Machine Learning to Language Model

Topic 01 - Introduction to Machine Learning

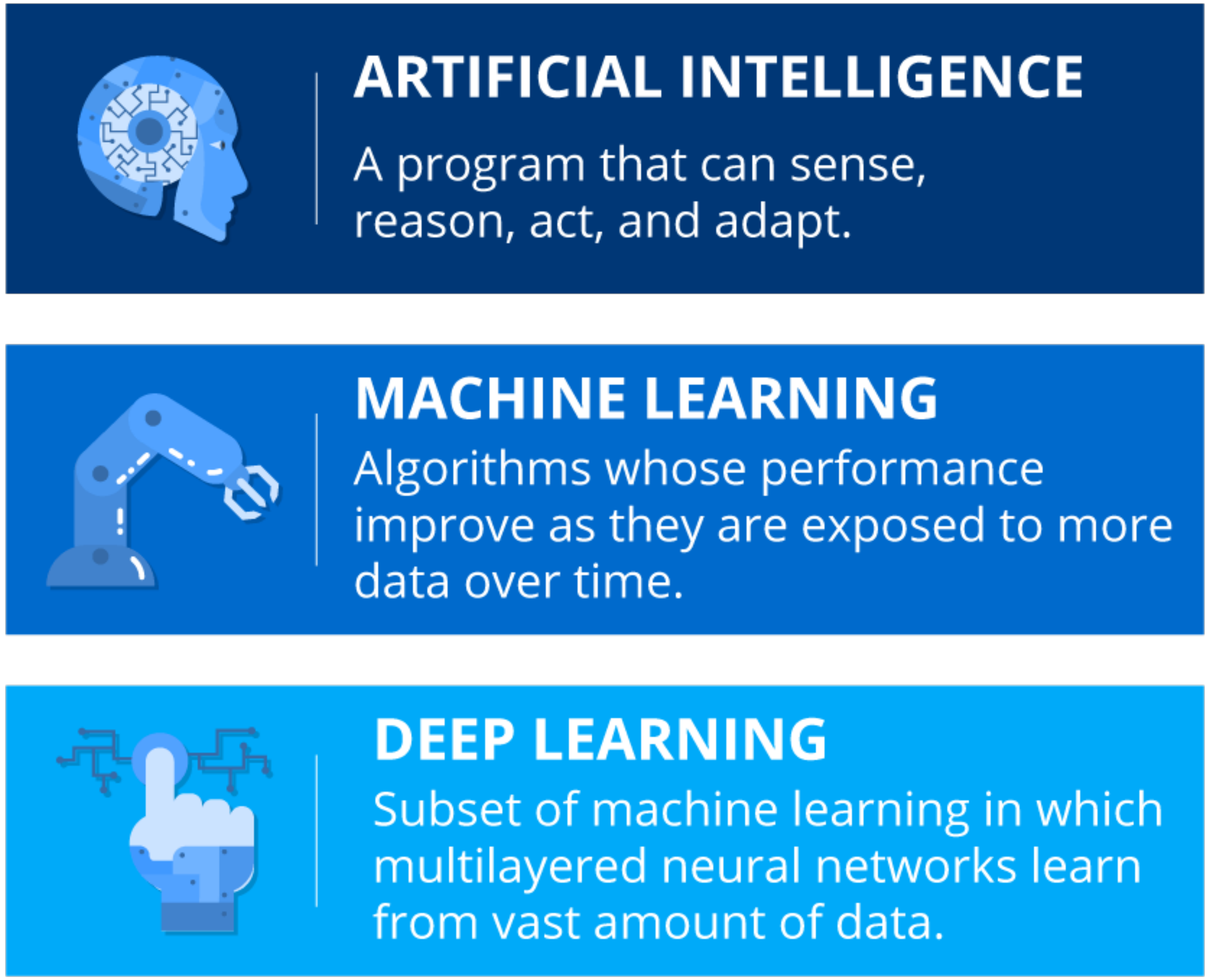
Jaihua Yen
https://jaihuayen.github.io/homeweb/

Contents

- AI Models
- Gradient Descent
- Demo
- Wrap Up

AI Models

AI Models



https://www.globaltechcouncil.org/artificial-intelligence/clearing-the-confusion-ai-vs-machine-learning-vs-deep-learning/

