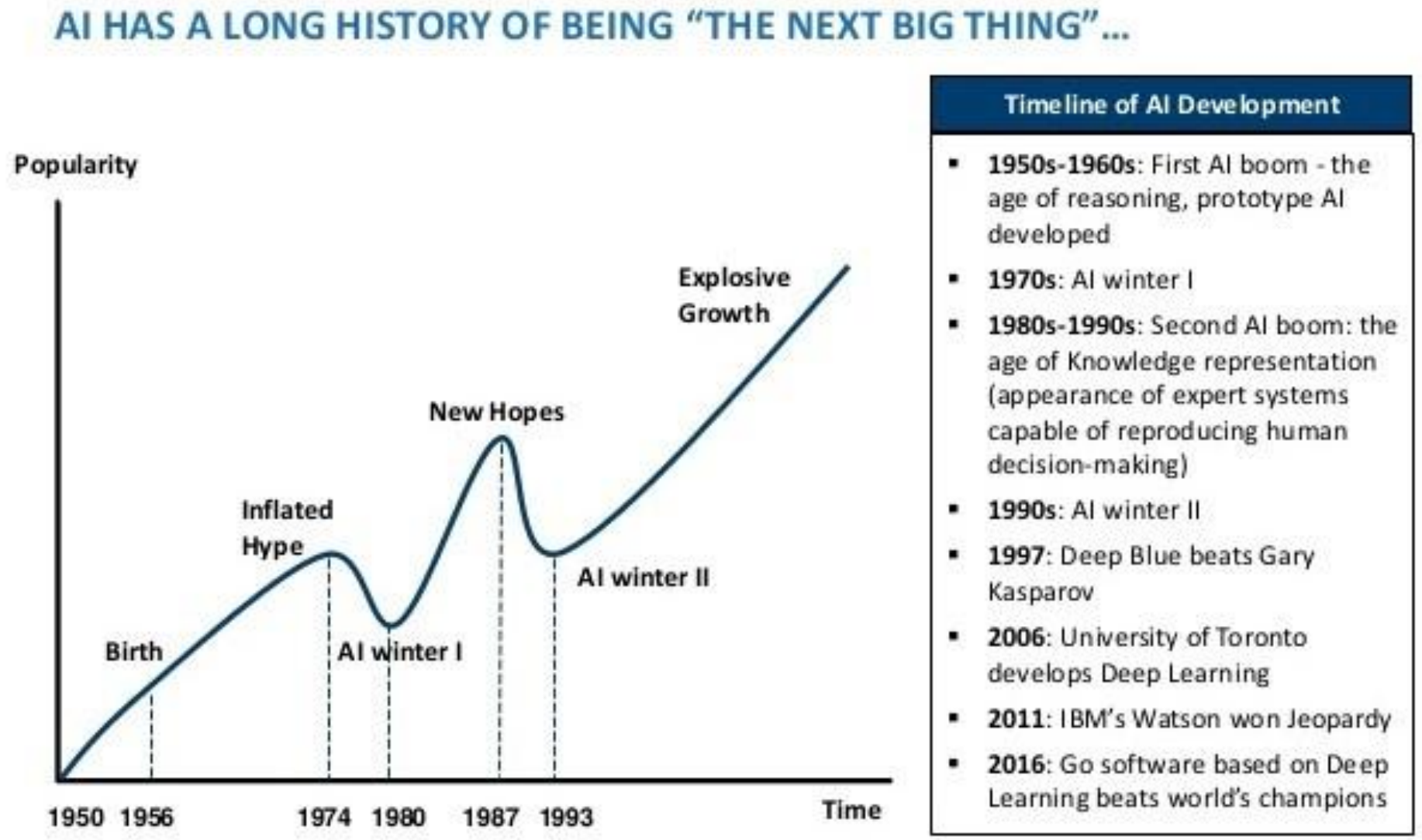
Shang-Hong Lai

Depart of Computer Science
National Tsing Hua University

Artificial Intelligence

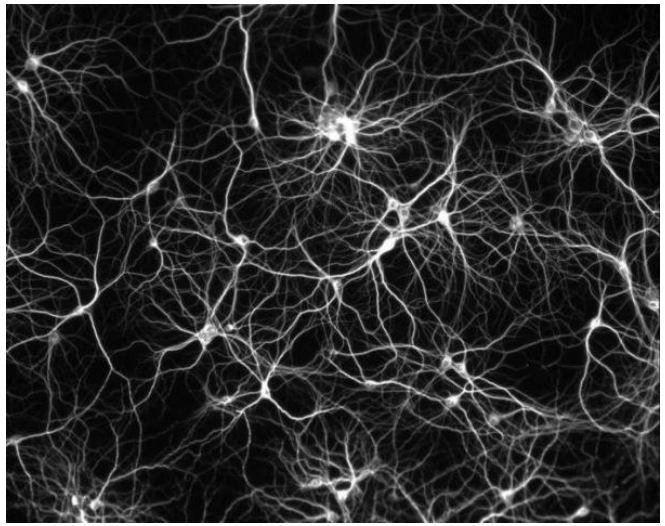
- Definition: the study of an intelligent agent that perceives its environment and take actions that maximize its chance of achieving its goals.
- Traditional goals of AI research include reasoning, knowledge representation, planning, learning, natural language processing, perception and the ability to move and manipulate objects.
- General intelligence is among the field's long-term goals. To solve these problems, AI researchers use versions of search and mathematical optimization, formal logic, artificial neural networks, and methods based on statistics, probability and economics.

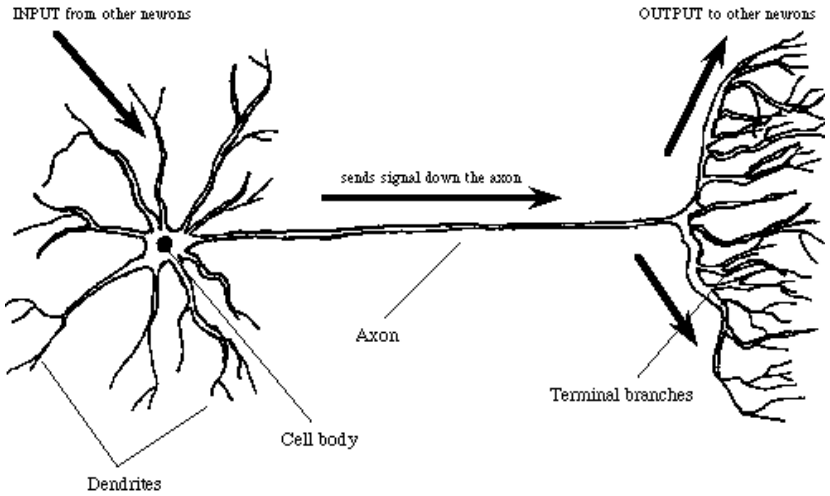
AI Winters

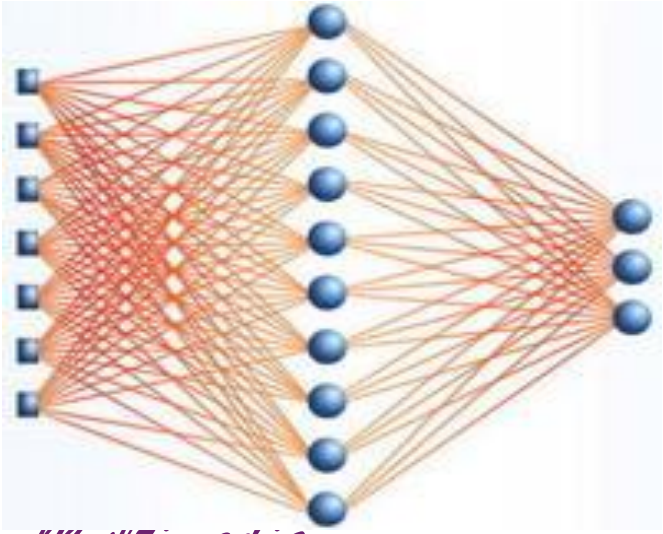


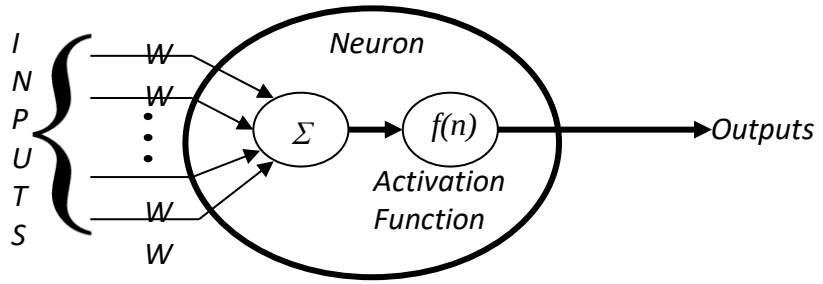
https://www.actuaries.digital/2018/09/05/history-of-ai-winters/

Artificial Neural Networks



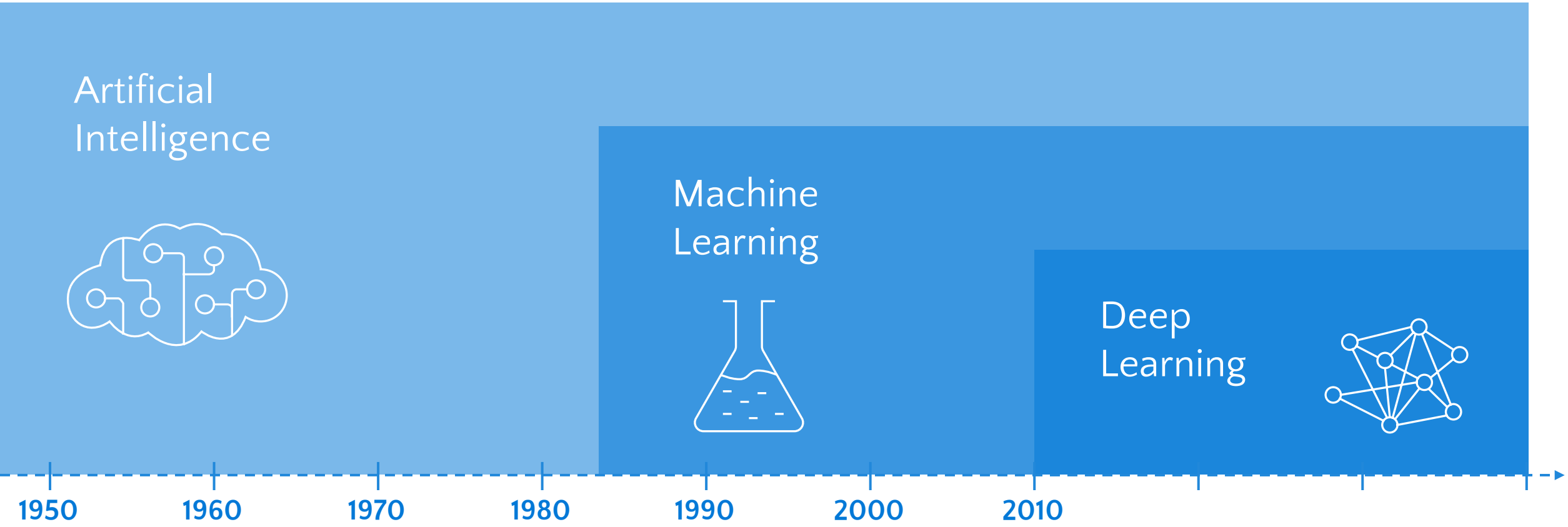


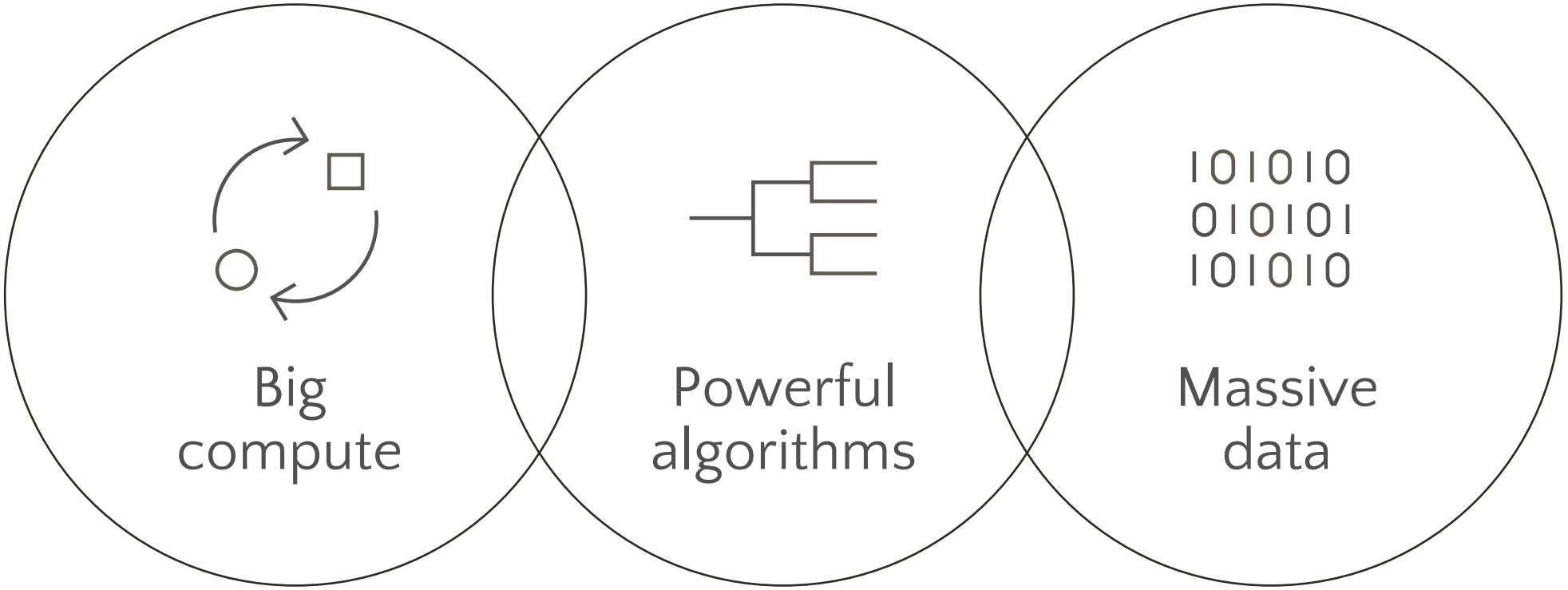




W=Weight

AI, Machine Learning and Deep Learning



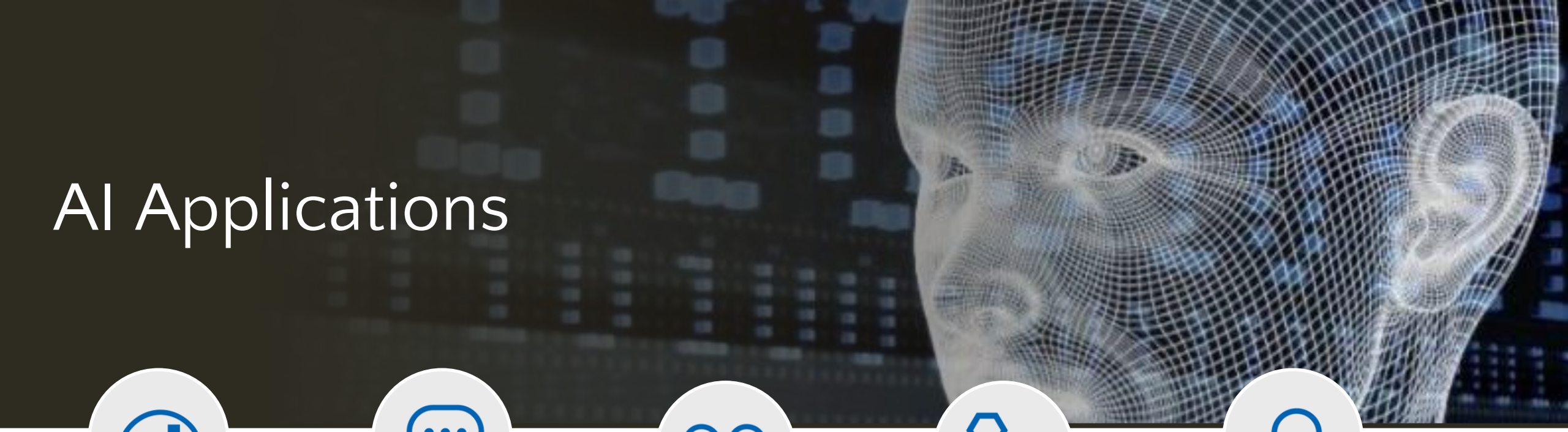


Cloud computing
GPU/TPU

CNN
ResNet
Transformer (Bert, VIT, GPT, CLIP)
RNN
LSTM

Web
Mobile camera
CCTV
IoT sensors

AI Applications



Vision

From faces to feelings, allow your apps to understand images and video

Speech

Hear and speak to your users by filtering noise, identifying speakers, and understanding intent

Language

Process text and learn how to recognize what users want

Knowledge

Tap into rich knowledge amassed from the web, academia, or your own data

Search

Access billions of web pages, images, videos, and news

A variety of real-world applications

Vision

What is in the image?

Computer Vision

Category	People; 5 faces
Adult/Racy?	False/False
Dominant colors	☐ ■ ■ ■
Accent color	

Speech

Give me directions to the nearest local branch.

Speech Recognition

Convert spoken audio to text

Convert text to spoken audio

Extract intent of user

Language

Play today's customer call recording.

Language Understanding

Natural Language Processing

```
Intent: PlayCall
Content: Customer#
DateTime.date: today
```

Now Playing

11/29/2016 Customer Call

Knowledge

Top publications in customer lifecycle trends?

Knowledge Exploration

Here are the top results:

Customer Relationship Management – 5 Key Trends for 2014 CRM

Oct 28, 2015 – Here are FIVE key **trends** in 2014 that would help marketers in rolling ... Of late, marketers are looking at **customer lifecycle** management (CLM)

Predictive Customer Lifecycle Management (CLM)

The purpose of **Customer Life-cycle** Management (CLM) is to maximize both customer retention and .... Predictive trend analysis provides business visibility.

Trends 2016: The Future of Customer Service

Jan 5, 2016 – The top 10 **customer** service **trends** for 2016 that .... North American Consumer

Language Around Customer Lifecycles in the Banking Industry

View PDF

Search

Search for 'fraud prevention'

News Search

Here is what I found:

Information Communications Media Market News

It also investigates the top three expected **Fraud** Detection and **Prevention** programs, in terms of demand in key markets...

The Big Question: In-House or Outsourced Fraud Protection?

First, let's point out that there is not one absolute answer—there are "pros" and "cons" to each. Those who favor in-house...

How to Protect Your Business from Online Fraud this Holiday Season

Michael heads fraud prevention tool. Online and mobile shopping are expected to continue growing apace...

Machine Learning

- Human learned from past experiences to improve future performance.
- For a machine, experiences come in the form of data.
- What does it mean to improve performance?
  - Learning is guided by an objective, associated with a particular notion of loss to be minimized (or, equivalently, gain to be maximized).
- Why machine learning?
  - We need computers to make informed decisions on new, unseen data.
  - Often it is too difficult to design a set of rules “by hand”.
  - Machine learning system is trained from a training dataset to automatically extract relevant information from new data and make sound decisions.