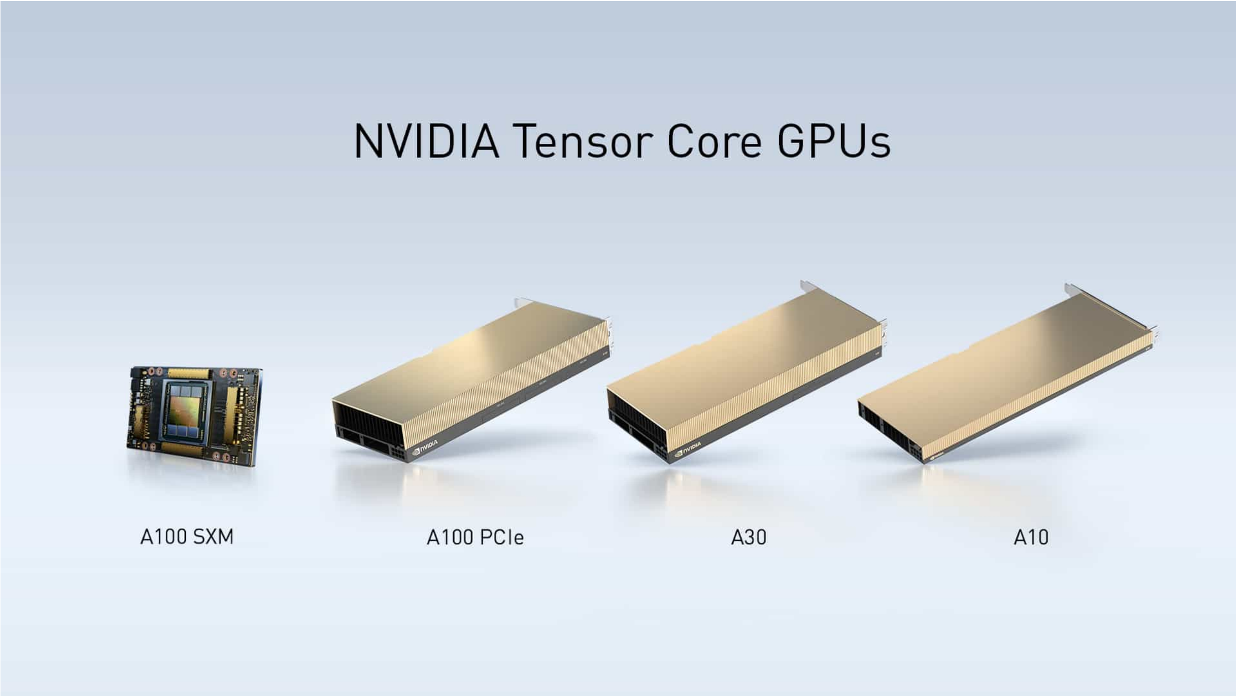
The Era of AI

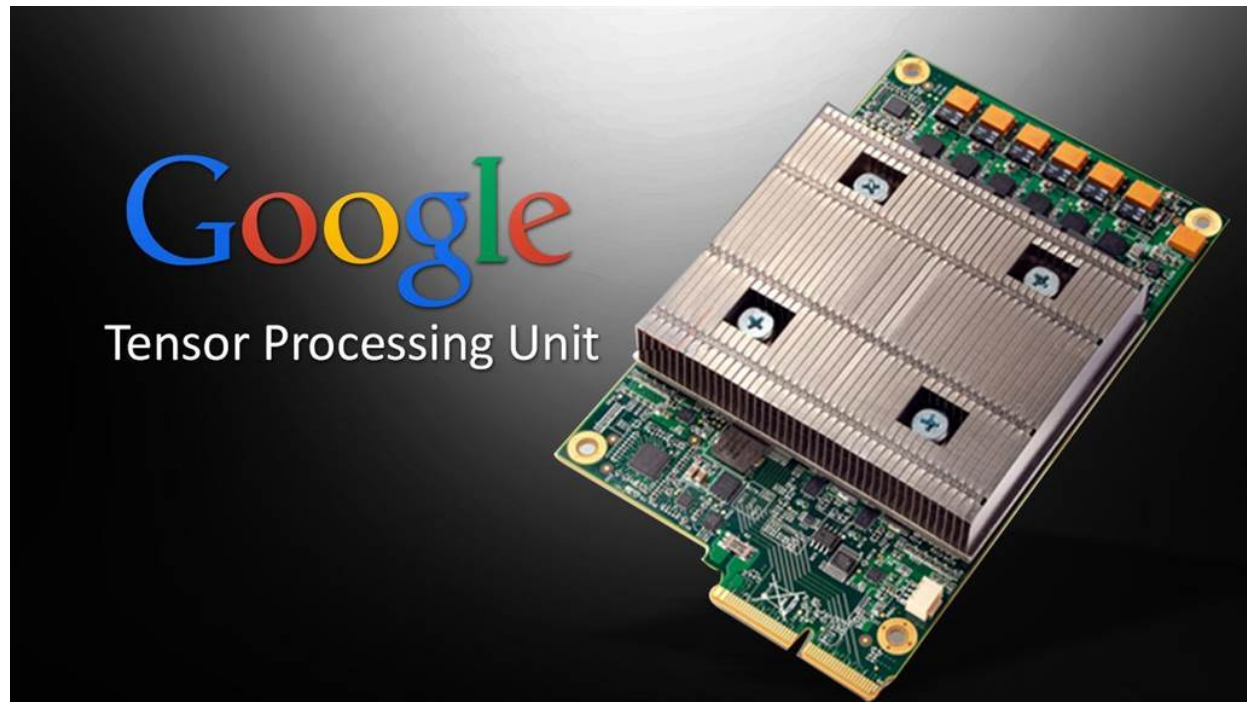
AI could finally be introduced into practice in general tasks!

- Tremendous improvement in computational resources.
- Enhancement of model architecture for model efficiency.
- Release open-source large pre-trained general models.
- Development of novel model training algorithms.

Hugging Face

PyTorch 2.0





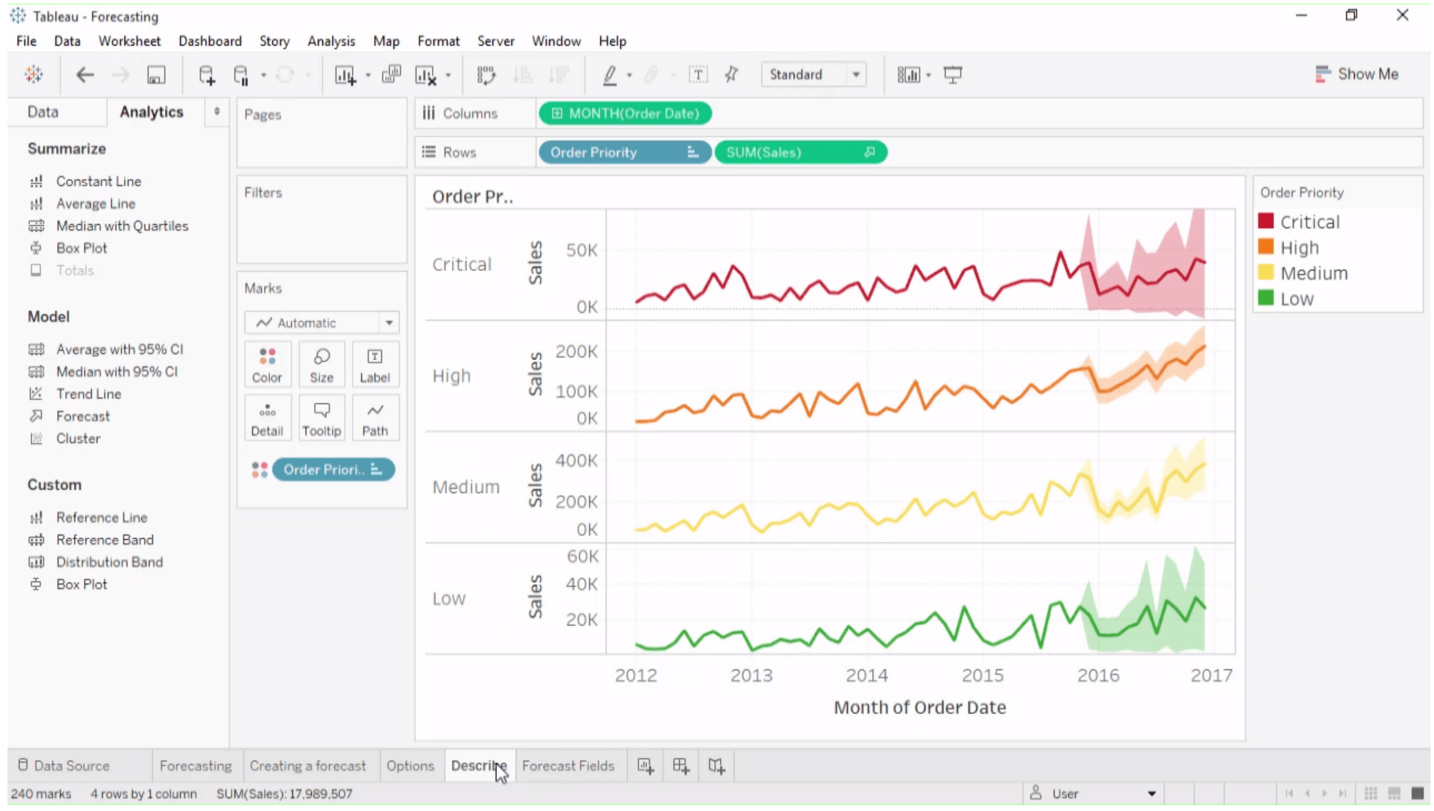
	ResNet Cloud AI	MobileNet Mobile AI	MCUNet Tiny AI
Memory (Activation)	16GB	4GB ← 13,000x smaller	320kB
Storage (Weights)	~TB/PB	256GB	1MB

[MCUNet: Tiny Deep Learning on IoT Devices, Han et al. 2020]

AI Models

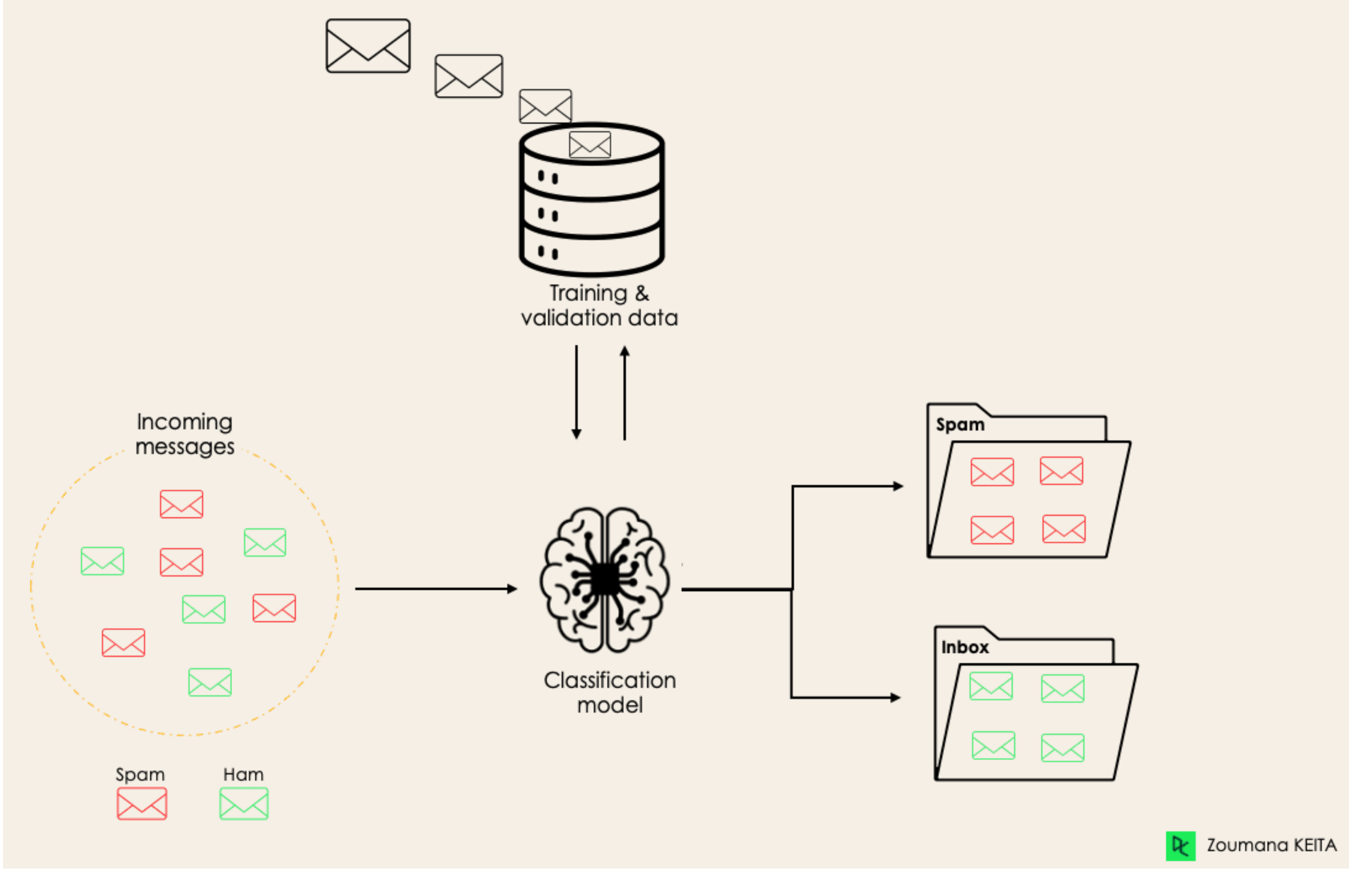
AI models could be used in these tasks:

Forecasting



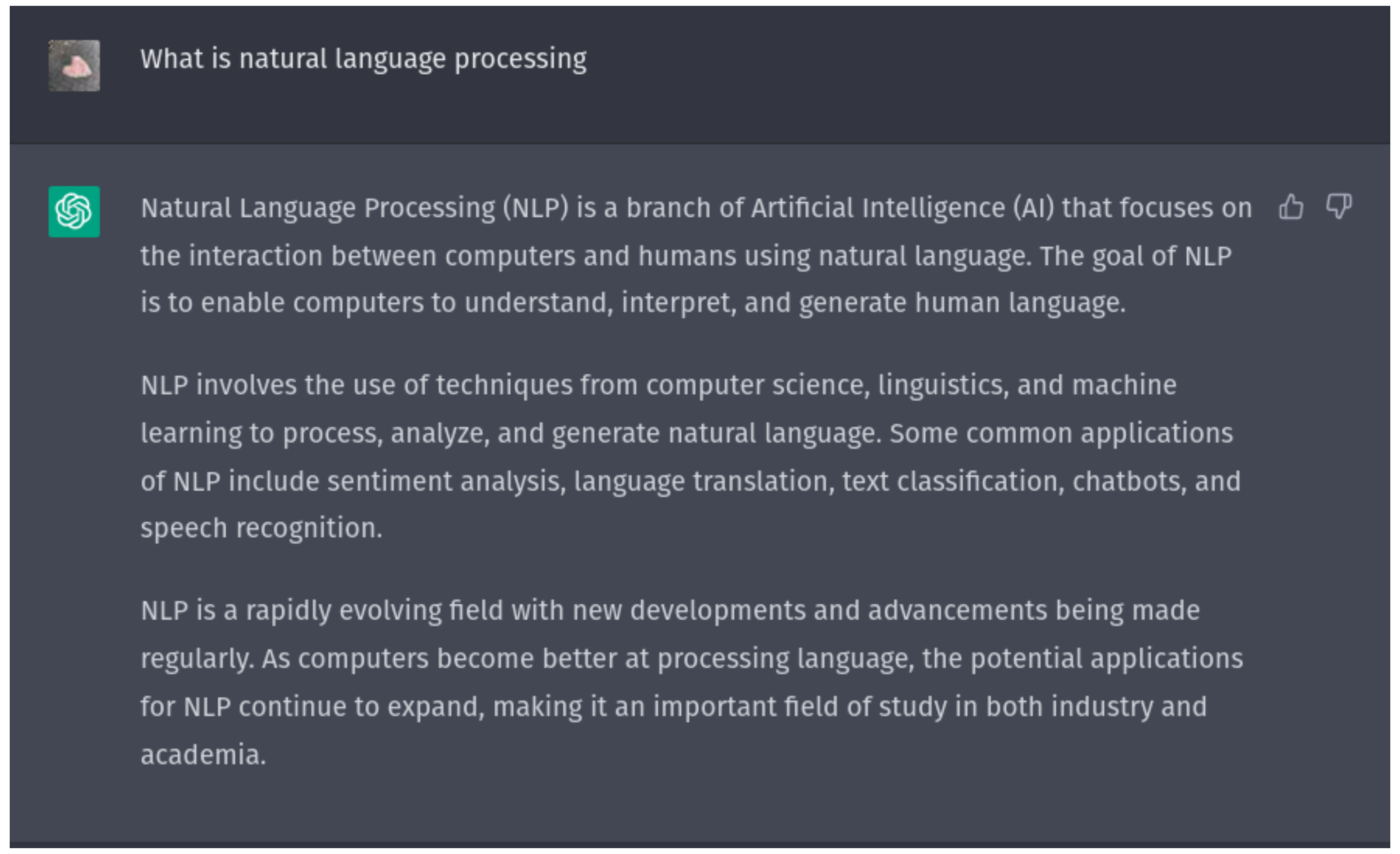
https://www.tableau.com/learn/articles/time-series-forecasting

Classification



https://www.datacamp.com/blog/classification-machine-learning

Generating



https://openai.com/blog/chatgpt

AI Models

AI models are functions!

AI

The image of dog.

Translate “你好” in English.