Machine Learning – Math

Problem Formulation:

Given a set of data samples (X_i) with associated labels (C_i), denoted by $\{ (X_i, C_i) \mid i=1, \ldots, N\}$. To find a function

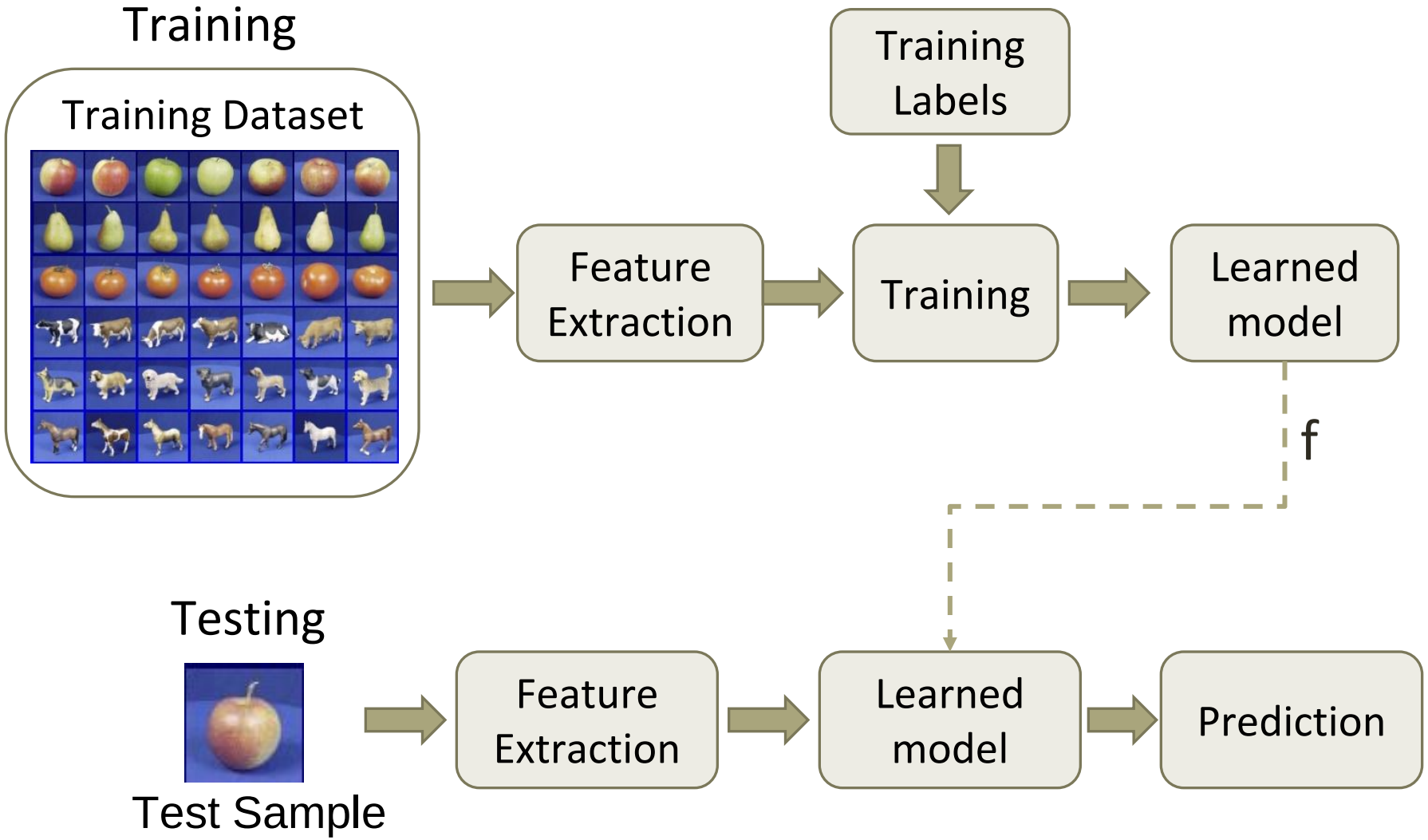
$$f(X) = C$$

such that $f(X_i) \rightarrow C_i$ for all training data as best as possible.

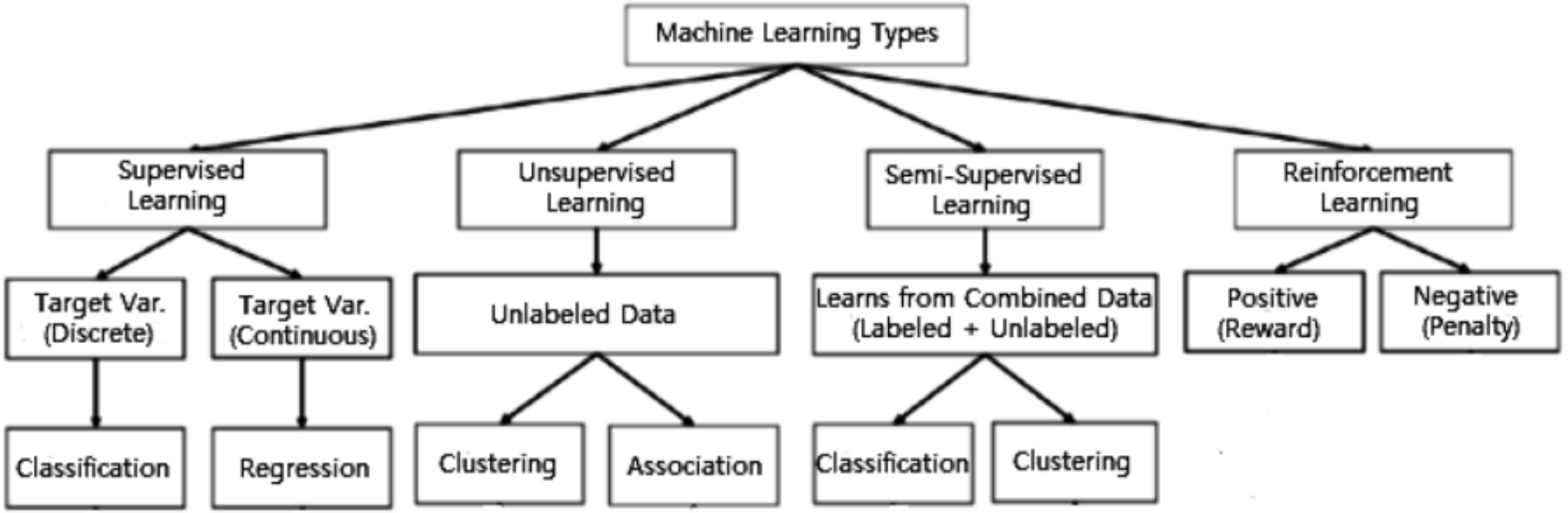
Machine learning is to find a suitable function f that is determined from the training data, and it can be applied to unseen data to predict the outcome.

f : prediction function/model (classifier, regression function)

Traditional Machine Learning System



Types of Machine Learning Techniques



I. H. Sarker, Machine Learning: Algorithms, Real-World Applications and Research Directions, SN Computer Science (2021)

Classification: Definition

Given a collection of records (*training set*)

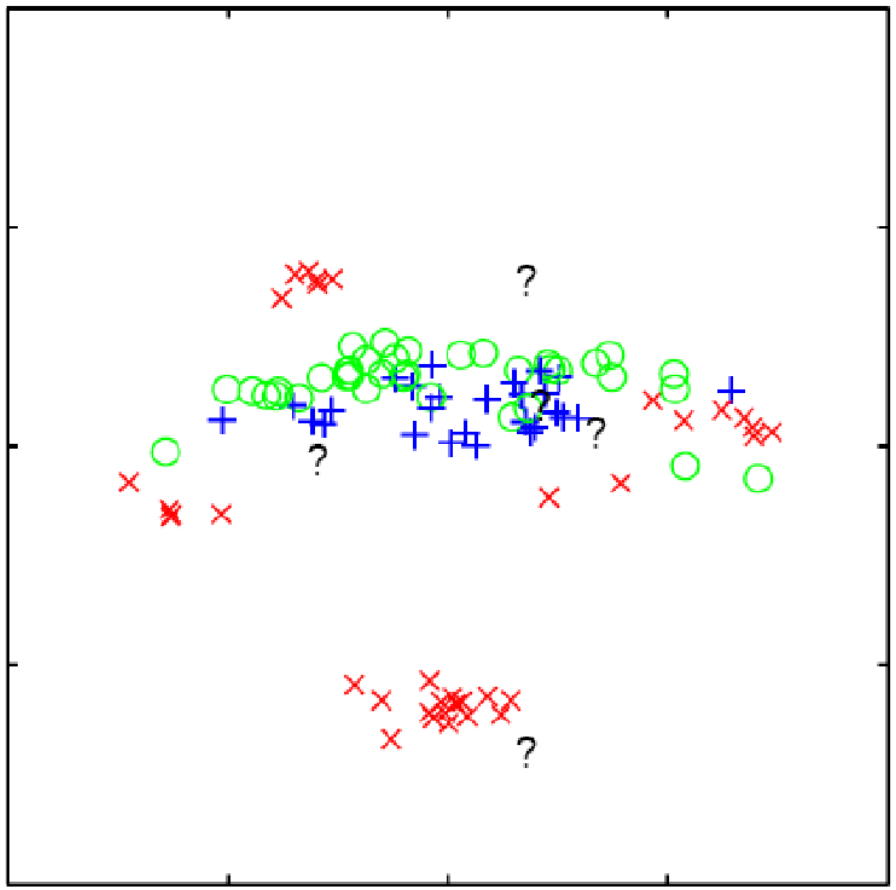
- Each record contains a set of *attributes*, one of the attributes is the *class*.

Find a *model* for class attribute as a function of the values of other attributes.

Goal: the model can be used to predict the classes of previously unseen records as accurately as possible.

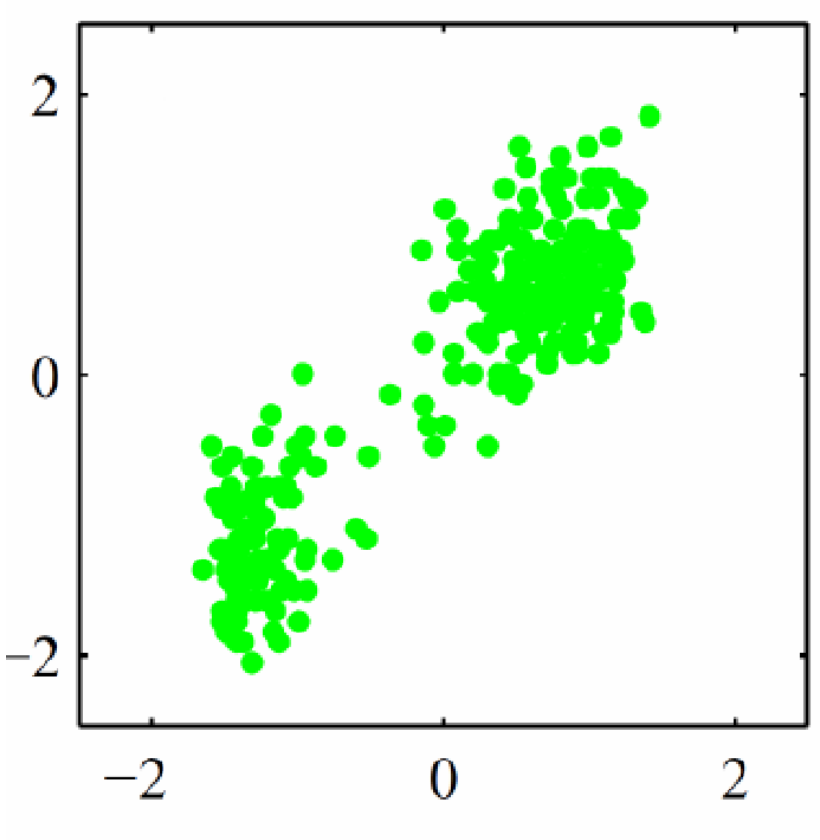
- A *test set* is used to evaluate the accuracy of the model.
- Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

Supervised Classification



- Training: Colored points are given and used for training (dividing feature space)
- Testing: Labels for test samples (question marks) can be inferred

Clustering (Unsupervised)



- Assign each data point to a cluster
- Data here suggests two clusters, some methods determine the number of clusters from the data, others use a predefined number of clusters

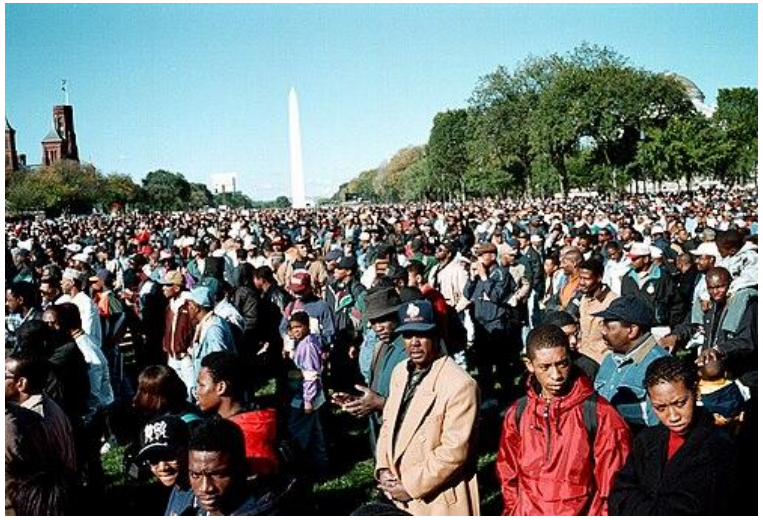
K-means clustering is a popular clustering technique.

Regression

Predict a value of a given continuous valued variable based on the values of other variables, assuming a linear or nonlinear model of dependency.

Examples:

- Predicting sales amounts of new product based on advertising expenditure.
- Predicting wind velocities as a function of temperature, humidity, air pressure, etc.
- Time series prediction of stock market indices.
- Predicting car speed from a video
- People counting from images or surveillance video



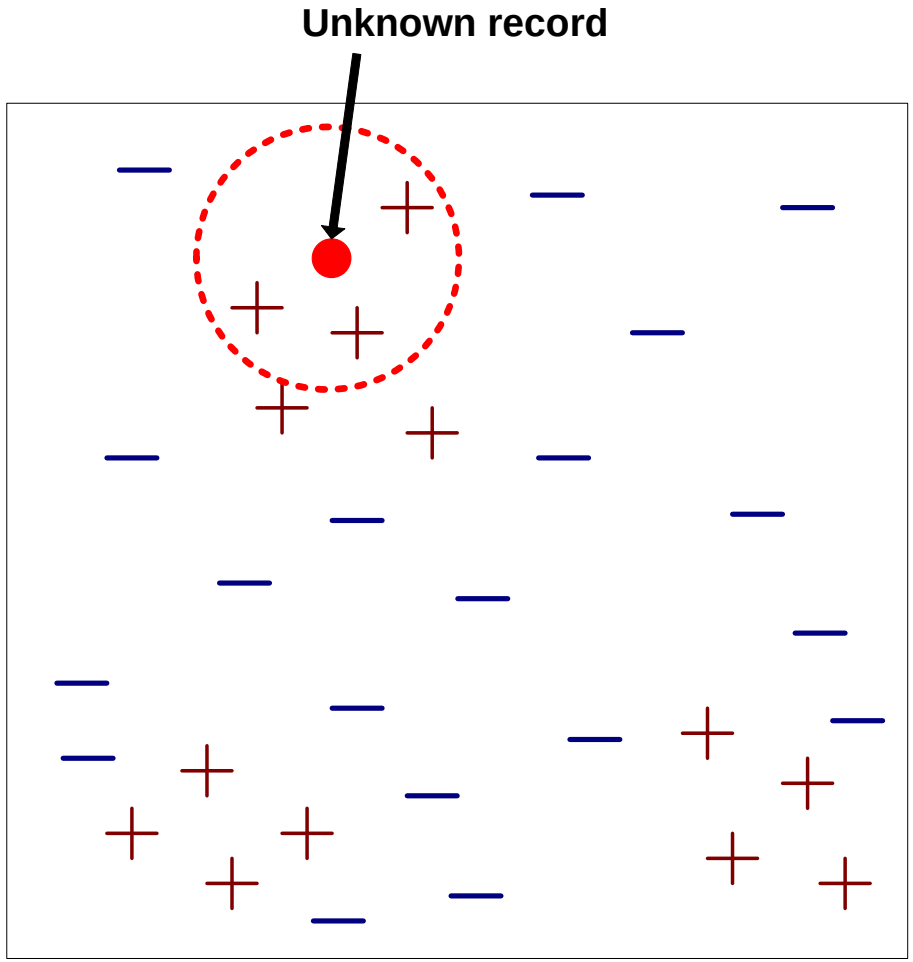
Nearest-Neighbor Classifiers

K-NN classifier requires three things

- The set of stored records (labeled training data)
- Distance Metric to compute distance between records
- The value of *k*, the number of nearest neighbors to retrieve

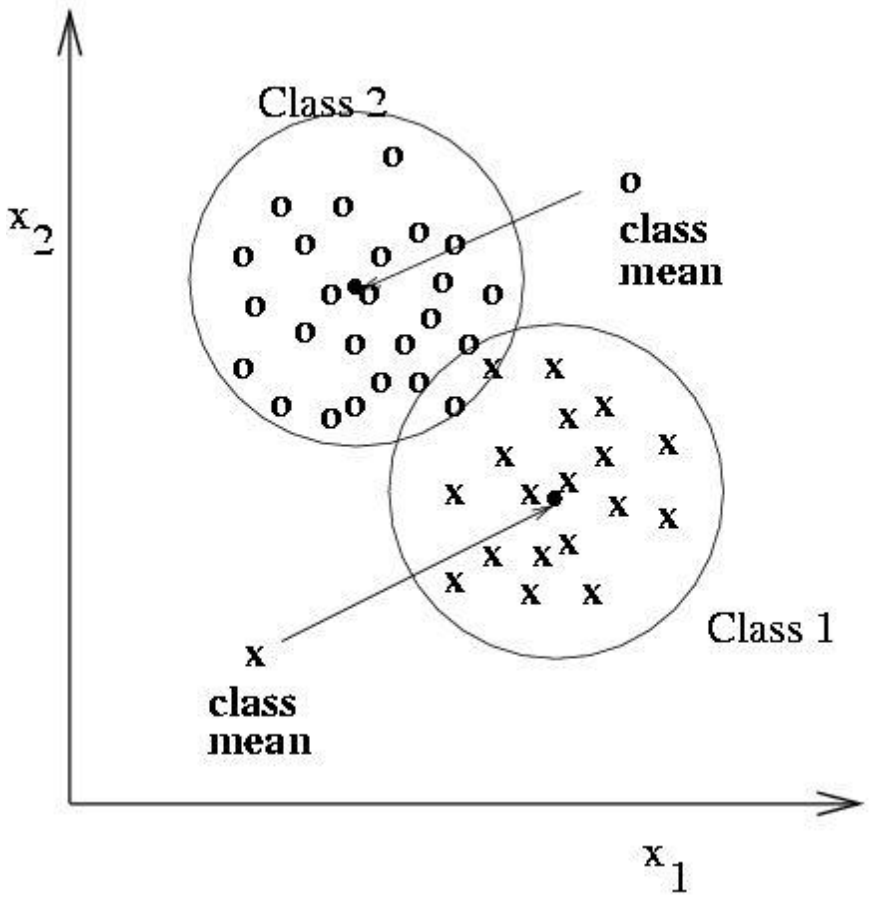
To classify an unknown record:

- Compute distance to other training records
- Identify *k* nearest neighbors
- Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)



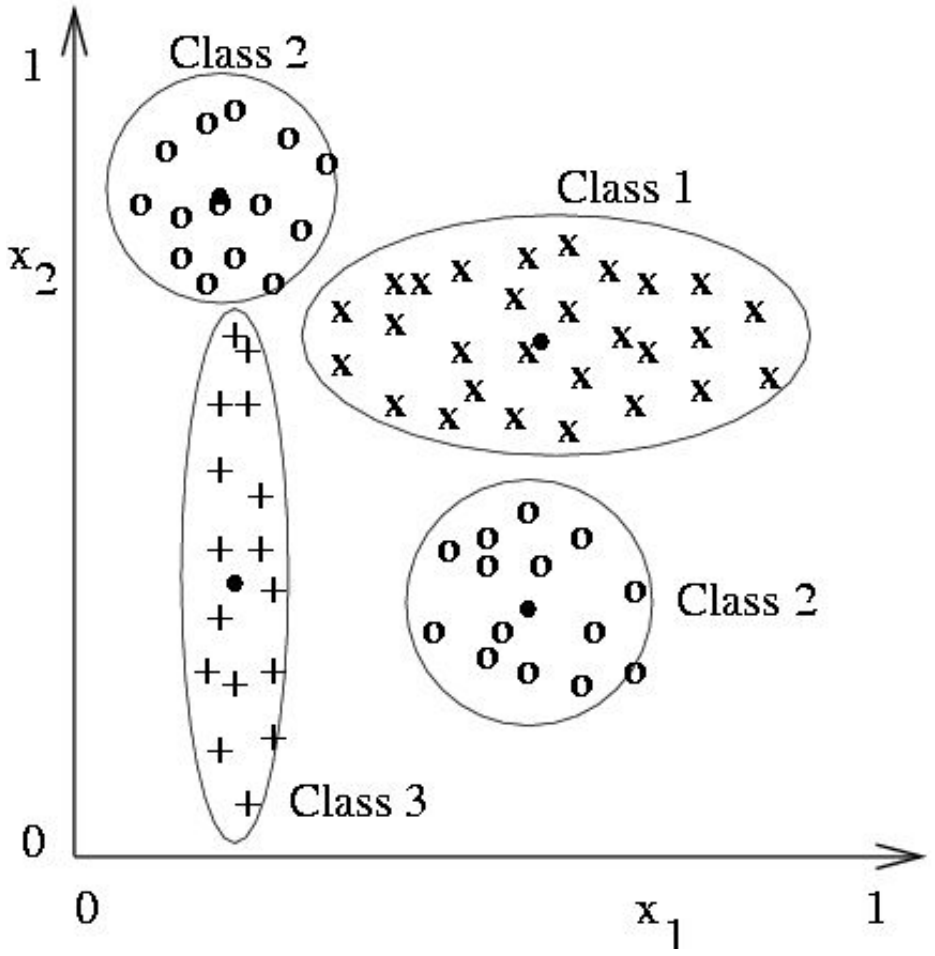
Classification Using Nearest-Class-Mean

- Compute the Euclidean distance between feature vector X and the mean of each class.
- Choose closest class, if close enough (reject otherwise)
- Distance measure between feature vectors is very crucial



Problem with Nearest –Class-Mean Classifier

- Nearest mean might yield poor results with complex structure
- Class 2 has two modes
- If modes are detected, two subclass mean vectors can be used



Bayesian Classifiers

Approach:

- compute the posterior probability P(C | A_1, A_2, …, A_n) for all values of C using the Bayes theorem

$$P(C \mid A_1 A_2 \ldots A_n) = \frac{P(A_1 A_2 \ldots A_n \mid C) P(C)}{P(A_1 A_2 \ldots A_n)}$$

- Choose value of C that maximizes

  P(C | A_1, A_2, …, A_n)

- Equivalent to choosing value of C that maximizes

  P(A_1, A_2, …, A_n|C) P(C)

How to estimate P(A_1, A_2, …, A_n | C)?

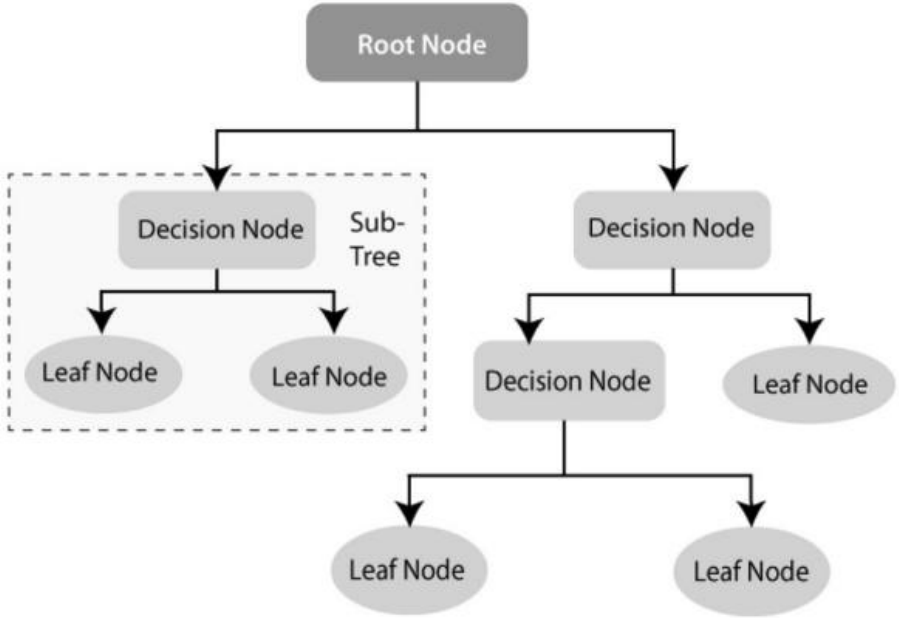
Naïve Bayes Classifier

Assume independence among attributes A_i when class is given:

- $P(A_1, A_2, \ldots, A_n \mid C) = P(A_1 \mid C_j)\, P(A_2 \mid C_j) \ldots P(A_n \mid C_j)$
- Can estimate $P(A_i \mid C_j)$ for all A_i and C_j.
- New sample data is classified to C_j if $P(C_j) \prod P(A_i \mid C_j)$ is maximal.

Decision Tree (DT)

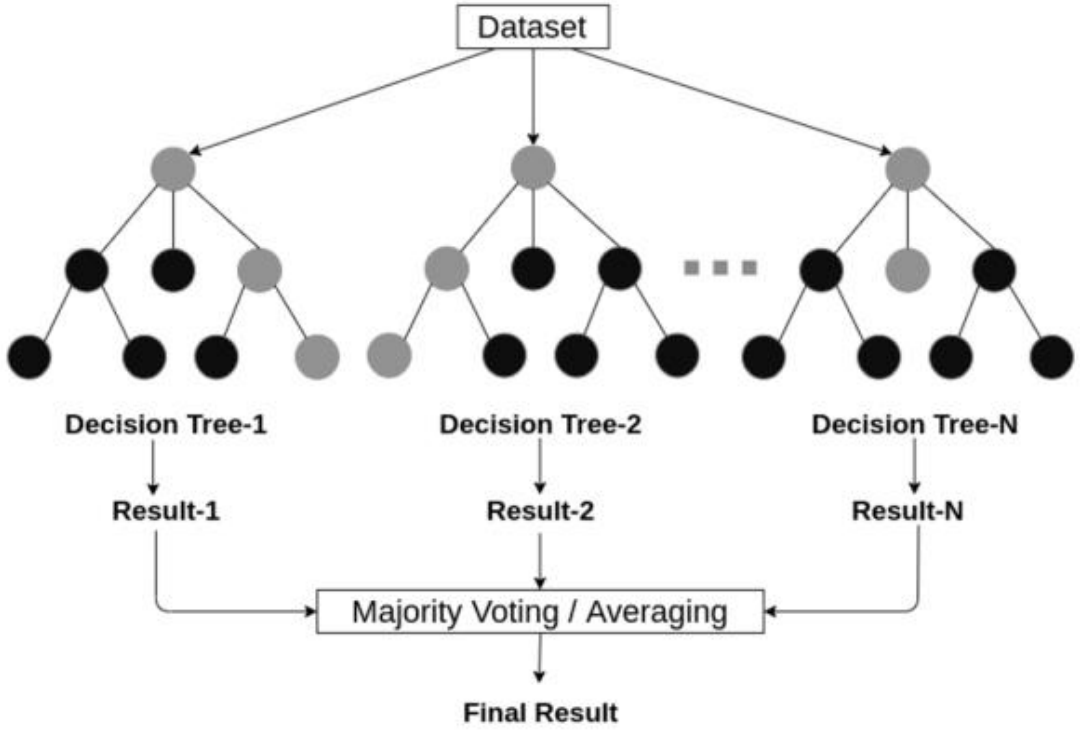
- DT is a flowchart-like structure in which each internal node represents a "test" on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label.
- The paths from root to leaf represent classification rules.
- non-parametric supervised learning method
- Can be used for classification and regression
- ID3, C4.5, and CART are well known algorithms for constructing DT from data



I. H. Sarker, Machine Learning: Algorithms, Real-World Applications and Research Directions, SN Computer Science (2021)

Random Forest (RF)

- an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time.
- RF learning model with multiple decision trees is typically more accurate than a single decision tree based model.



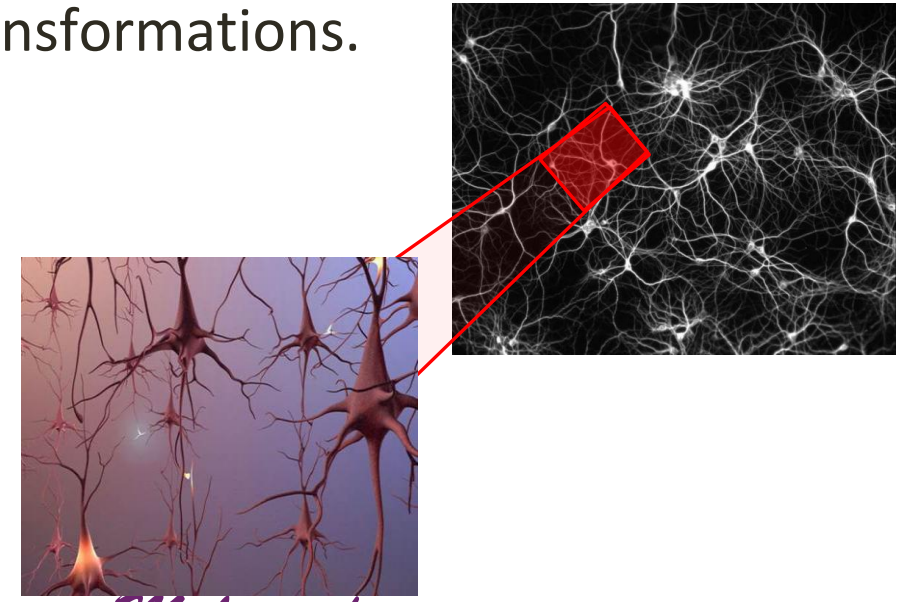
I. H. Sarker, Machine Learning: Algorithms, Real-World Applications and Research Directions, SN Computer Science (2021)

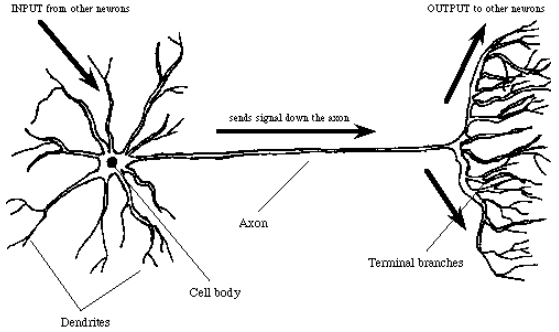
DEEP LEARNING

Deep Learning

From Wikipedia:

- **Deep learning** (**deep structured learning**, **hierarchical learning** or **deep machine learning**) is a branch of machine learning based on a set of algorithms that attempt to model high-level abstractions in data by using multiple processing layers, with complex structures or otherwise, composed of multiple non-linear transformations.

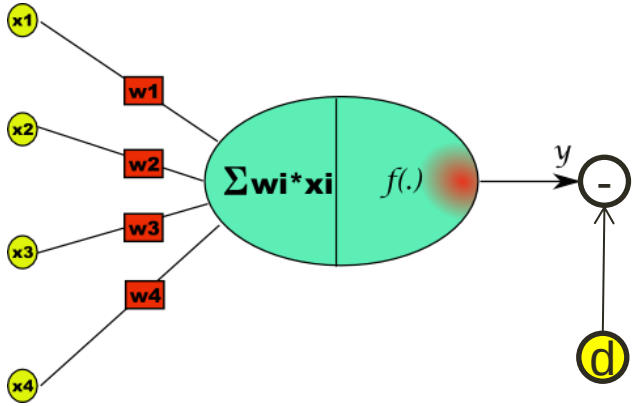




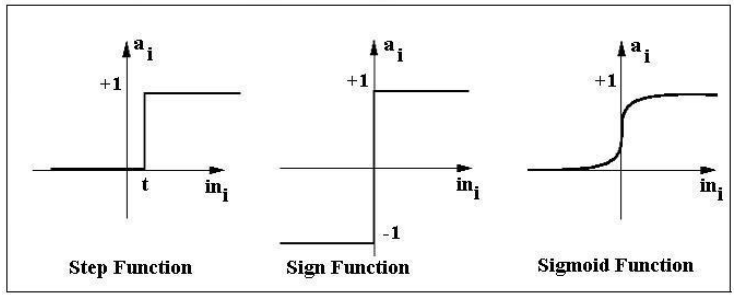
The simplest model- the Perceptron

- **Perceptron was introduced in 1957 by Frank Rosenblatt.**

Perceptron:



Activation functions:



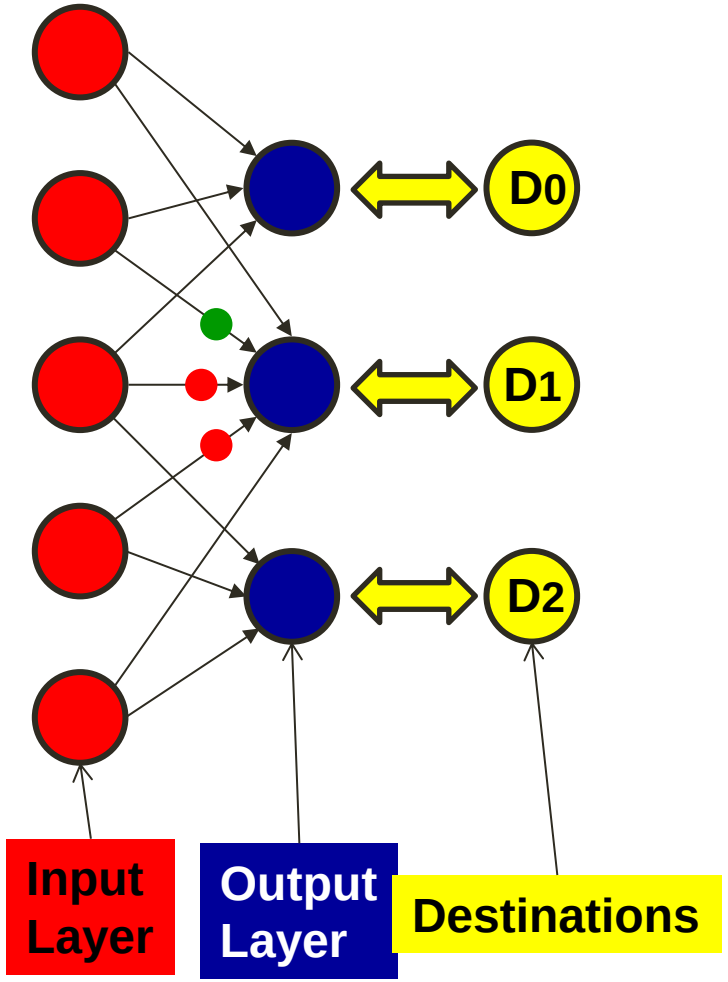
Learning:

$$y^{(t)} = f\left\{\sum_i w_i^{(t)} x_i^{(t)}\right\}$$

Update:

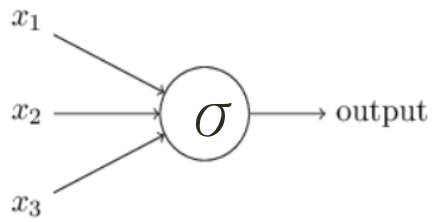
$$\Delta w_i^{(t)} = \varepsilon(d^{(t)} - y^{(t)}) x_i^{(t)}$$

$$w_i^{(t+1)} = w_i^{(t)} + \Delta w_i^{(t)}$$



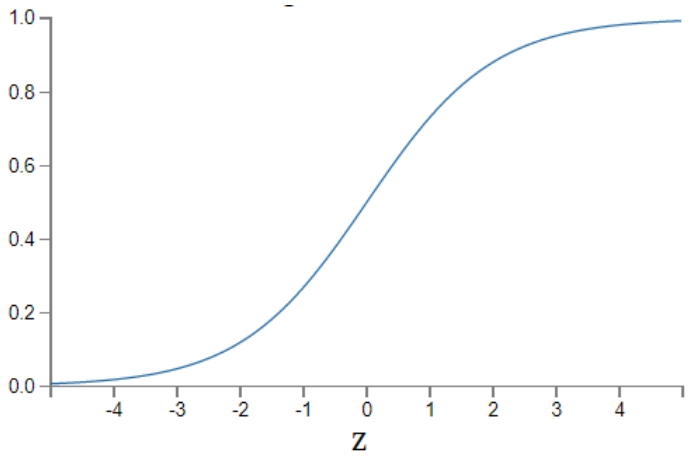
Sigmoid Function

Inputs: x_1, x_2, ..., x_n, weights: w_1, w_2, ..., w_n and bias b

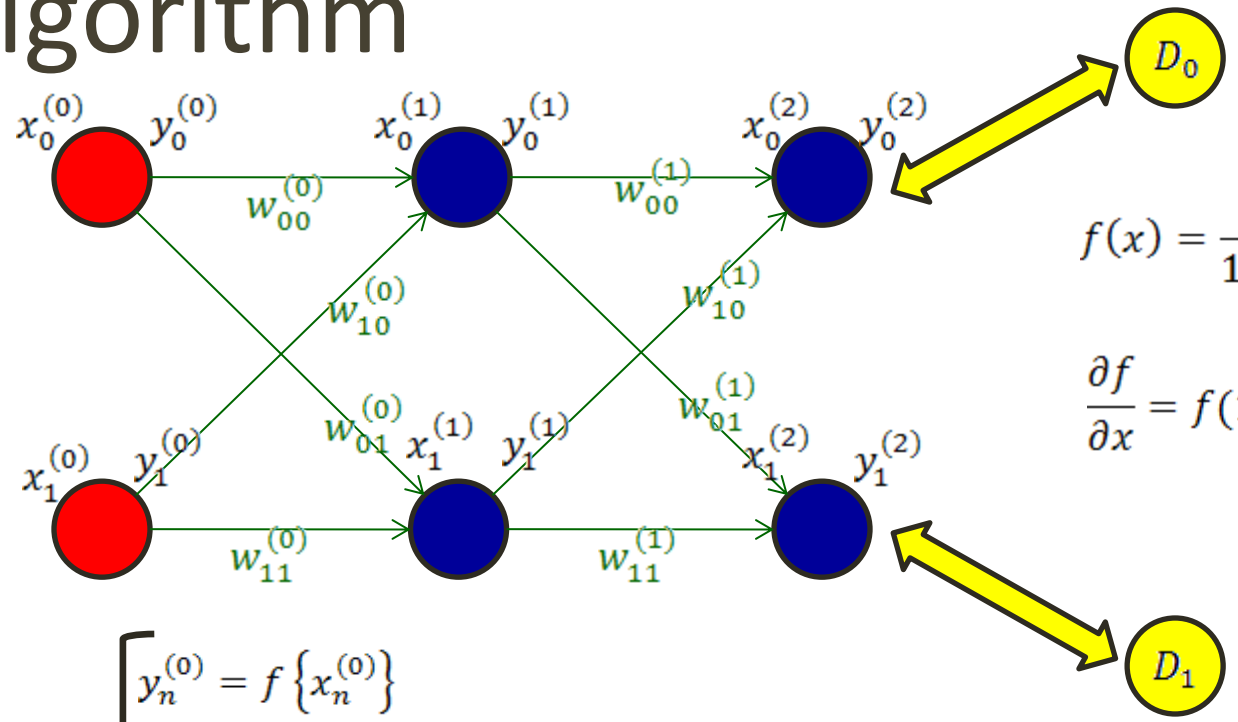


Sigmoid function σ(w·x+b)