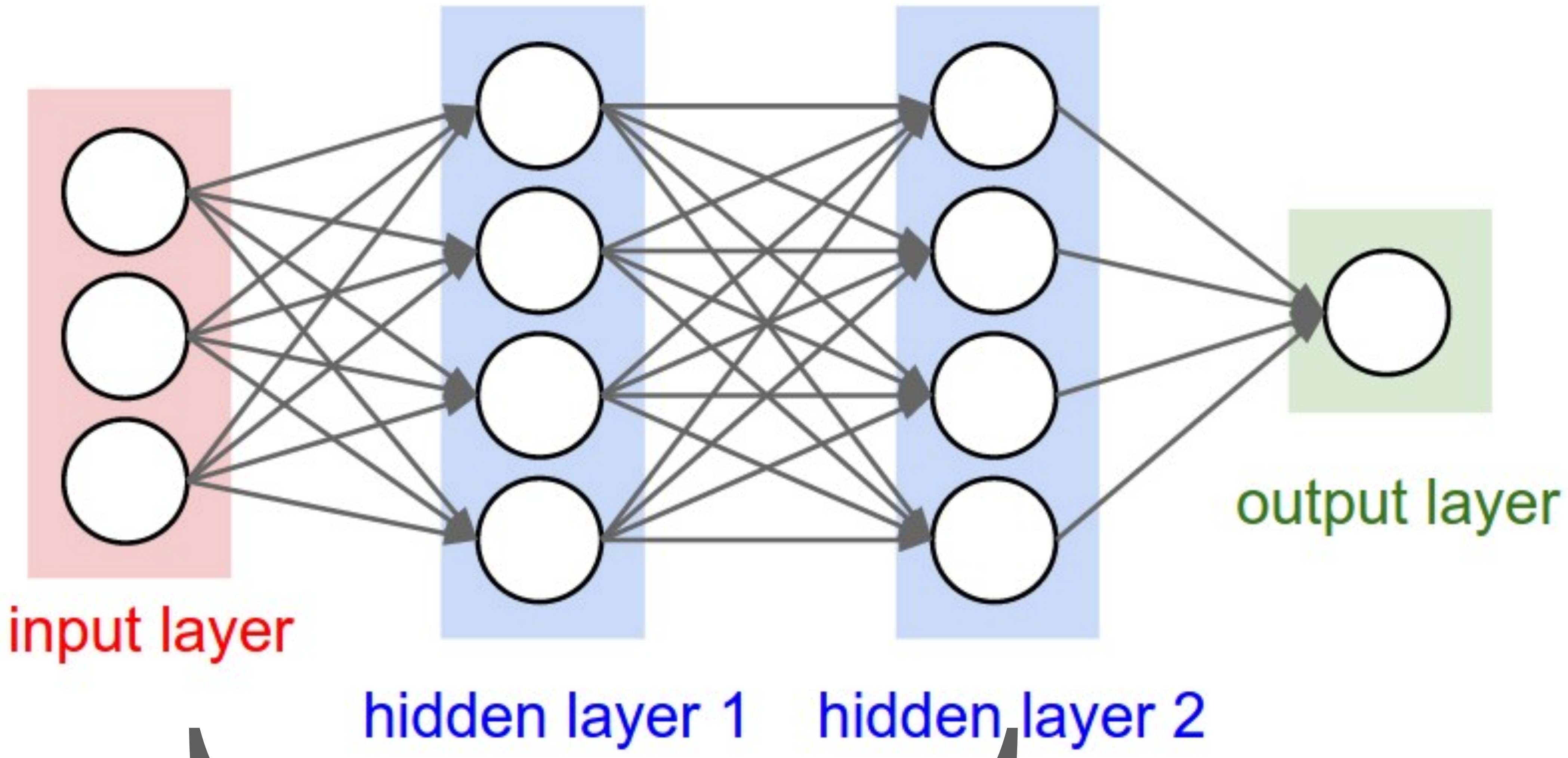
$f(x)$

Hello

AI Models

AI models are functions built by neural networks

Neurons Activations



$\boxed{g(x)}$

https://cs231n.github.io/convolutional-networks/

The function we want to learn from data by training

Universal Approximation Theorem

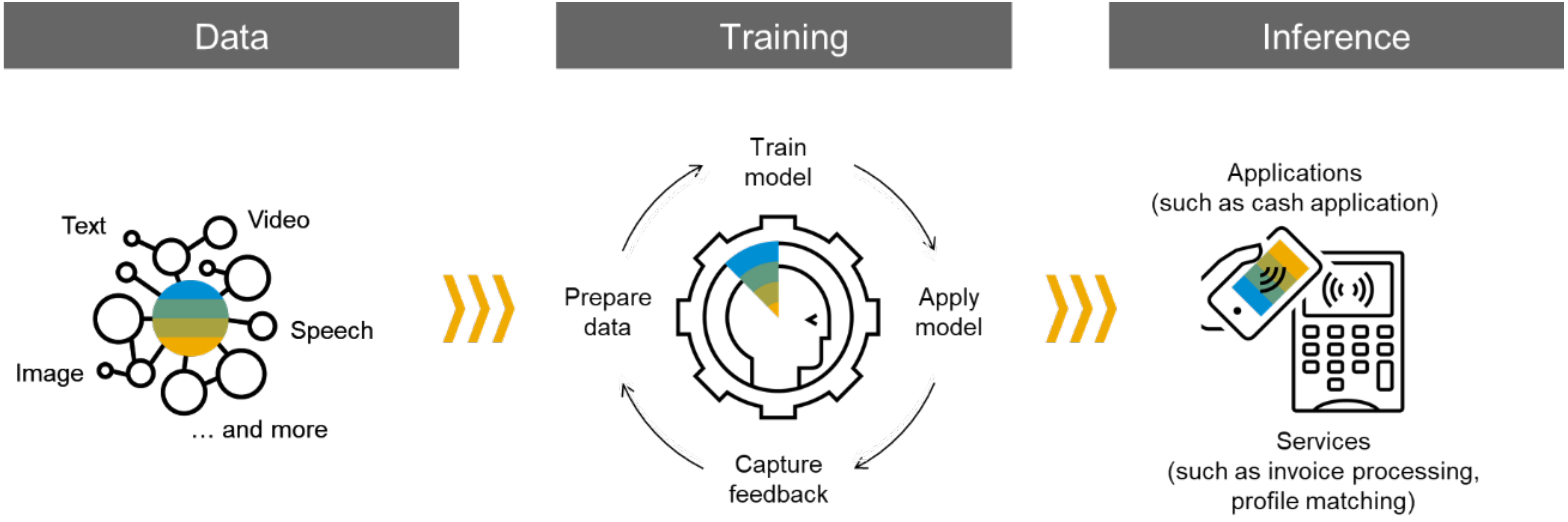
$$g(x) \approx f(x)$$

Any function can be approximated
by neural network!

How to Train AI Models?

Machine Learning

Mathematical Fundations of Model Training

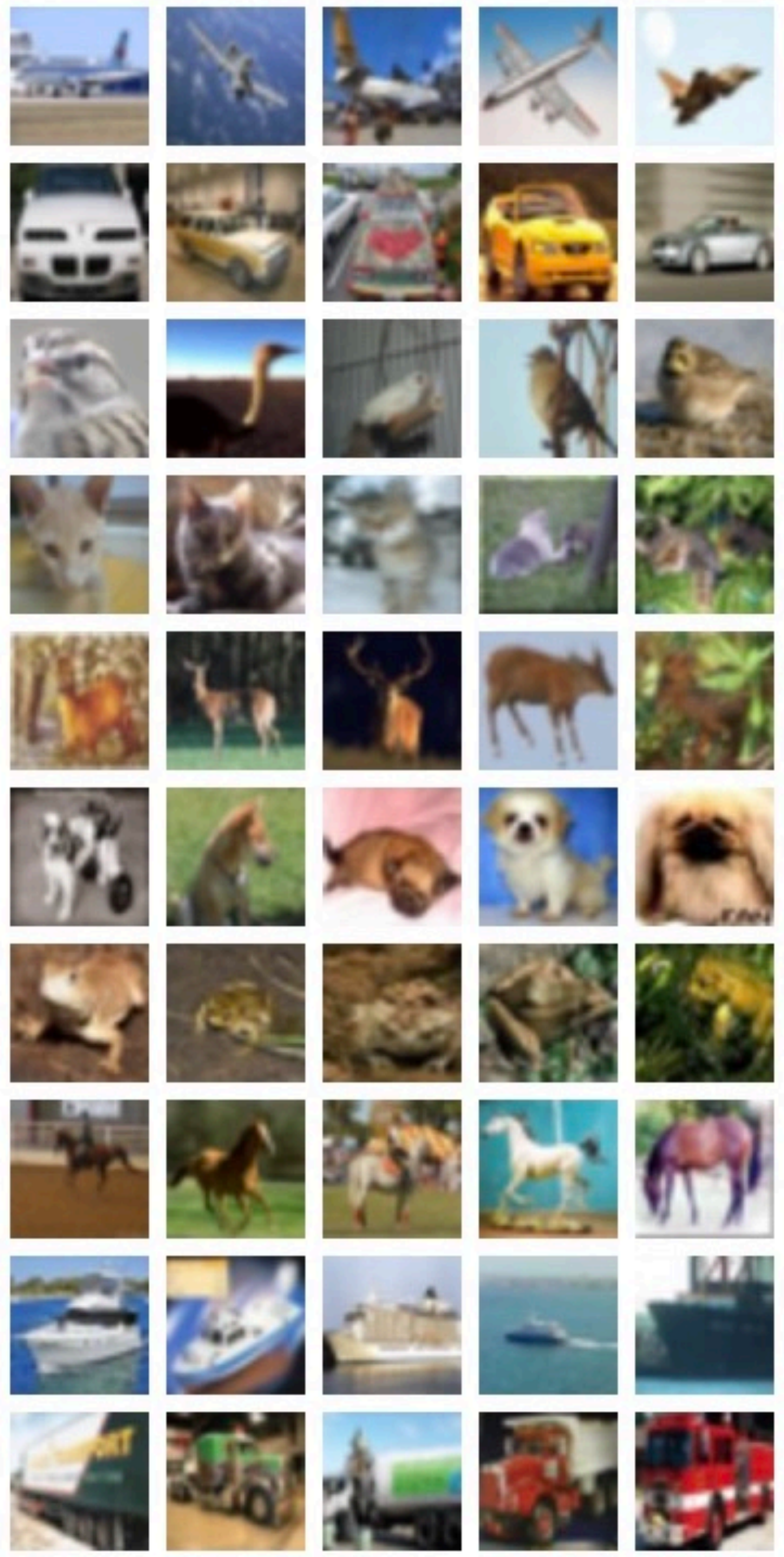


https://blogs.sap.com/2019/04/05/machine-learning-in-sap-strategy/

Training Dataset

Given datasets and tags to train AI model.

x



$$y = g(x)$$

y

airplane

automobile

bird

cat

deer

dog

frog

horse

ship

truck

https://www.cs.toronto.edu/~kriz/cifar.html

Model Training

Cannot perfectly predict the true label in the first time

The difference between the true label and predicted label

x $\quad g(x)$ $\quad \hat{y}$ $\leftrightarrow$ y

automobile ✗ **ship** ✓

We want to improve the model output in order to have better performance!

Loss Function

Measurement of model performance

The difference between the true label and predicted label

$\hat{y}$ $\quad\leftrightarrow\quad$ y

automobile — **ship**

$[0,0.4,0,0,0,0,0,0,0.6,0]$ $\quad$ $[0,0,0,0,0,0,0,0,1,0]$

Loss function (Cross Entropy Loss)

$$L = -\sum_{i=1} y_i \log \hat{y}_i$$

$y_i = 1$ The higher the predicted probability the lower the loss you will get

$y_i = 0$ The whole term will be 0 so it doesn't matter

Loss Function

Measurement of model performance

The difference between the true label and predicted label

$\hat{y} \quad \leftrightarrow \quad y$

automobile **ship**

$[0,0.4,0,0,0,0,0,0,0.6,0]$ $[0,0,0,0,0,0,0,0,1,0]$

Loss function (Cross Entropy Loss)

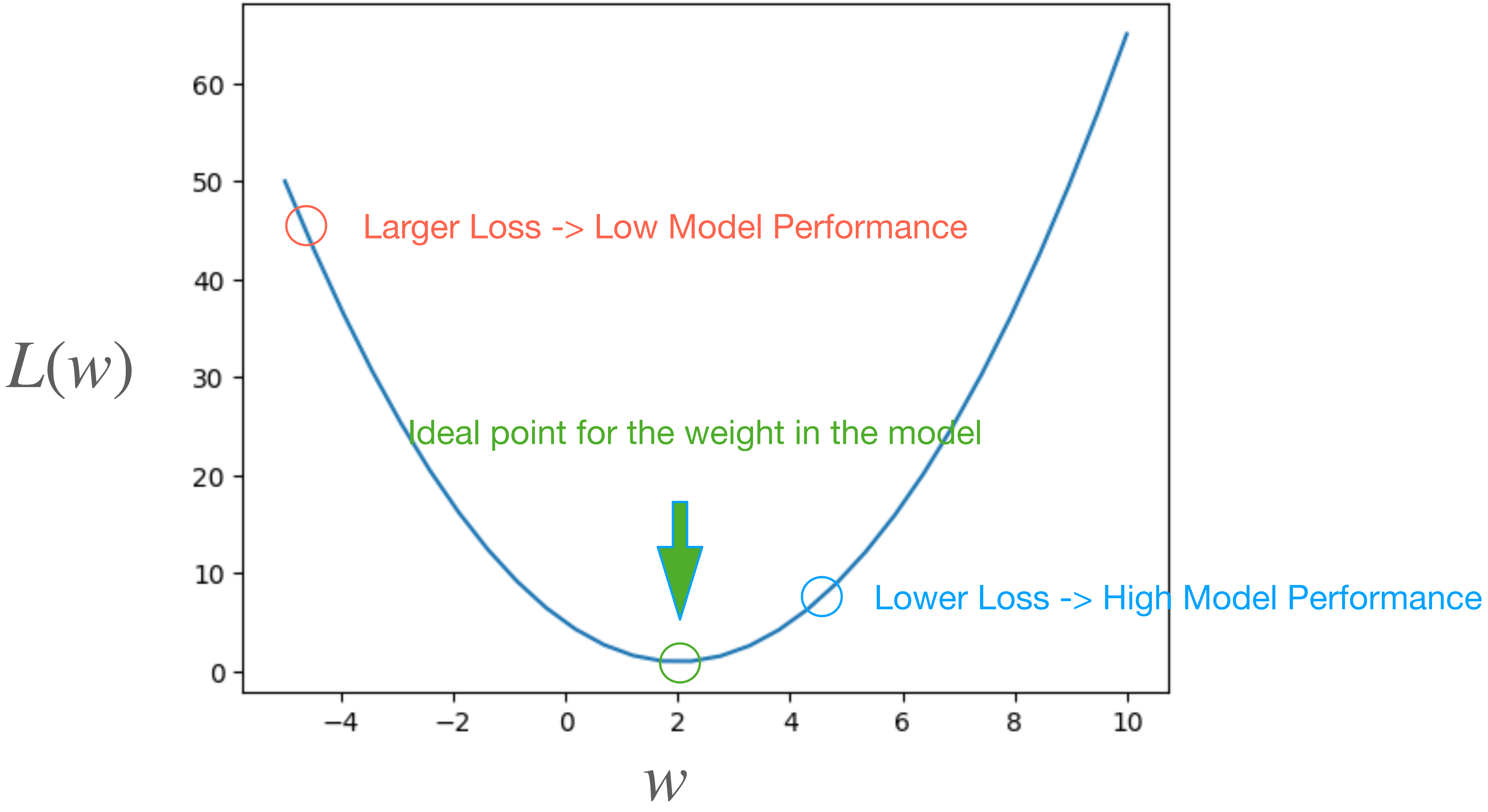
$$L = -\sum_{i=1} y_i \log \hat{y}_i$$

$$= -\sum_{i=1} y_i \log g(x_i)$$