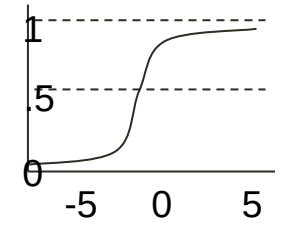
$$\frac{1}{1+\exp(-\sum_j w_j x_j - b)}.$$



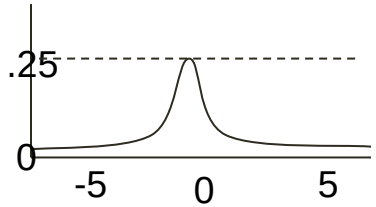
BP-algorithm



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



$$\frac{\partial f}{\partial x} = f(1-f)$$



Activations

$$\begin{cases} y_n^{(0)} = f\left\{x_n^{(0)}\right\} \\ y_n^{(1)} = f\left\{\sum_{i=0}^{1} y_i^{(0)} w_{in}^{(0)}\right\} \\ y_n^{(2)} = f\left\{\sum_{i=0}^{1} y_i^{(1)} w_{in}^{(1)}\right\} \end{cases}$$

Errors

$$\begin{cases} \frac{\partial E}{\partial w_{ij}^{(1)}} = -2\left(D_j - y_j^{(2)}\right) f\left\{x_j^{(2)}\right\}\left(1 - f\left\{x_j^{(2)}\right\}\right) y_i^{(1)} \stackrel{\text{def}}{=} \delta_j^{(1)} y_i^{(1)} \\ \frac{\partial E}{\partial w_{ij}^{(0)}} = f\left\{x_j^{(1)}\right\}\left(1 - f\left\{x_j^{(1)}\right\}\right) \sum_{n=0}^{1} \delta_n^{(1)} w_{jn}^{(1)} x_i^{(0)} \stackrel{\text{def}}{=} \delta_j^{(0)} x_i^{(0)} \end{cases}$$

The error: $E = \sum_{n=0}^{1} (D_n - y_n^{(2)})^2$

Update Weights: $w_{ij}^{(k)} = w_{ij}^{(k)} + \varepsilon \frac{\partial E}{\partial w_{ij}^{(k)}}$

Update

$$\begin{cases} w_{ij}^{(1)} = w_{ij}^{(1)} + \varepsilon \delta_j^{(1)} y_i^{(1)} \\ w_{ij}^{(0)} = w_{ij}^{(0)} + \varepsilon \delta_j^{(0)} x_i^{(0)} \end{cases}$$

CNN (Convolutional Neural Networks)

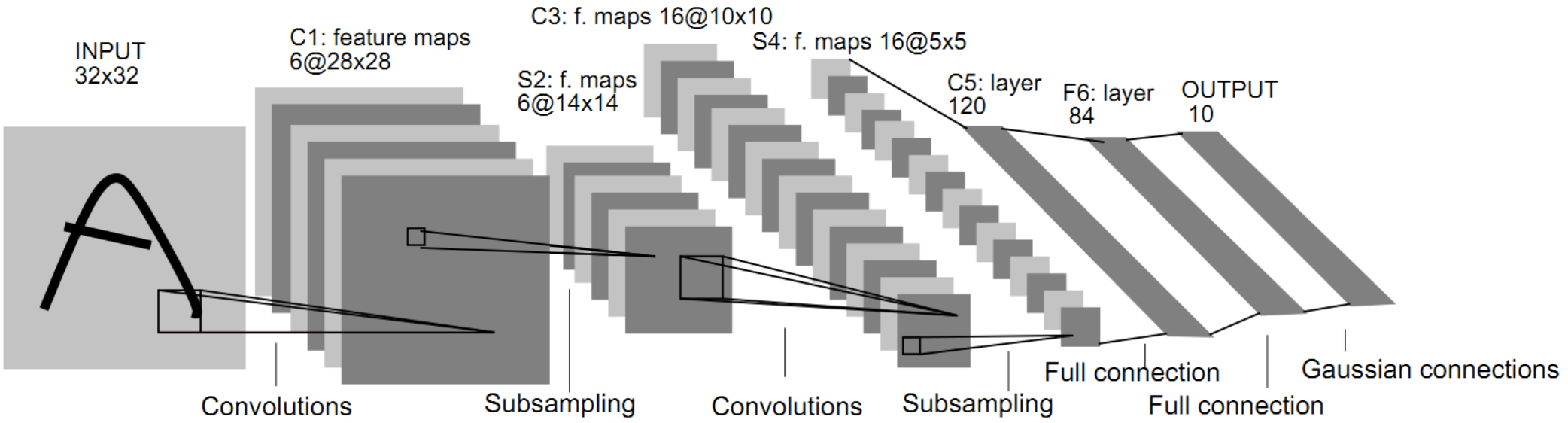


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

- The CNN architecture is LeNet used to classify hand-written digits.
- CNN has been well studied and successfully used in recognition and general object detection problems.

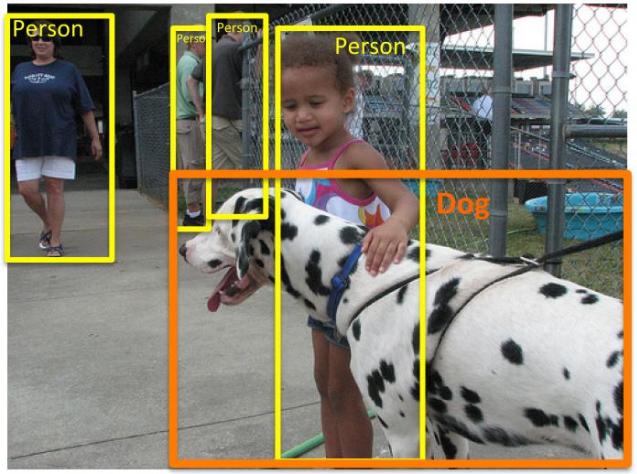
Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998

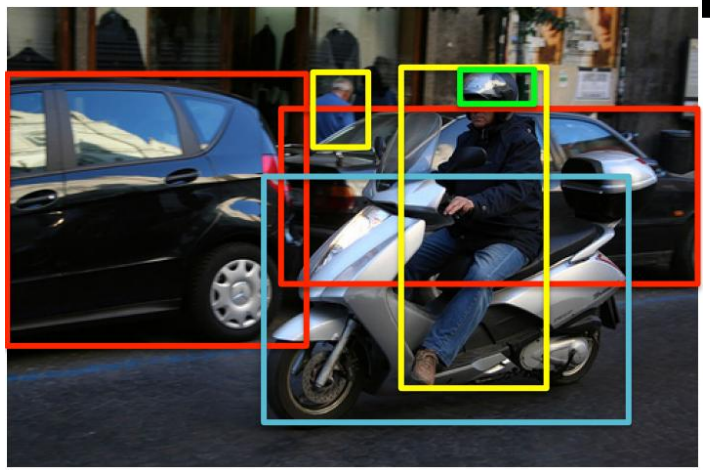
IMAGENET Large Scale Visual Recognition Challenge (ILSVRC)

ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) has been held each year since 2010.

200 object classes	**456,567 images**	**DET**
1000 object classes	**1,431,167 images**	**CLS-LOC**

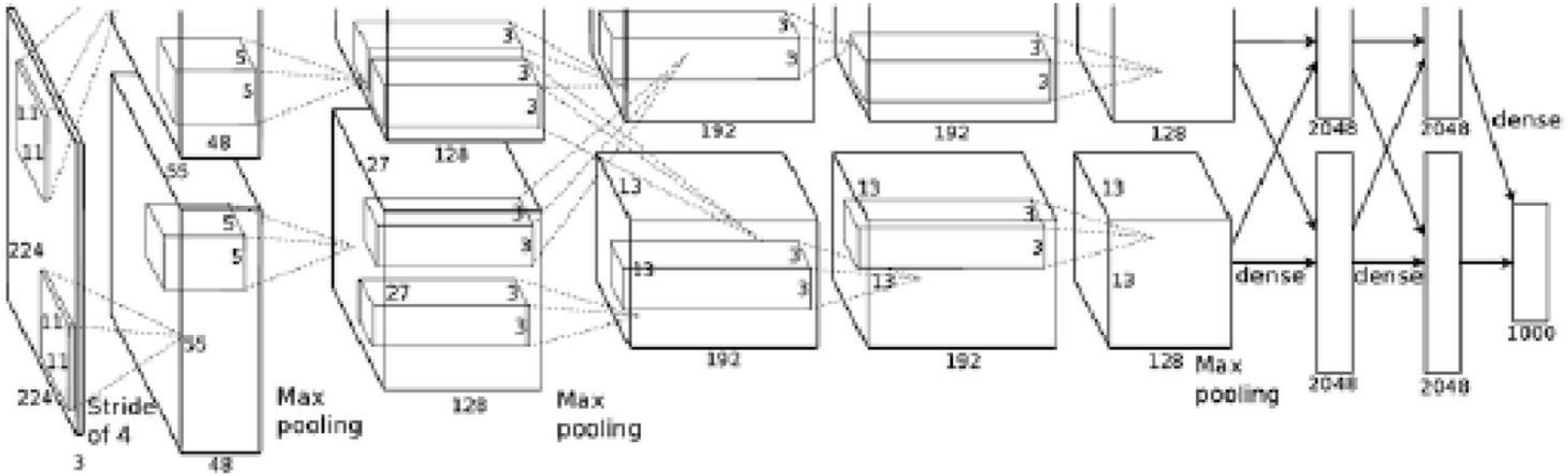
Person
Car
Motorcycle
Helmet

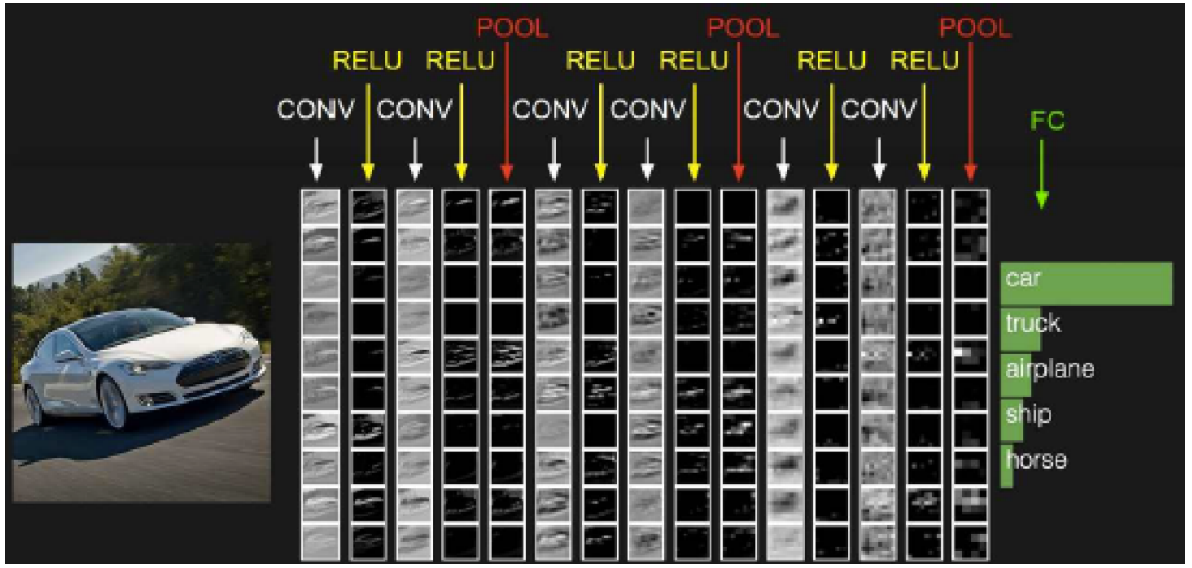




http://image-net.org/challenges/LSVRC/

AlexNet (Krizhevsky, Sutskever, Hinton 2012)





VGGNet

Small filters, Deeper networks

8 layers (AlexNet)

-> 16 - 19 layers (VGG16Net)

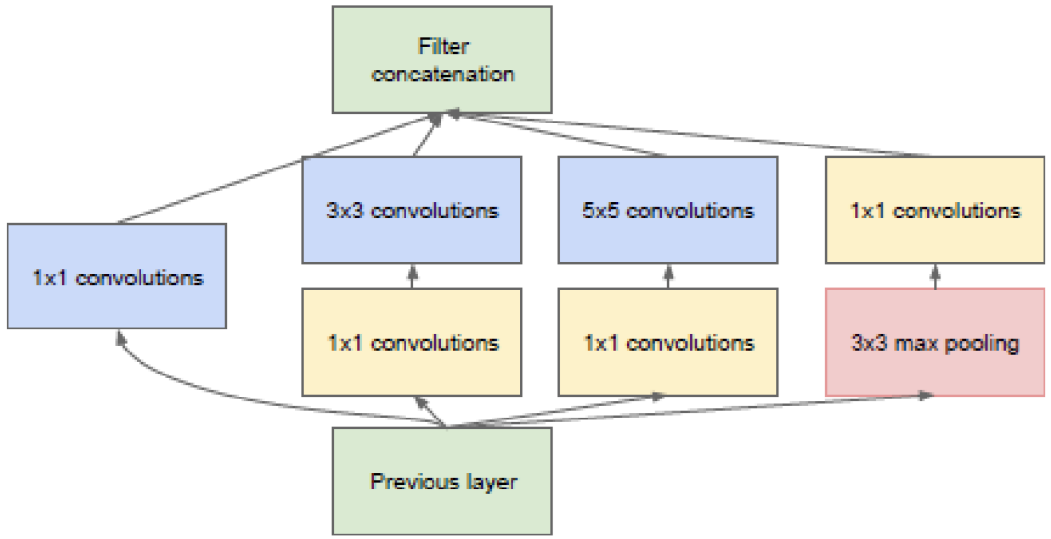
Only 3x3 CONV stride 1, pad 1

and 2x2 MAX POOL stride 2

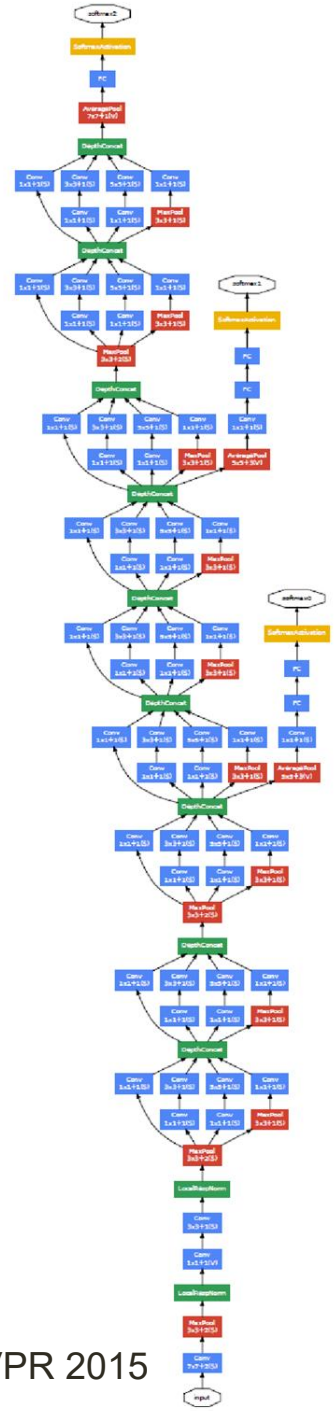
11.7% top 5 error in ILSVRC'13

-> 7.3% top 5 error in ILSVRC'14

GoogLeNet

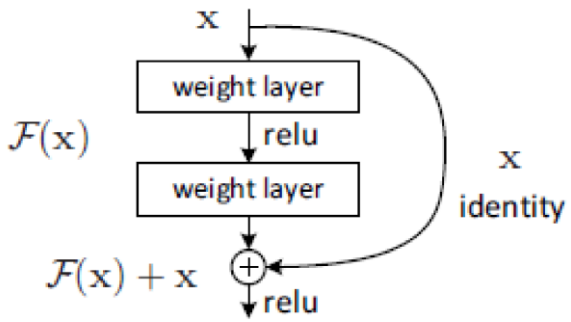


Inception Module



It has 5 Million or 12X fewer parameters than AlexNet.

ResNet



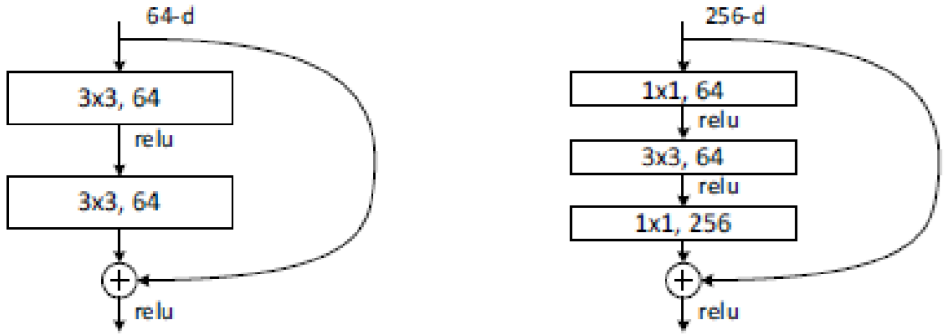
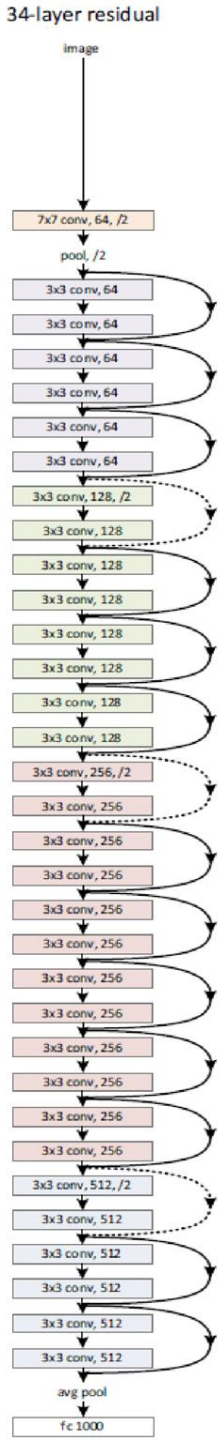


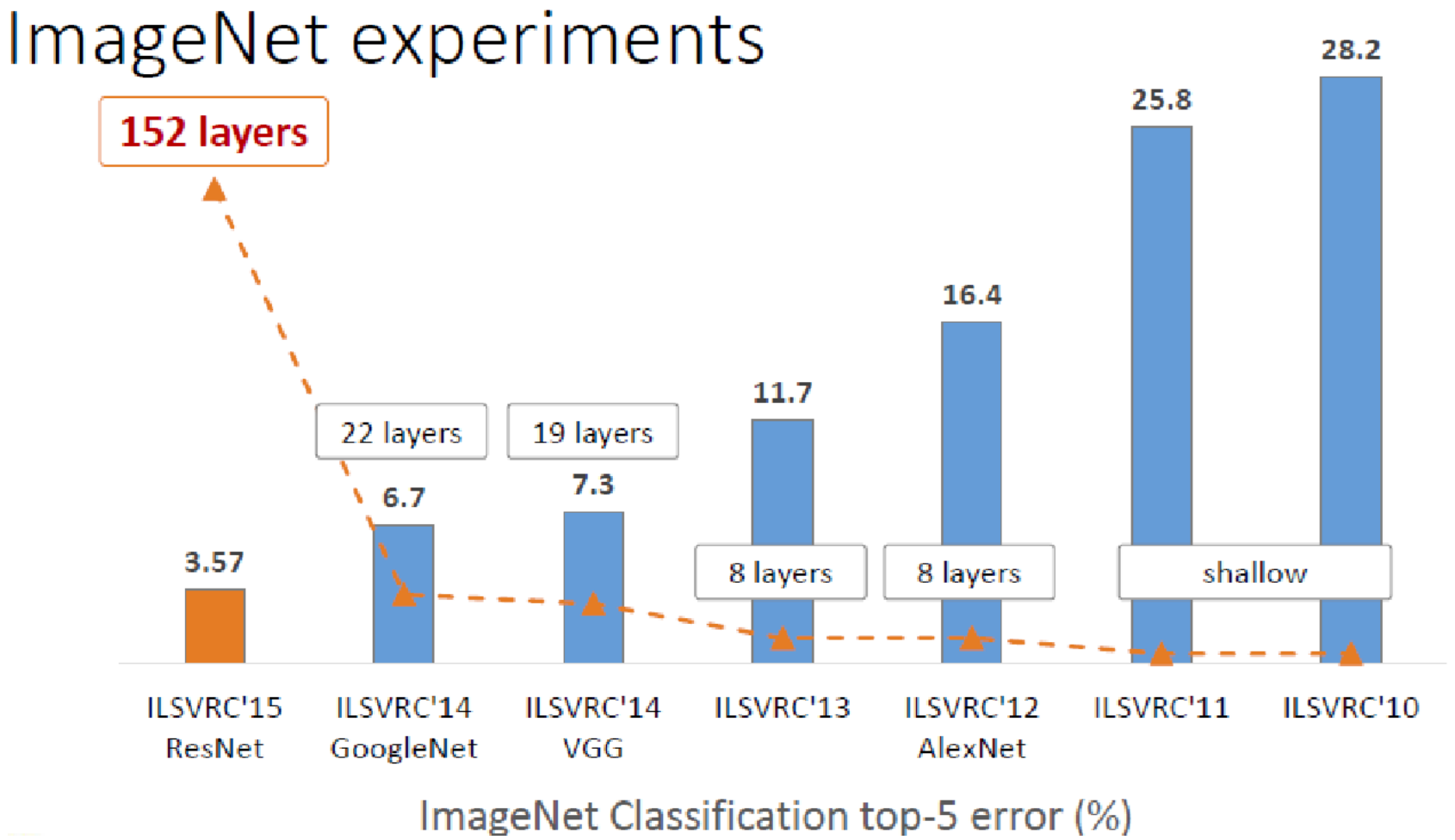
Figure 5. A deeper residual function $\mathcal{F}$ for ImageNet. Left: a building block (on 56×56 feature maps) as in Fig. 3 for ResNet-34. Right: a "bottleneck" building block for ResNet-50/101/152.

- If identity were optimal, easy to set weights as 0
- If optimal mapping is closer to identity, easier to find small fluctuations



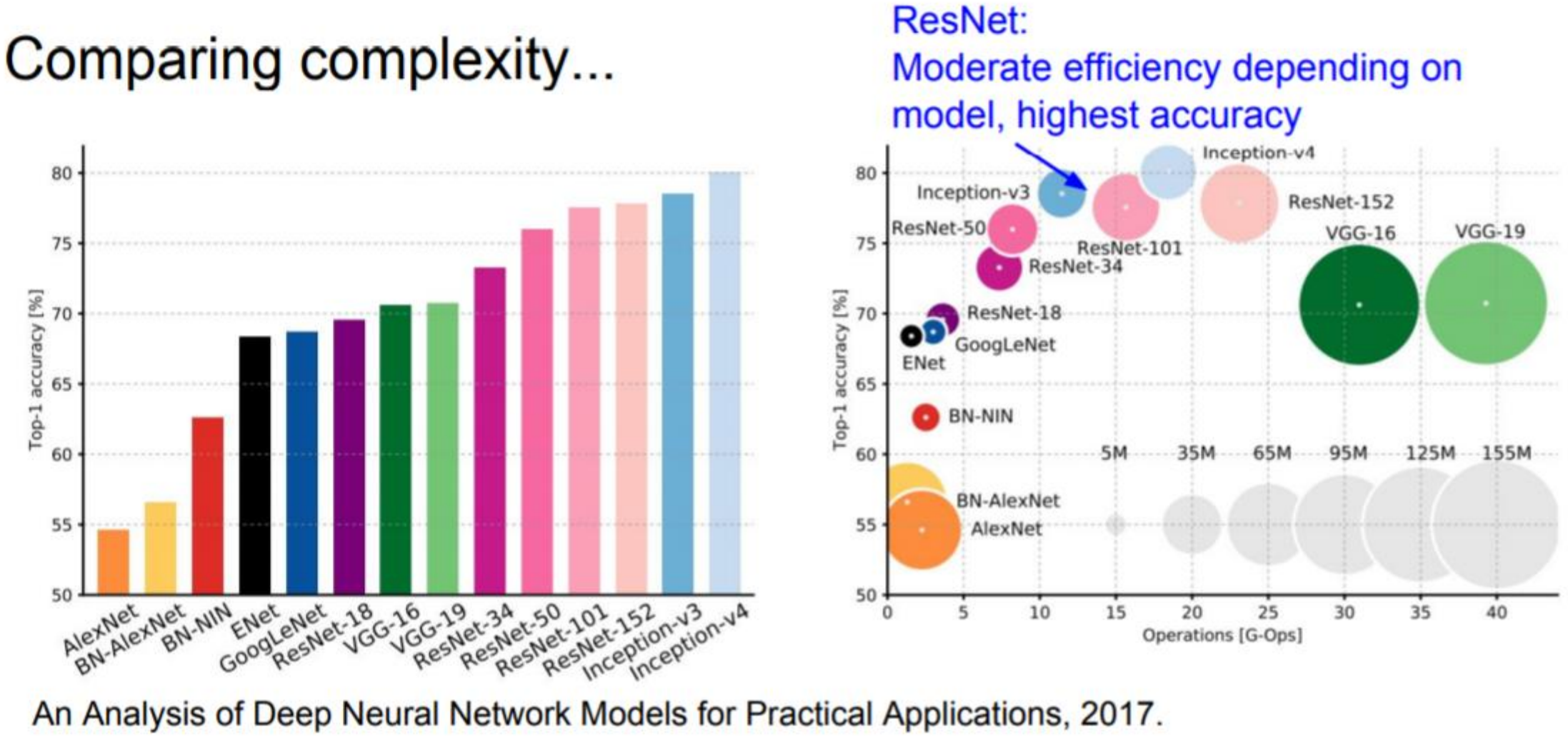
K. He et al., "Deep Residual Learning for Image Recognition", CVPR 2016

Going Deeper



K. He et al., "Deep Residual Learning for Image Recognition". arXiv 2015

Comparison of CNN Models



An Analysis of Deep Neural Network Models for Practical Applications, 2017.

Recurrent Neural Network (RNN)

A special ANN whose connections between nodes form a directed graph along a sequence, which can model temporal dynamic behavior for a time sequence.

RNN is applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition.

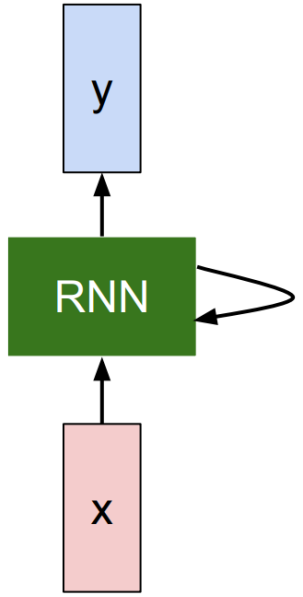
Recurrence formula is applied to the input vector x at every time instant:

h is the hidden state vector:

$$h_t = f_W(h_{t-1}, x_t)$$

$$\downarrow$$

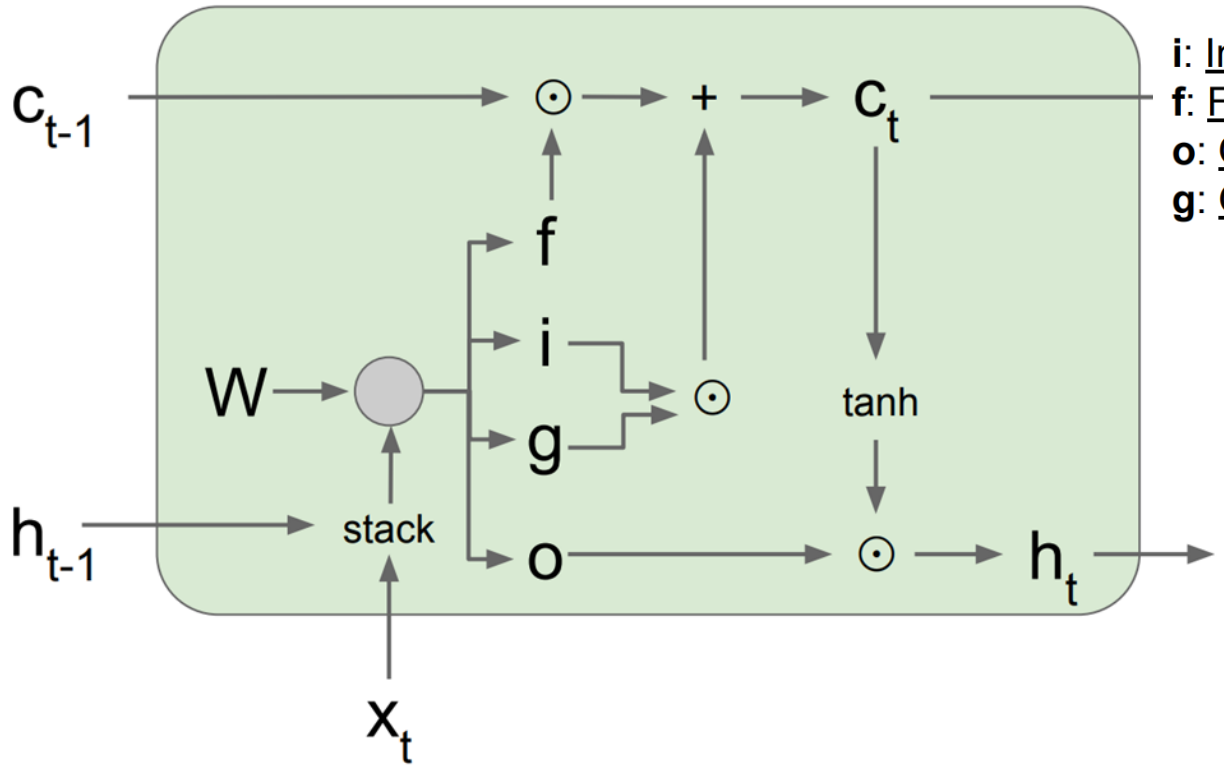
$$h_t = \sigma_h(W_h x_t + U_h h_{t-1} + b_h)$$
$$y_t = \sigma_y(W_y h_t + b_y)$$



Long Short-Term Memory (LSTM)

LSTM is normally augmented by recurrent gates called "forget" gates.

LSTM works even given long delays between significant events and can handle signals that mix low and high frequency components.



i: Input gate, whether to write to cell
f: Forget gate, Whether to erase cell
o: Output gate, How much to reveal cell
g: Gate gate (?), How much to write to cell

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

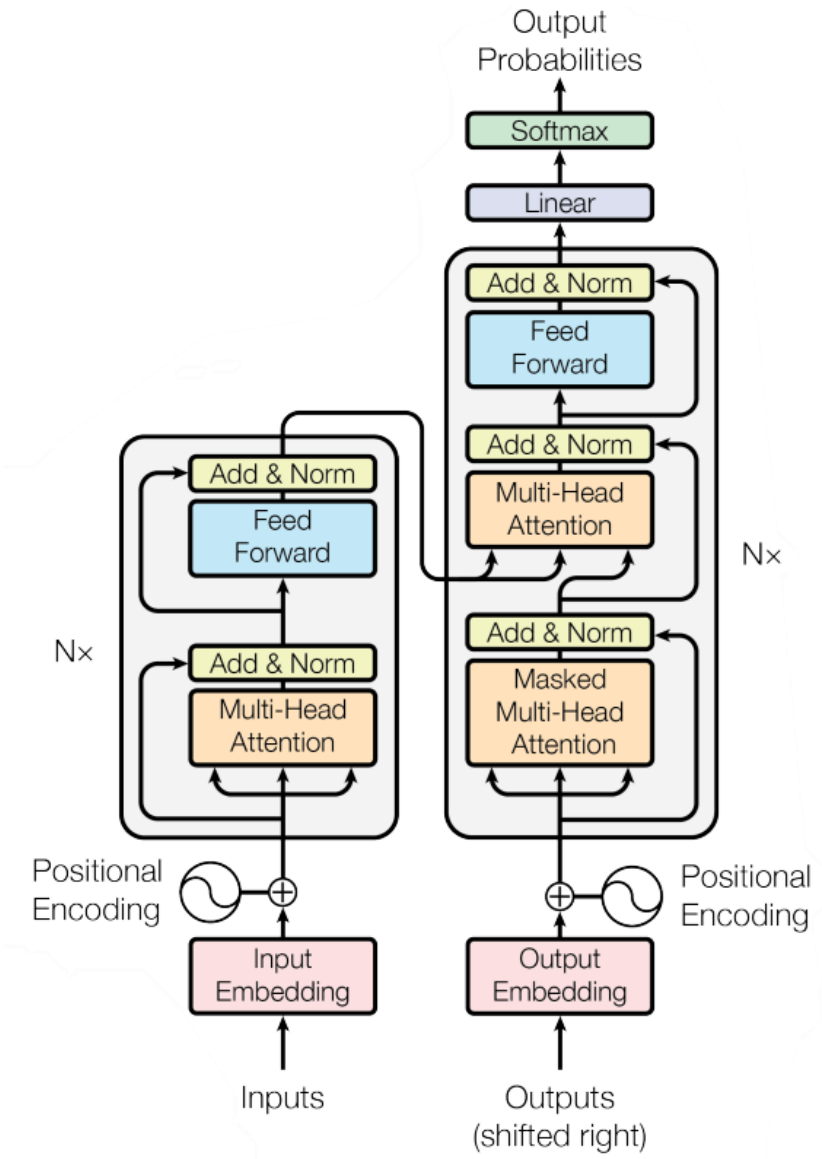
$$h_t = o \odot \tanh(c_t)$$

Vanilla Transformer

First proposed for machine translation

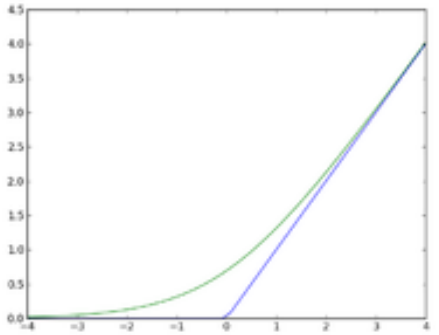
A sequence-to-sequence model consisting of encoder and decoder

Natural language processing (NLP) has seen tremendous development because of the Transformer.



Vaswani, Ashish et al. "Attention is All you Need." *ArXiv* abs/1706.03762 (2017)

Rectified Linear Units (ReLU)



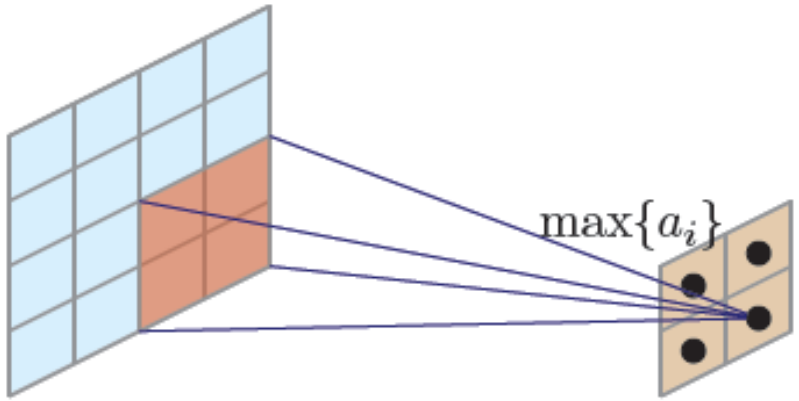
More efficient gradient propagation, derivative is 0 or constant, just fold into learning rate

More efficient computation: Only comparison, addition and multiplication.

- Leaky ReLU $f(x) = x$ if > 0 else ax where $0 \leq a <= 1$, so that derivative is not 0 and can do some learning for that case.