$$L(w) \triangleq -\sum_{i=1} y_i \log(wx_i)$$

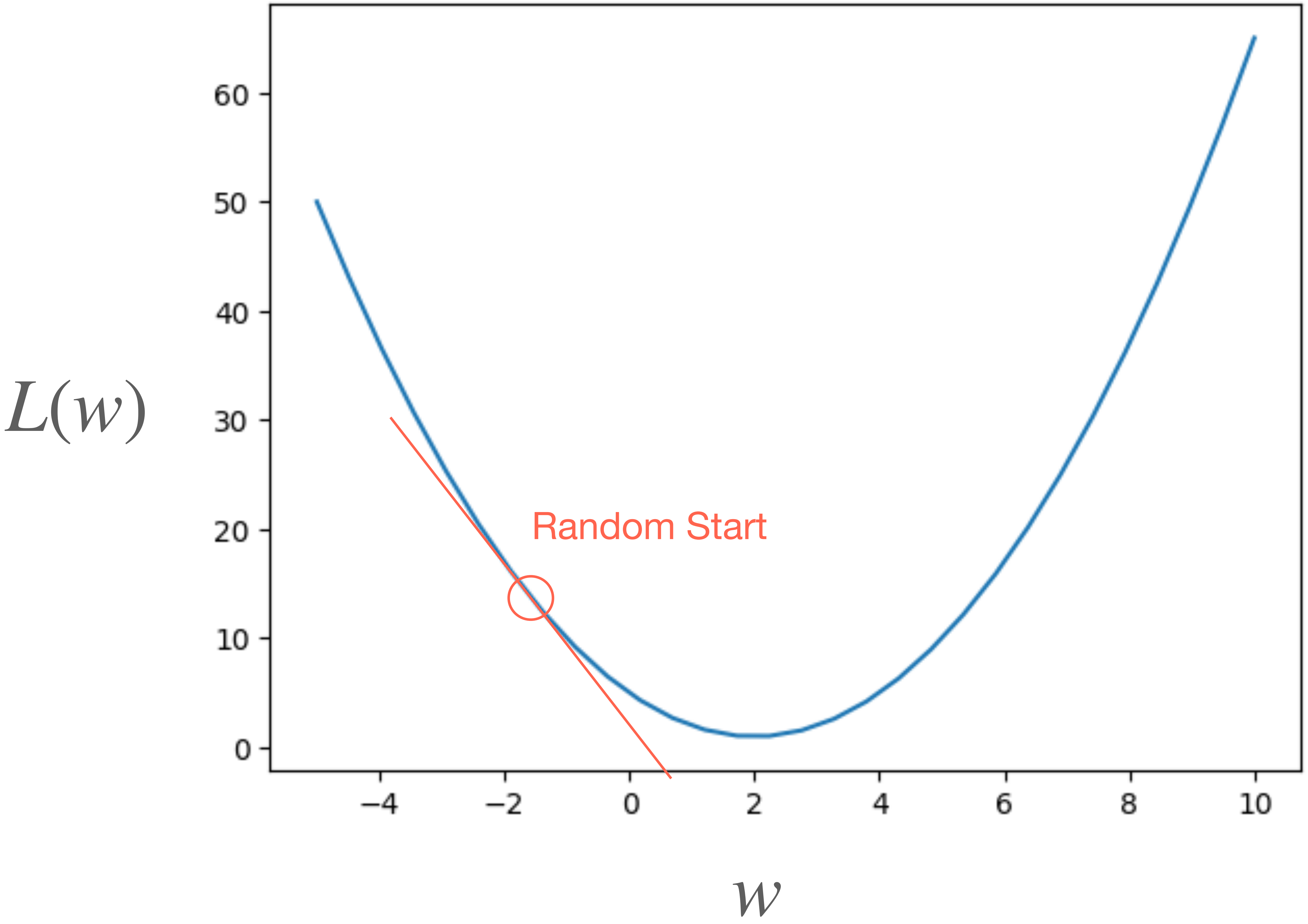
Loss Function

Measurement of model performance



Training Procedure

Start for Random Initialization of Weight



Intuition:
Walk to the lower point of the loss function

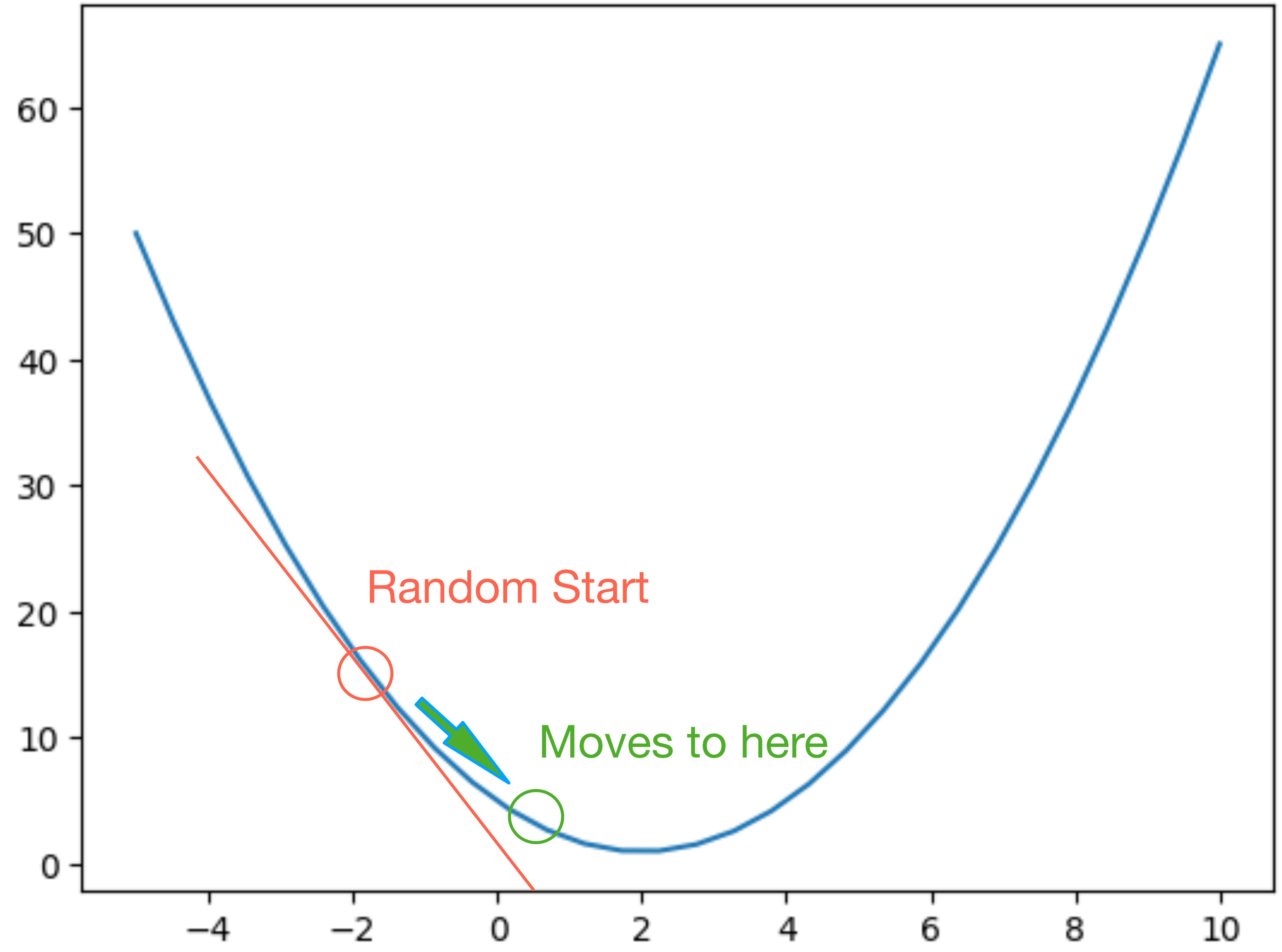
In mathematical language:
Find the direction where the loss will decrease
i.e. Slope < 0

$$Slope = \frac{L(w+h) - L(w)}{h}$$

$$\frac{dL}{dw} = \lim_{h \to 0} \frac{L(w+h) - L(w)}{h}$$

Gradient Descent

A Approach to lower the function value



$L(w)$

w

learning rate

$$w_t = w_{t-1} - \boxed{r} \underbrace{\frac{dL}{dw_{t-1}}}$$

If < 0, this term will be < 0 and w increases
If > 0, this term will be > 0 and w decreases

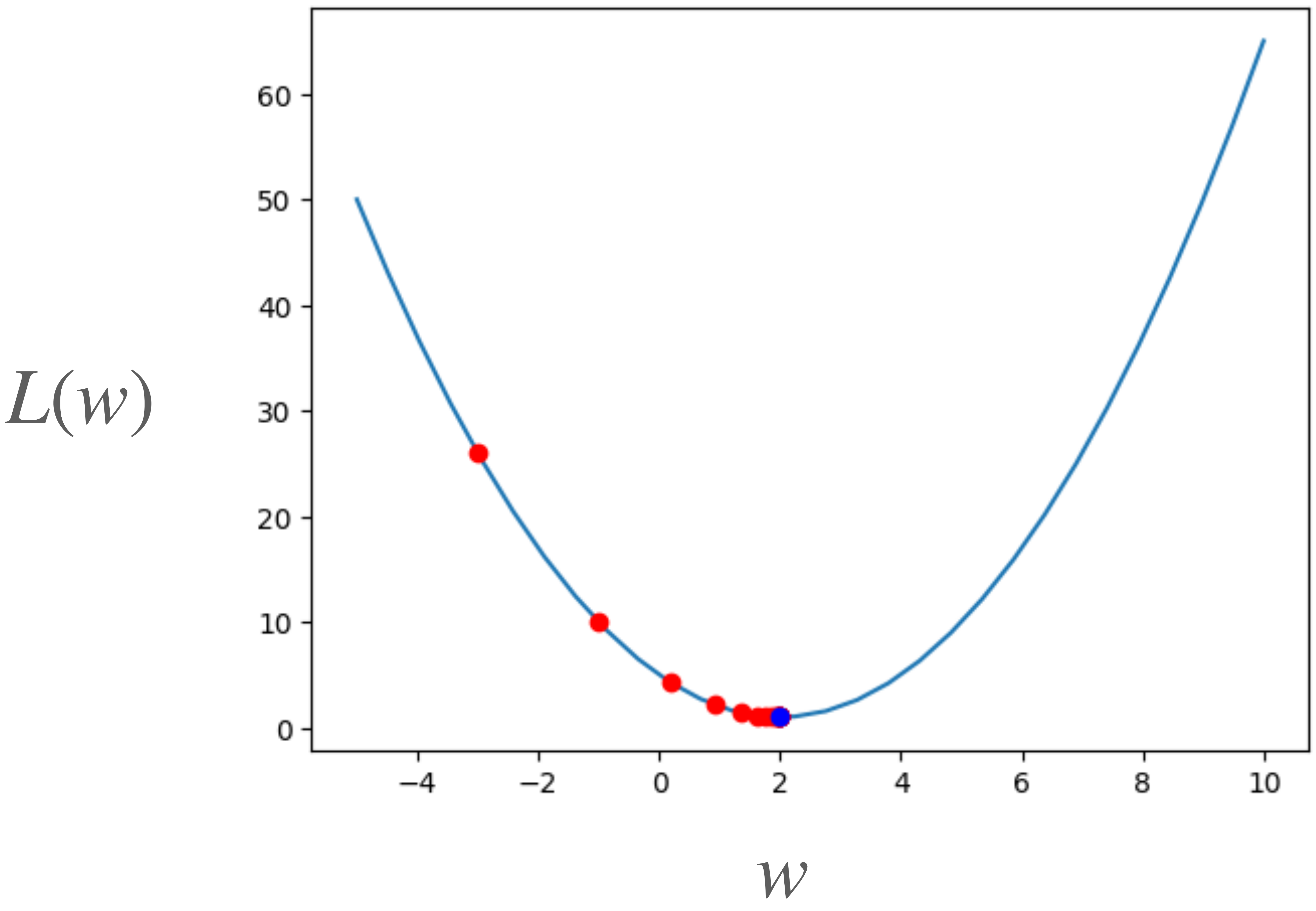
The procedure continues until some given threshold