Sparse activation: For example, in a randomly initialized networks, only about 50% of hidden units are activated (having a non-zero output)

Pooling



- Other options: Average pooling, L2-norm pooling, random pooling

- Pooling helps the representation become slightly invariant to small translations of the input.
- If input is translated by small amount: values of most pooled outputs don't change

Batch Gradient Descent

Algorithm 1 Batch Gradient Descent at Iteration k

Require: Learning rate ϵ_k

Require: Initial Parameter θ

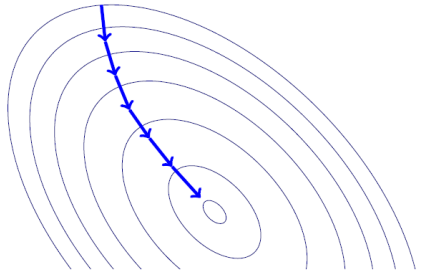
1: **while** stopping criteria not met **do**

2: Compute gradient estimate over N examples:

3: $\hat{\mathbf{g}} \leftarrow +\frac{1}{N}\nabla_\theta \sum_i L(f(\mathbf{x}^{(i)};\theta), \mathbf{y}^{(i)})$

4: Apply Update: $\theta \leftarrow \theta - \epsilon\hat{\mathbf{g}}$

5: **end while**



- Positive: Gradient estimates are stable
- Negative: Need to compute gradients over the entire training for one update

Stochastic Gradient Descent

Algorithm 2 Stochastic Gradient Descent at Iteration k

Require: Learning rate ϵ_k

Require: Initial Parameter θ

1: **while** stopping criteria not met **do**

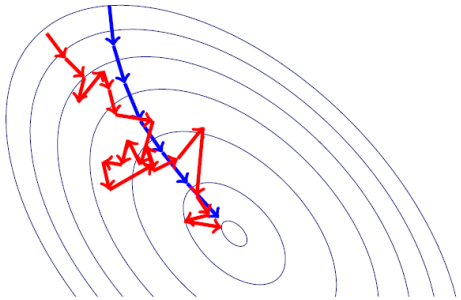
2: Sample example $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ from training set

3: Compute gradient estimate:

4: $\hat{\mathbf{g}} \leftarrow +\nabla_\theta L(f(\mathbf{x}^{(i)}; \theta), \mathbf{y}^{(i)})$

5: Apply Update: $\theta \leftarrow \theta - \epsilon\hat{\mathbf{g}}$

6: **end while**



- ϵ_k is learning rate at step k
- Sufficient condition to guarantee convergence:

$$\sum_{k=1}^{\infty} \epsilon_k = \infty \text{ and } \sum_{k=1}^{\infty} \epsilon_k^2 < \infty$$

Learning Rate Schedule

- In practice the learning rate is decayed linearly till iteration τ

$$\epsilon_k = (1 - \alpha)\epsilon_0 + \alpha\epsilon_\tau \text{ with } \alpha = \frac{k}{\tau}$$

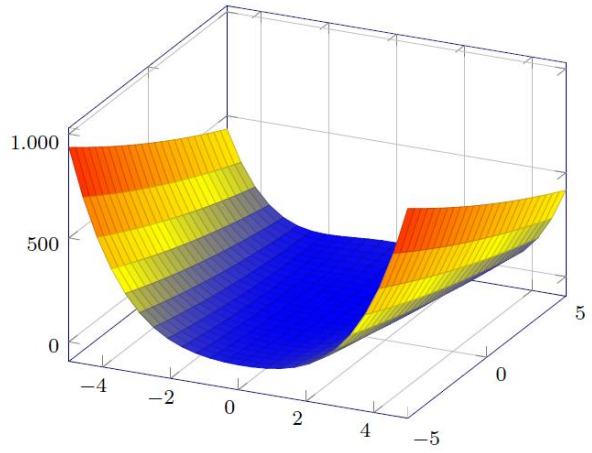
- τ is usually set to the number of iterations needed for a large number of passes through the data
- ϵ_τ should roughly be set to 1% of ϵ_0
- How to set ϵ_0?

Minibatching

- Potential Problem: Gradient estimates can be very noisy
- Obvious Solution: Use larger mini-batches
- Advantage: Computation time per update does not depend on number of training examples N
- This allows convergence on extremely large datasets
- See: Large Scale Learning with Stochastic Gradient Descent by Leon Bottou

Momentum

- The Momentum method is a method to accelerate learning using SGD
- In particular SGD suffers in the following scenarios:
  - Error surface has high curvature
  - We get small but consistent gradients
  - The gradients are very noisy



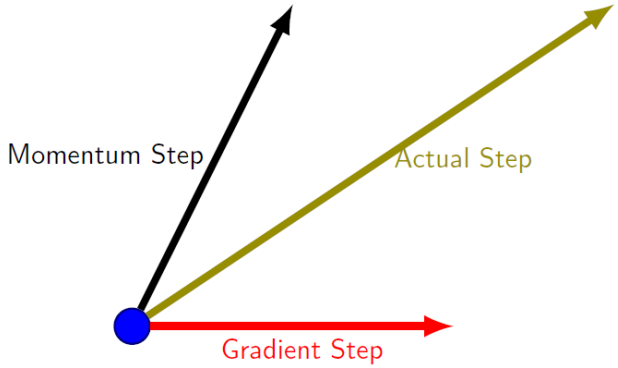
- Gradient Descent would move quickly down the walls, but very slowly through the valley floor

Momentum

- How do we try and solve this problem?
- Introduce a new variable $\mathbf{v}$, the velocity
- We think of $\mathbf{v}$ as the direction and speed by which the parameters move as the learning dynamics progresses
- The velocity is an exponentially decaying moving average of the negative gradients

$$\mathbf{v} \leftarrow \alpha \mathbf{v} - \epsilon \nabla_\theta \left(L(f(\mathbf{x}^{(i)}; \theta), \mathbf{y}^{(i)}) \right)$$

- $\alpha \in [0, 1)$Update rule: $\theta \leftarrow \theta + \mathbf{v}$

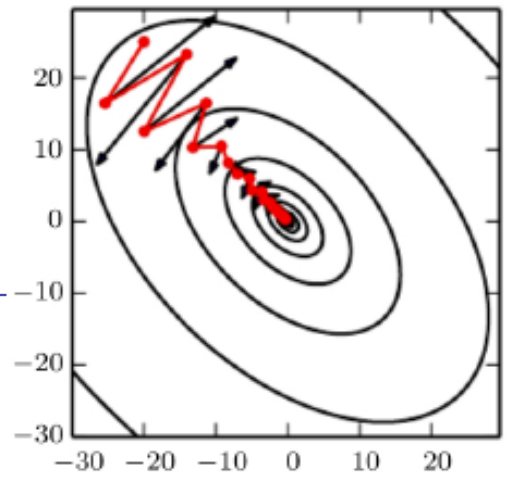


SGD with Momentum

Algorithm 2 Stochastic Gradient Descent with Momentum

Require: Learning rate ϵ_k
Require: Momentum Parameter α
Require: Initial Parameter θ
Require: Initial Velocity $\mathbf{v}$

while stopping criteria not met **do**
  Sample example $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ from training set
  Compute gradient estimate:
  $\hat{\mathbf{g}} \leftarrow +\nabla_\theta L(f(\mathbf{x}^{(i)}; \theta), \mathbf{y}^{(i)})$
  Compute the velocity update:
  $\mathbf{v} \leftarrow \alpha\mathbf{v} - \epsilon\hat{\mathbf{g}}$
  Apply Update: $\theta \leftarrow \theta + \mathbf{v}$
end while



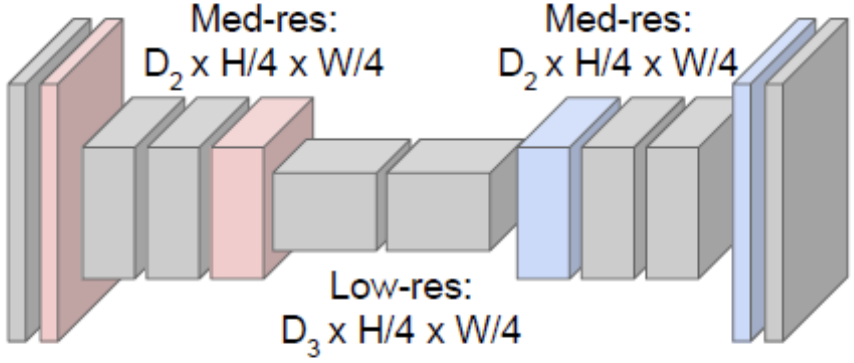
Semantic Segmentation: FCN

Downsampling:
Pooling, strided convolution

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

Upsampling:
Unpooling or strided transpose convolution

Input:
3 x H x W



High-res:
D1 x H/2 x W/2

High-res:
D1 x H/2 x W/2

Predictions:
H x W

Long, Shelhamer, and Darrell, "Fully Convolutional Networks for Semantic Segmentation", CVPR 2015
Noh et al, "Learning Deconvolution Network for Semantic Segmentation", ICCV 2015

Slide credit: Stanford University Course CS231n

Successful AI Models in Various Fields

ResNet

Faster R-CNN

Yolo object detector

MobileNet

EfficientNet

Bert

Vision Transformer

GTP-X

CLIP

…..

Generative AI

Generative AI

Generative AI (GenAI) refers to a category of artificial intelligence (AI) algorithms that generate new outputs based on the data they have been trained on. Unlike traditional AI systems that are designed to recognize patterns and make predictions, generative AI creates new content in the form of images, text, audio, and more.

Generative Adversarial Networks (GANs) are a type of GenAI that can create new content. A GAN consists of two neural networks: a generator that creates new data and a discriminator that evaluates the data. The generator and discriminator work together, with the generator improving its outputs based on the feedback it receives from the discriminator until it generates content that is indistinguishable from real data.

Applications of Generative AI

- **Images:** Generative AI can create new images based on existing ones, such as creating a new portrait based on a person's face or a new landscape based on existing scenery
- **Text:** Generative AI can be used to write news articles, poetry, and even scripts. It can also be used to translate text from one language to another
- **Audio:** Generative AI can generate new music tracks, sound effects, and even voice acting

Generative AI is a type of artificial intelligence (AI) that uses machine learning algorithms to create new and original content like images, videos, text, and audio.

1 Forming a Database

A neural network, consisting of various information or media files like images, text, data, sounds, etc., forms the basis of artificial intelligence.

2 Inputting a Prompt

The user provides the AI with a description or sample of the desired content...

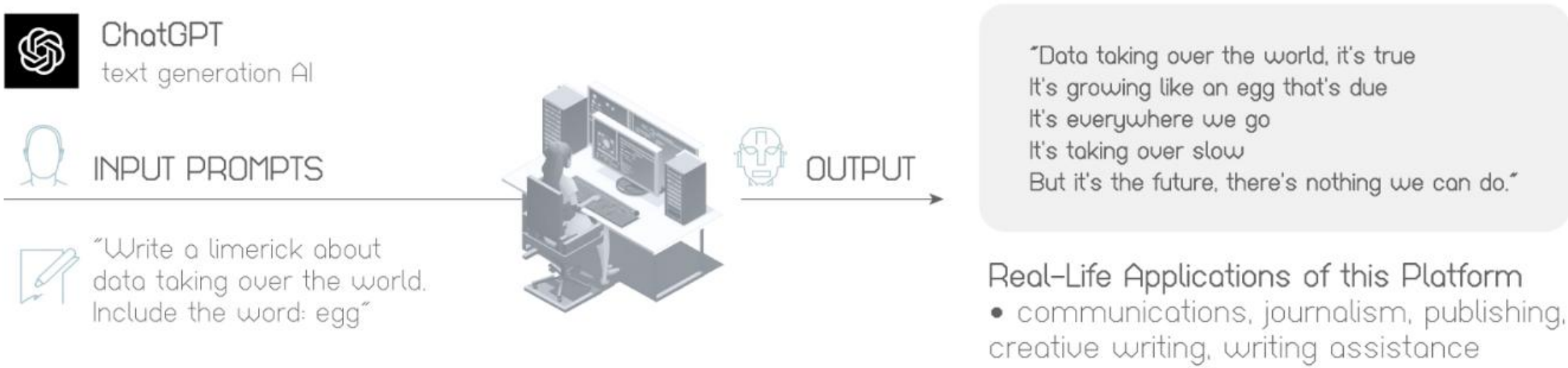
Prompts can be any user-submitted material, like words, numbers, or photos.

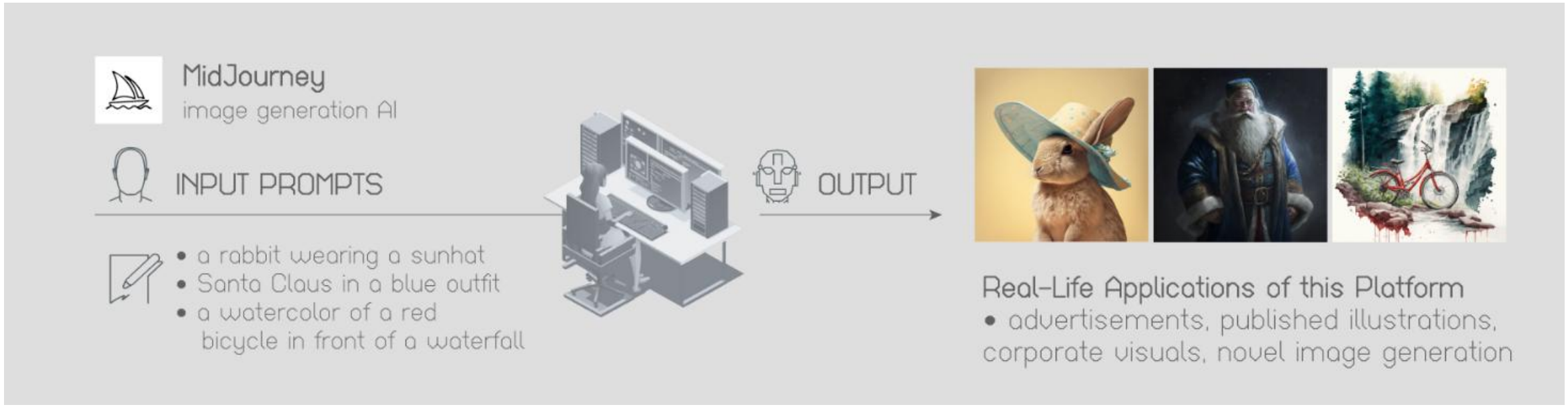
3 Generating Content

...and the AI uses its neural network to generate new examples that are similar to the ones it has trained from.

This image was created on MidJourney using the following text prompt:
a technical illustration of a woman sitting behind a desktop computer on a long table, isometric view, 3D rendering, realistic, 4k

Examples of Generative AI Applications




MidJourney
image generation AI
INPUT PROMPTS
• a rabbit wearing a sunhat
• Santa Claus in a blue outfit
• a watercolor of a red bicycle in front of a waterfall
OUTPUT
Real-Life Applications of this Platform
• advertisements, published illustrations, corporate visuals, novel image generation

Codex
computer coding generation AI

INPUT PROMPTS

"Write script to change a website's background color in Python"

OUTPUT

Here's an example script that uses the 'requests' and 'BeautifulSoup' libraries in Python to change the background color of a website:

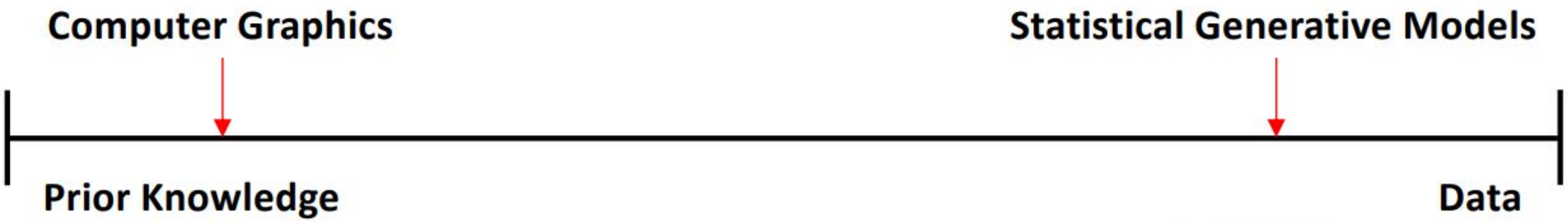
```
import requests
from bs4 import BeautifulSoup

# send a GET request to the website
response = requests.get("https://yourwebsite.com")

# parse the HTML content
soup = BeautifulSoup(response.content, 'html.parser')

# locate the element you want to change the background color of
element = soup.find('body')

# add a new style attribute to the element
element['style'] = 'background-color: pink;'

# print the modified HTML
print(soup.prettify())
```

Real-Life Applications of this Platform

- web design, software development, coding/scripting, technology

Deep Generative Models

Data-driven



Material, Physical Modeling, Lighting ….