$$|L(w_t) - L(w_{t-1})| < \epsilon$$

or given update steps t

Gradient Descent

A Approach to lower the function value



$$w_t = w_{t-1} - \overset{\text{learning rate}}{\boxed{r}}\underbrace{\frac{dL}{dw_{t-1}}}$$

If < 0, this term will be < 0 and w increases
If > 0, this term will be > 0 and w decreases

The procedure continues until some given threshold

$$|L(w_t) - L(w_{t-1})| < \epsilon$$

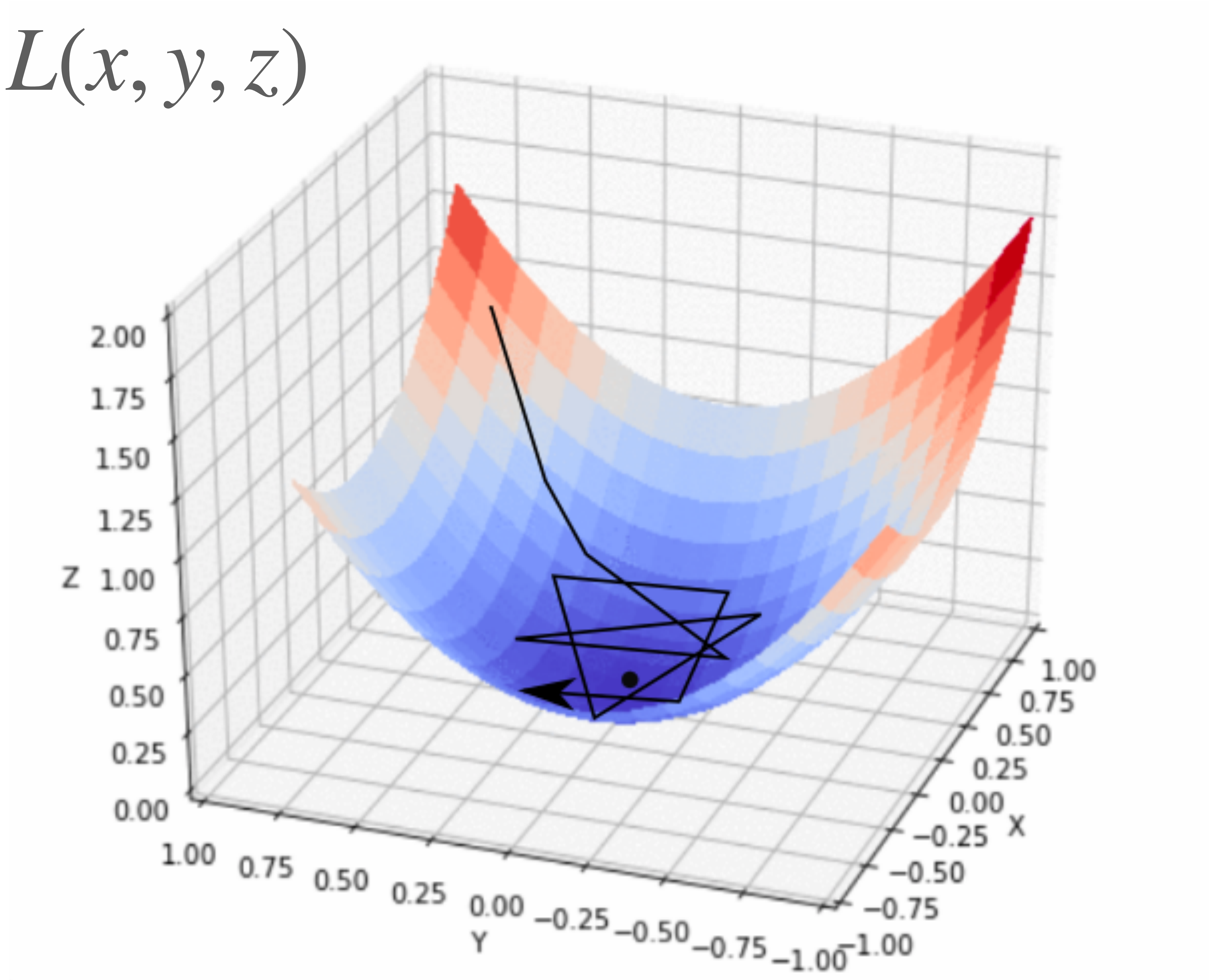
or given update steps t

Gradient Descent

A High-Dimension Overview

Update the three parameters at the same time!