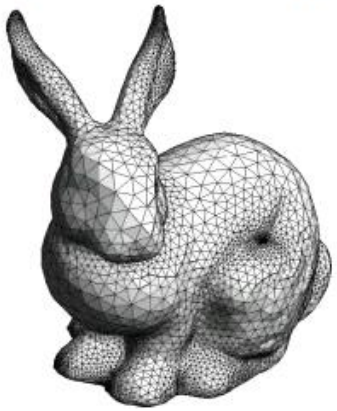
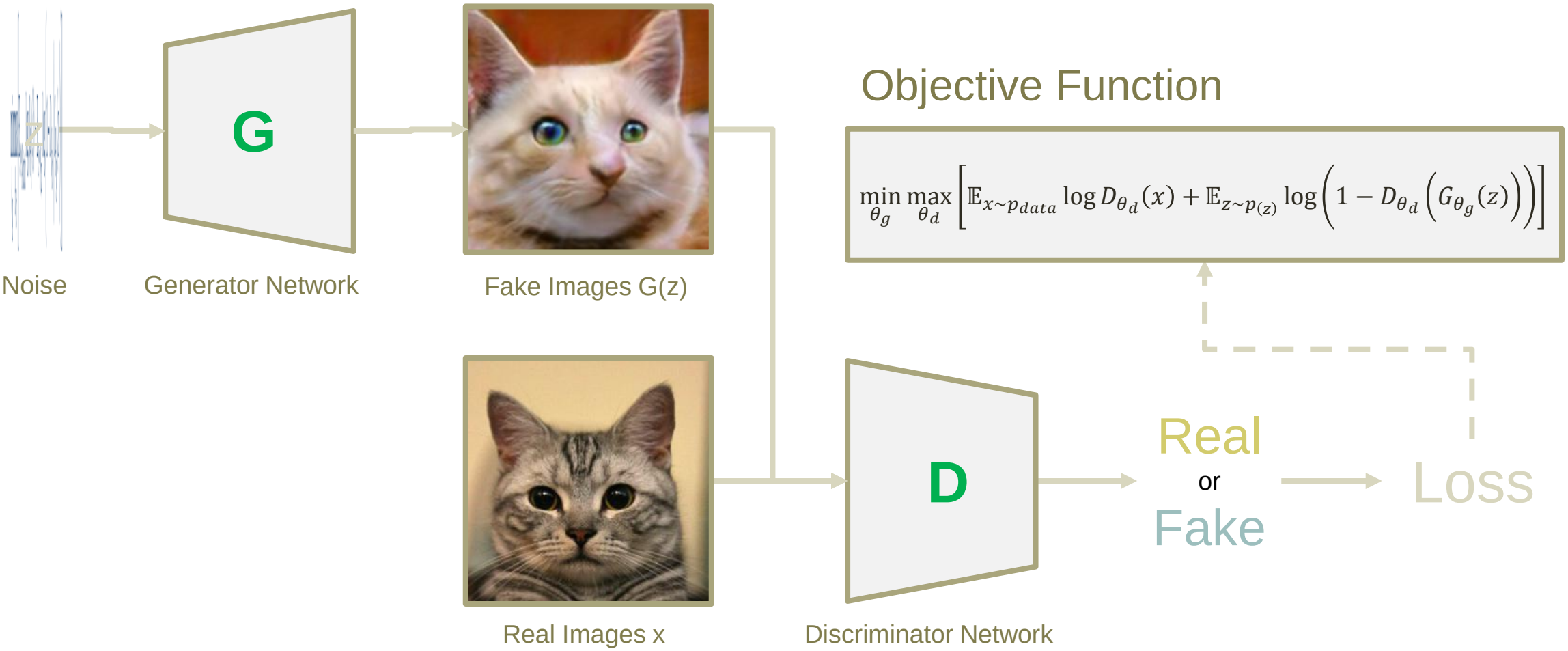


Image Generation

Cat

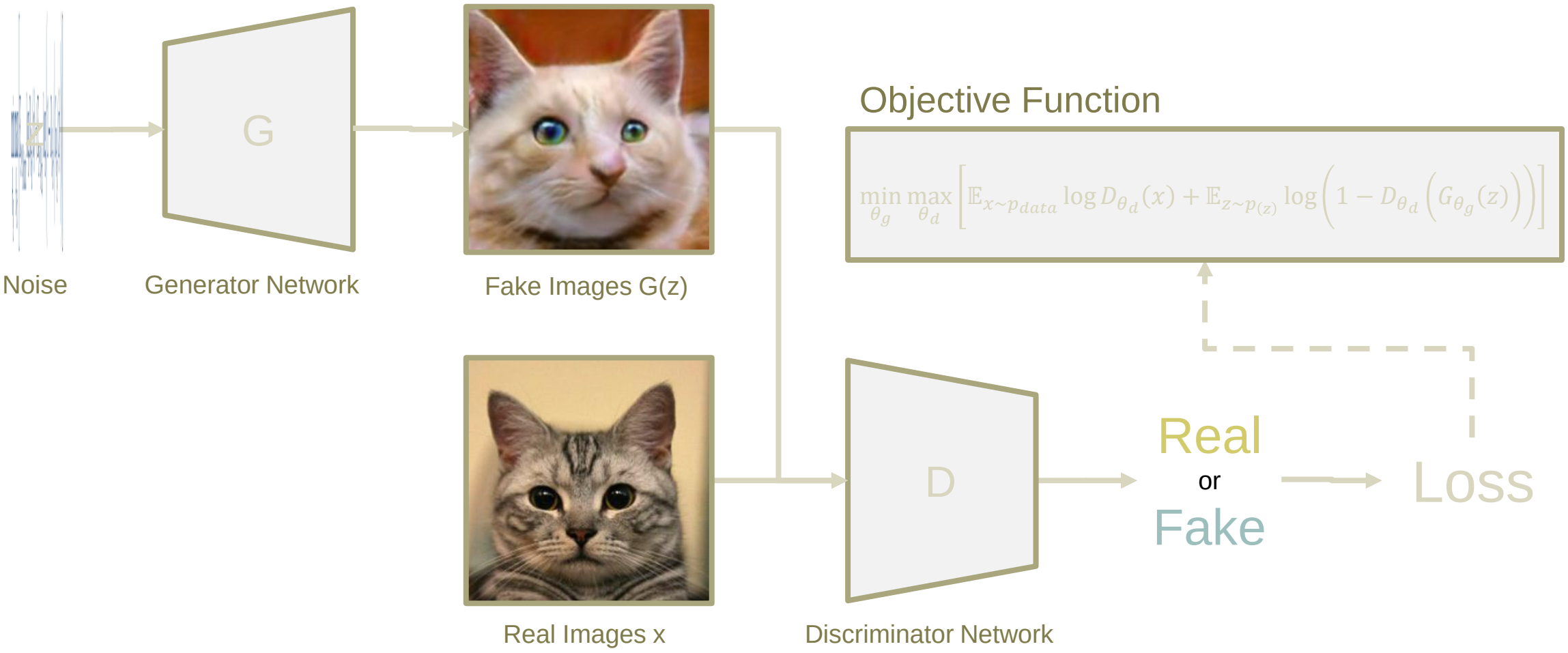
?

GAN Architecture



Generative adversarial nets, Goodfellow, J Pouget-Abadie, M Mirza, B Xu, D Warde-Farley, S Ozair, NIPS 2014

GAN Architecture



Generative adversarial nets, Goodfellow, J Pouget-Abadie, M Mirza, B Xu, D Warde-Farley, S Ozair, NIPS 2014

Conditional GAN

Conditional GANs learn a mapping from observed image x and random noise vector z, to y, G : {x, z} → y.

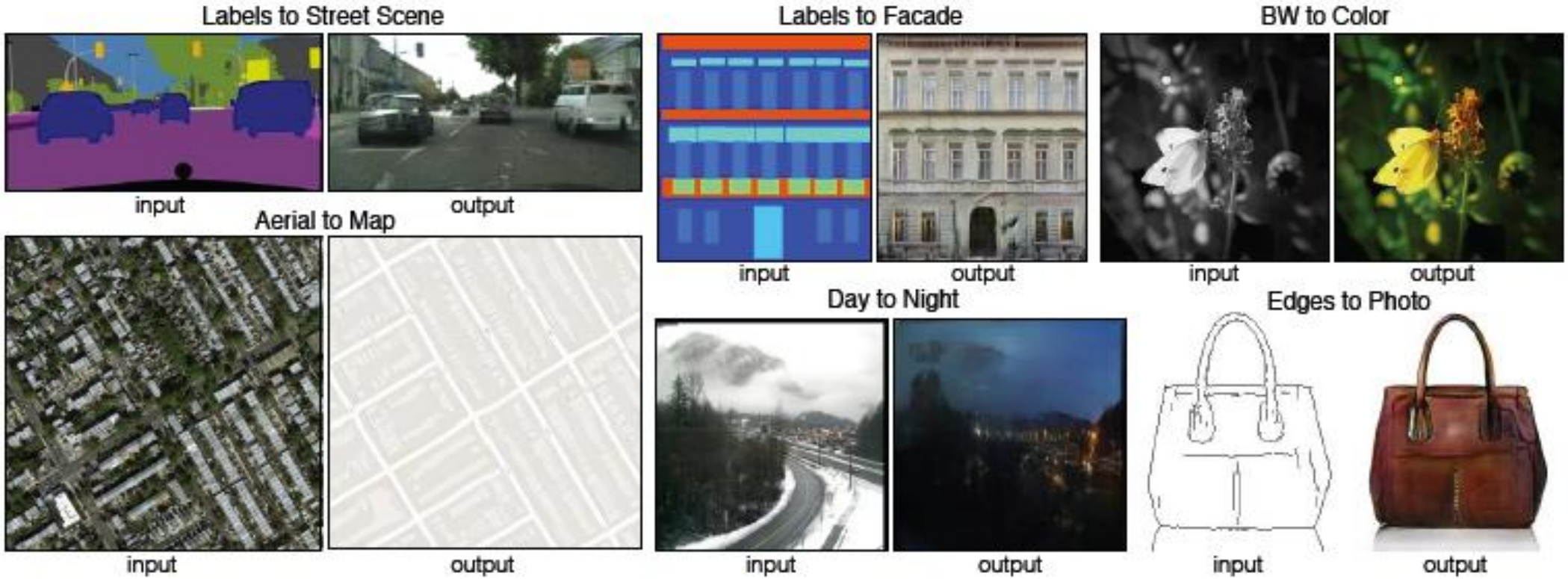
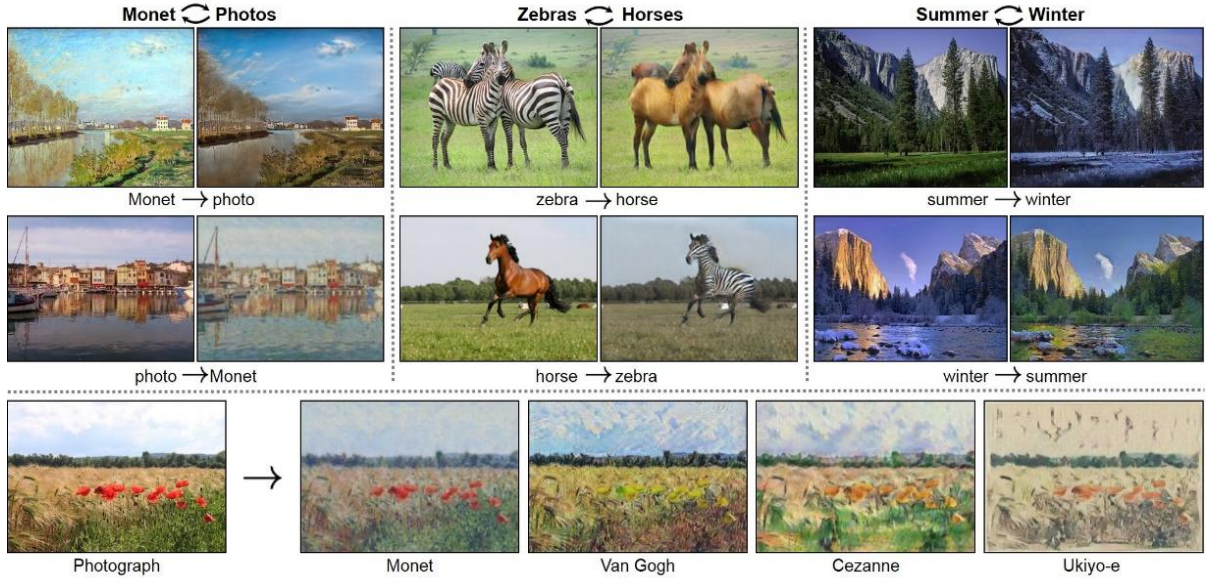


Image-to-Image Translation with Conditional Adversarial Networks, P. Isola et al., CVPR 2017

Unsupervised Image-to-Image Translation

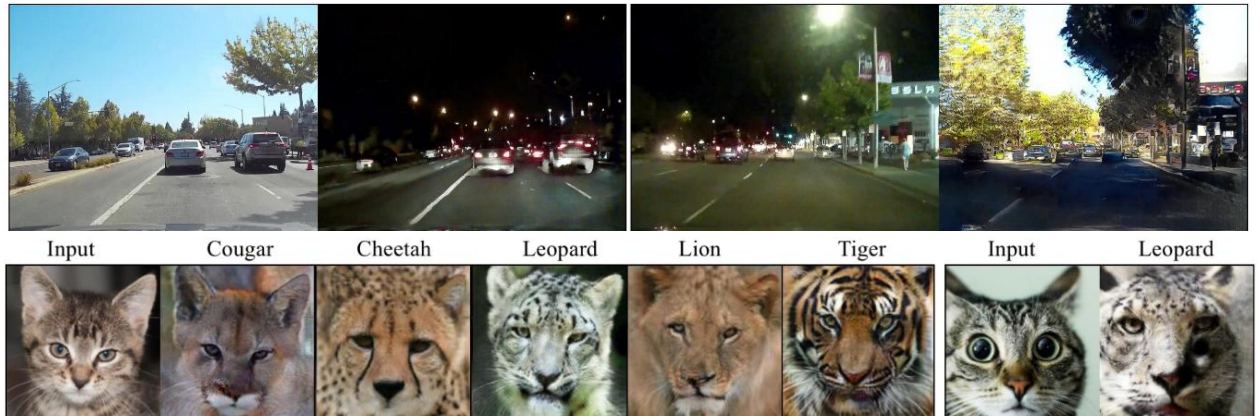
(Unpaired Data Training)

CycleGAN



[J. Y. Zhu et al., ICCV 2017]

UNIT [M. Y. Liu et al., NIPS 2017]



MUNIT

[X. Huang et al., ECCV 2018]

Discriminative vs. Generative Models

Discriminative models: classify data

finding **conditional distribution** $P(Y|X)$

$P(Y = Cat|X =$  $) = 0.99$








Decision boundary

Generative models: generate data

finding **joint distribution** $P(Y, X)$

$Y = Cat, X =$ 

$Y = Dog, X =$ 

The data distribution can be high-dimensional, like images

Example Applications of Generative Models

"Class" conditional generative models

$$P(X = \text{  } | Y = Cat)$$

"Text" conditional generative models

$$P(X = \text{  } | Y = \text{“a flower with white petals and yellow stamen”})$$

"Text-image" conditional generative models

Joint distribution

$$P(X = \text{  } | Y_1 = \text{  }, Y_2 = \text{“a yellow bird with grey wings”})$$

Generative Adversarial Text to Image Synthesis. S. Reed, Z. Akata et al. ICML. 2016.
Semantic Image Synthesis via Adversarial Learning. H. Dong, S. Yu et al. ICCV 2017

Image Super Resolution

$P(High\ resolution\ image \mid Low\ resolution\ image)$

$P(High\ quality\ image \mid Low\ quality\ image)$

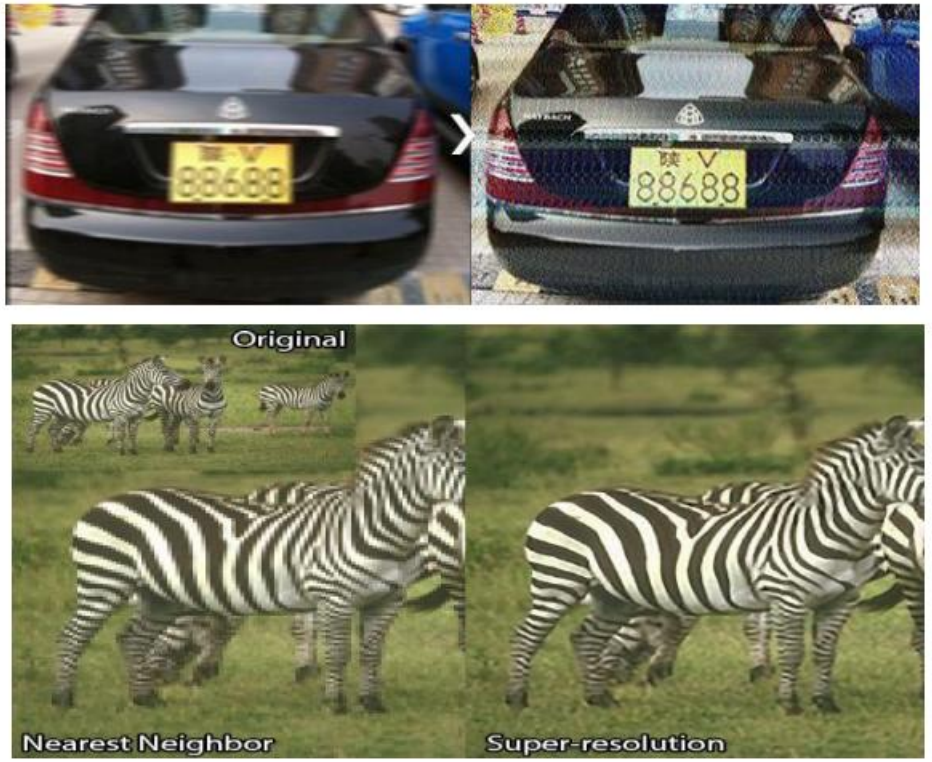
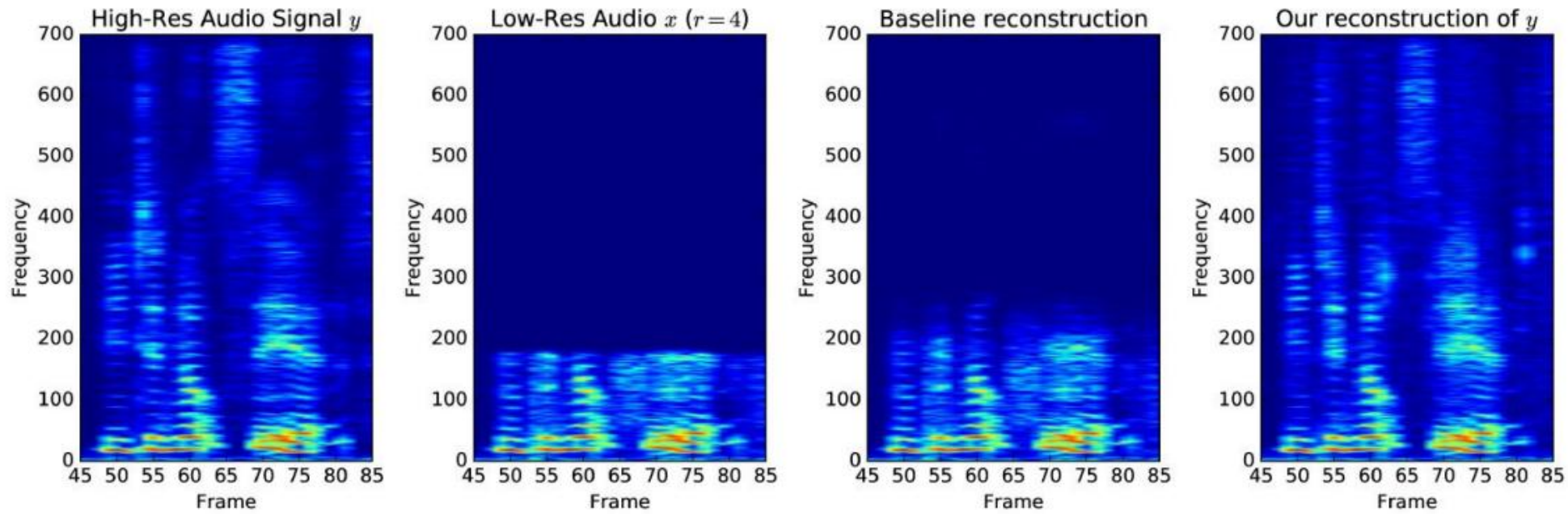


Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. C. Ledig, L. Theis et al. CVPR 2017

Audio Super Resolution

$$P(High\ resolution\ signal \mid Low\ resolution\ signal)$$



Content Generation with StyleGAN3

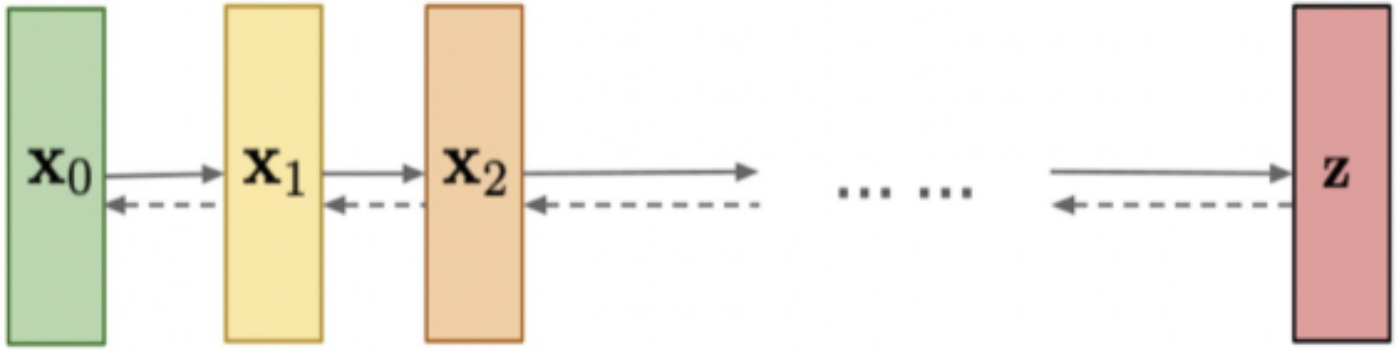


Alias-Free Generative Adversarial Networks, Tero Karras et al., arXiv:2106.12423v4

Diffusion Models

- Idea: Estimating and analyzing small step sizes is more tractable/easier than a single step from random noise to the learned distribution
- Convert a well-known and simple *base distribution* (like a Gaussian) to the *target (data) distribution* iteratively, with small step sizes, via a Markov chain:

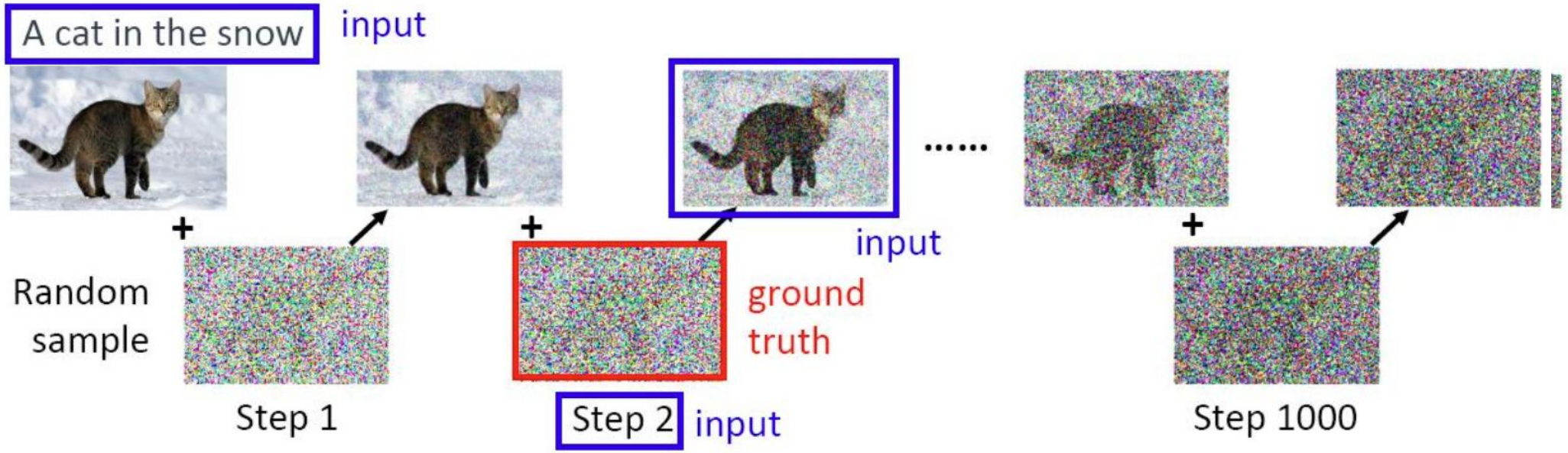
Diffusion models:
Gradually add Gaussian noise and then reverse



- Markov chain: outlines the probability associated with a sequence of events occurring based on the state in the previous event.

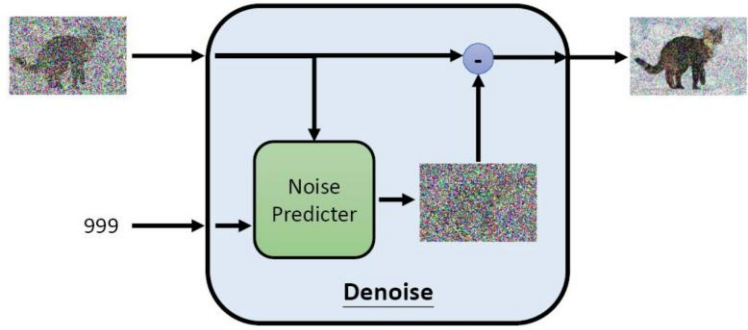
Diffusion Model – Forward Process

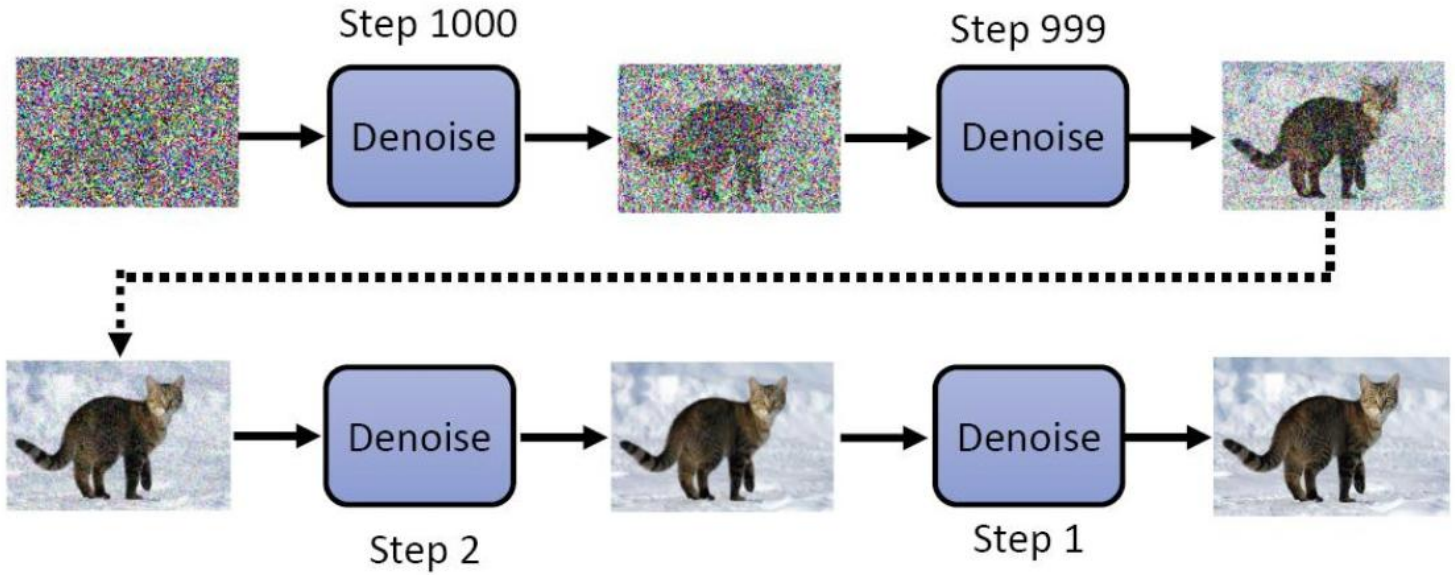
- The creation of training paired images are controllable.
  - The amount of noises in each step is performed by linear scheduling.
  - Then, the training pairs are used to train the model for de-noising.
  - Overcome the challenge of GAN to output generated image in ONE step.
    - To remove an amount of noises of a cat image vs. to generate the cat directly



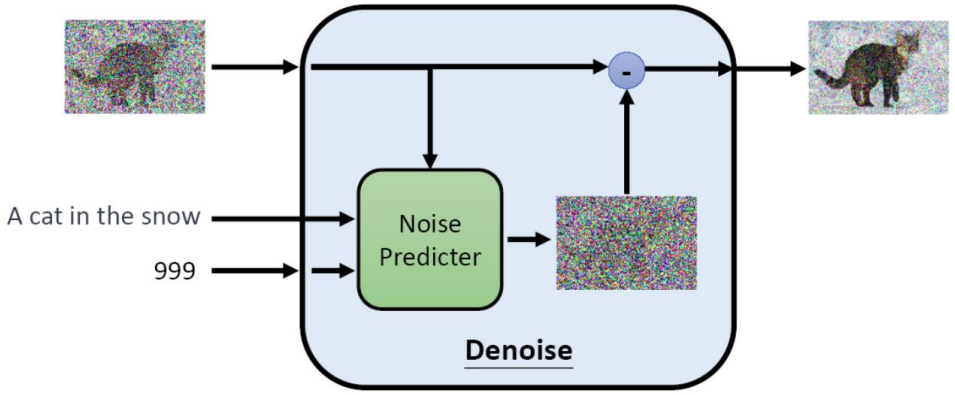
Diffusion Model – Reverse Process

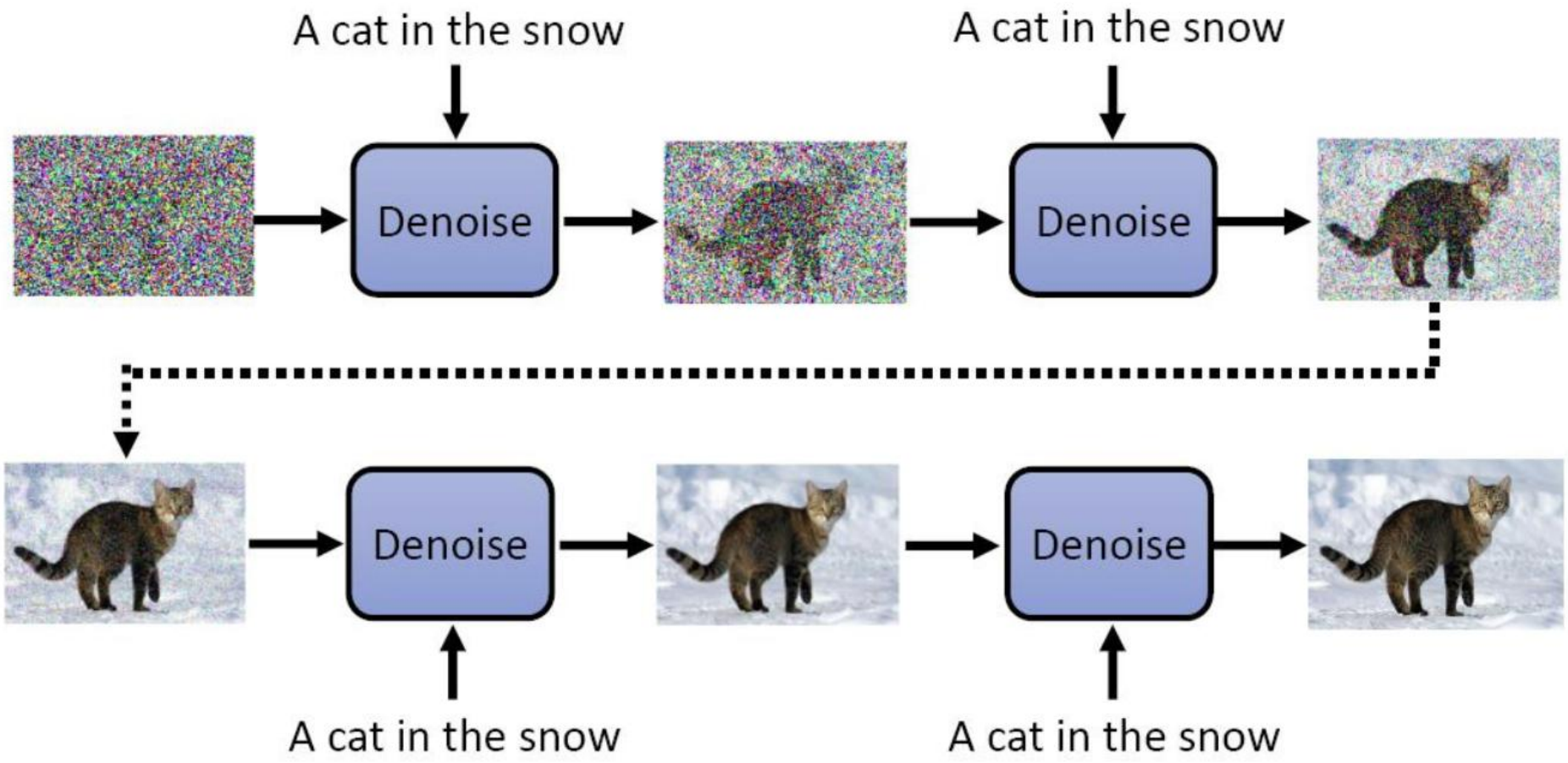
- Step number as conditions to the model
  - The SAME model is used in all steps.





Text-to-Image Generation





DALL-E 3 (OpenAI)

Midjourney

Emu (Meta)

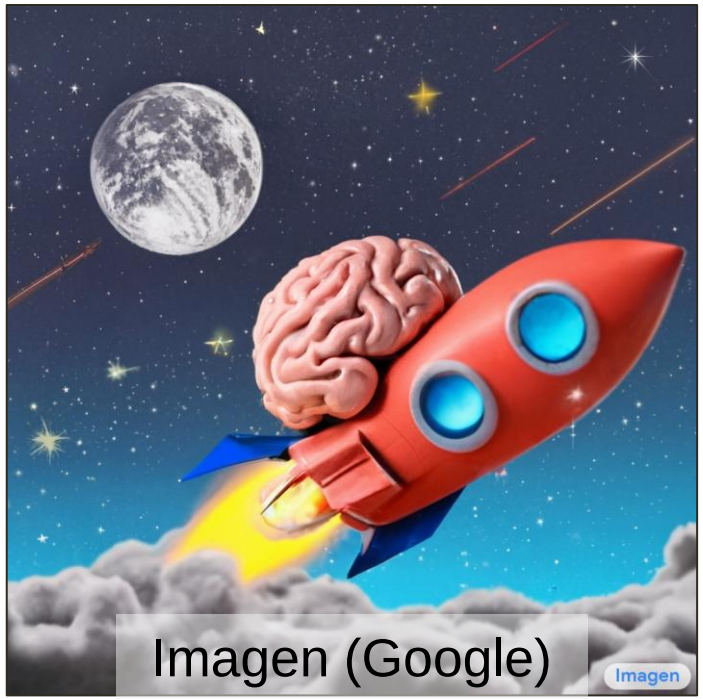
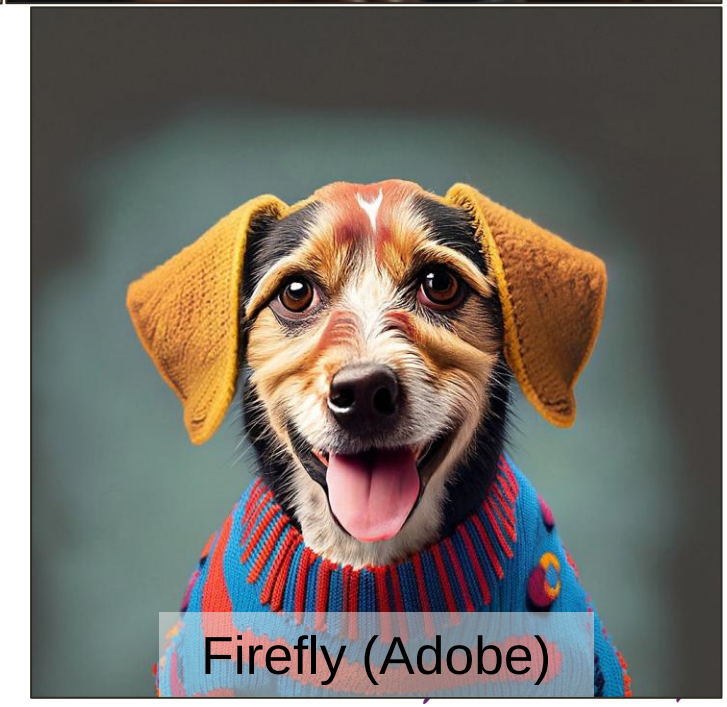


Imagen (Google)

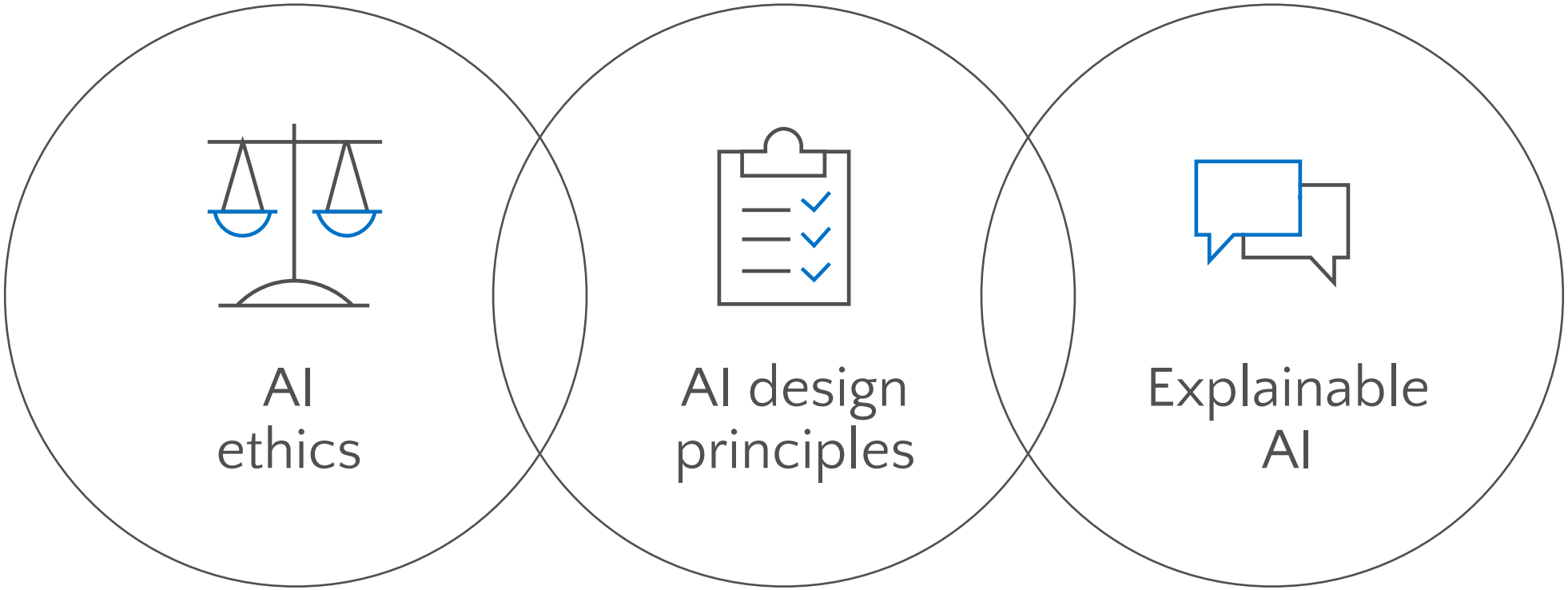
SDXL (Stability.ai)



Firefly (Adobe)


THIS IS
DEEP FAKE

Our shared responsibility



Conclusion

- Applications of Generative AI technologies to multimedia content generation are rapidly increasing.
- Mis-information and deepfake detection is a major challenge today
- How to ensure proper use of generative AI is an urgent topic in our society.