$L(x, y, z)$



$$x_t = x_{t-1} - r\frac{\partial L(x, y, z)}{\partial x_{t-1}}$$

$$y_t = y_{t-1} - r\frac{\partial L(x, y, z)}{\partial y_{t-1}}$$

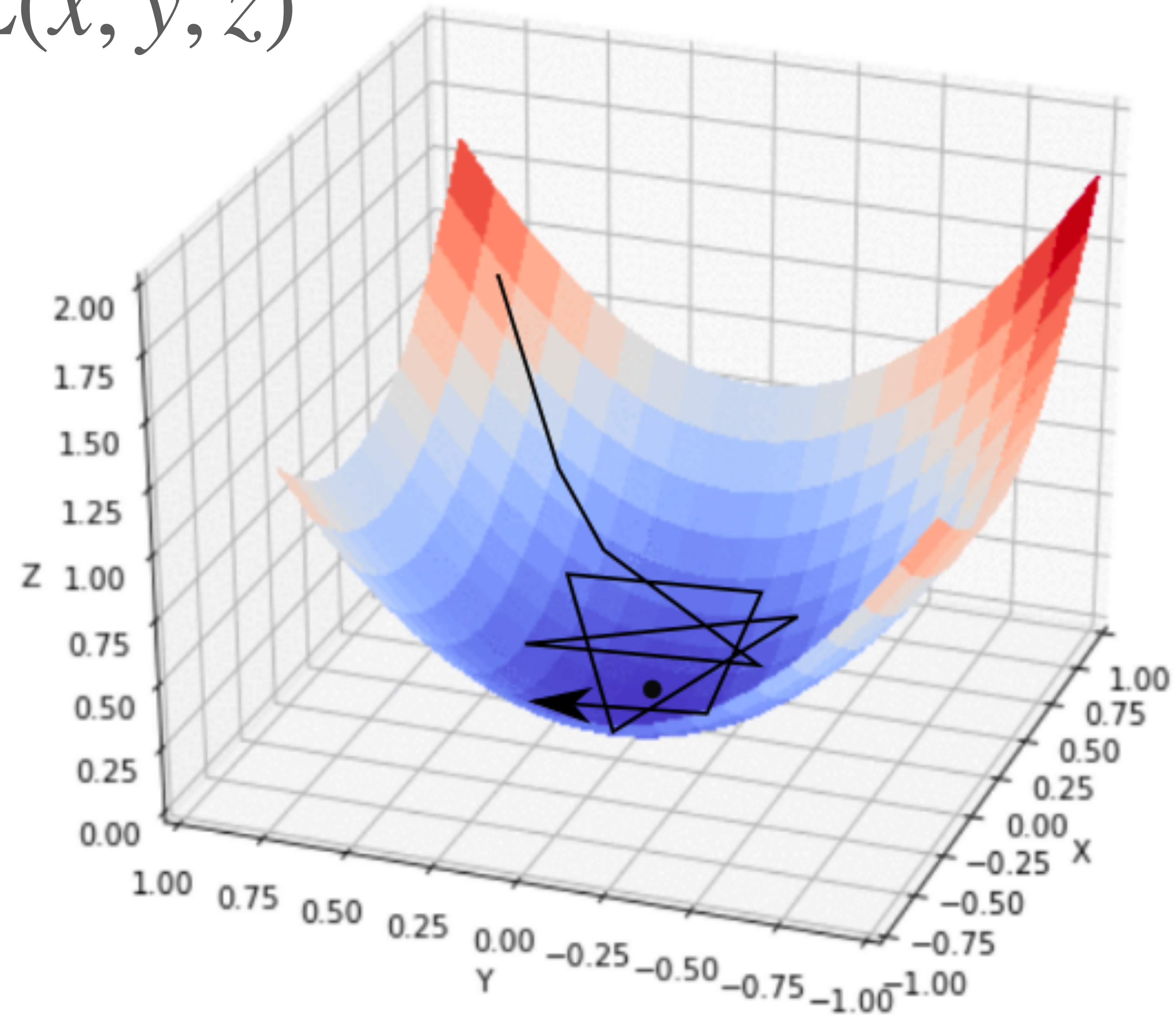
$$z_t = z_{t-1} - r\frac{\partial L(x, y, z)}{\partial z_{t-1}}$$

https://blog.paperspace.com/intro-to-optimization-in-deep-learning-gradient-descent/

Gradient Descent

A High-Dimension Overview

$L(x, y, z)$



https://blog.paperspace.com/intro-to-optimization-in-deep-learning-gradient-descent/

$$\boldsymbol{w} = [x, y, z]$$

$$\boldsymbol{w_t} = \boldsymbol{w_{t-1}} - r\nabla L(x, y, z)$$

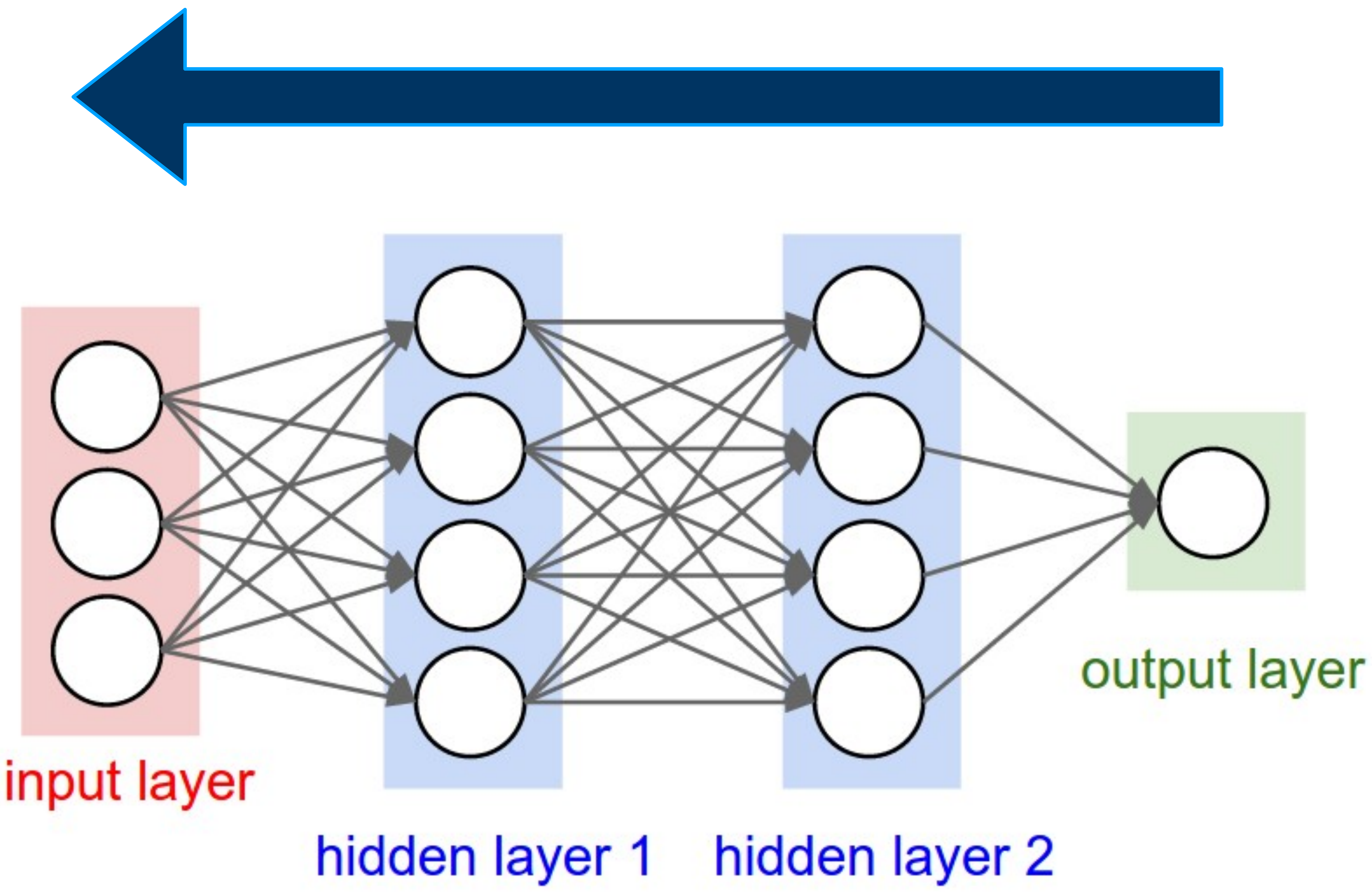
This is called gradient, which makes this method called **gradient descent**!

$$\begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix} = \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ z_{t-1} \end{bmatrix} - r \begin{bmatrix} \frac{\partial L}{x_{t-1}} \\ \frac{\partial L}{y_{t-1}} \\ \frac{\partial L}{z_{t-1}} \end{bmatrix}$$

Backpropogation

A High-Dimension Overview

Use loss to update the weights in backward order



https://cs231n.github.io/convolutional-networks/

$$w = [x, y, z]$$

$$w_t = w_{t-1} - r \nabla L(x, y, z)$$

$$\begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix} = \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ z_{t-1} \end{bmatrix} - r \begin{bmatrix} \frac{\partial L}{x_{t-1}} \\ \frac{\partial L}{y_{t-1}} \\ \frac{\partial L}{z_{t-1}} \end{bmatrix}$$

Let's do this in [Colab](#)!

Demo

Bi-Gram Model

A Very Simple Language Model

- We only predict the next word based on the previous word.
- Model the predicted probability of a certain word based on a given word.

$$P(c_i \mid c_{i-1})$$

c_i is the word in the i position.

- Here we give an example of a sentence:

<s> AI could finally be introduced into practice in general tasks <e>

Bi-Gram Model

A Very Simple Language Model

<s> AI could finally be introduced into practice in general tasks <e>

c_{i-1}	c_i
<s>	AI

Bi-Gram Model

A Very Simple Language Model

<s> AI could finally be introduced into practice in general tasks <e>

c_{i-1} c_i

AI could

Bi-Gram Model

A Very Simple Language Model

<s> AI could finally be introduced into practice in general $\boxed{\text{tasks <e>}}$

$$c_{i-1} \qquad c_i$$

tasks <e>

Bi-Gram Model

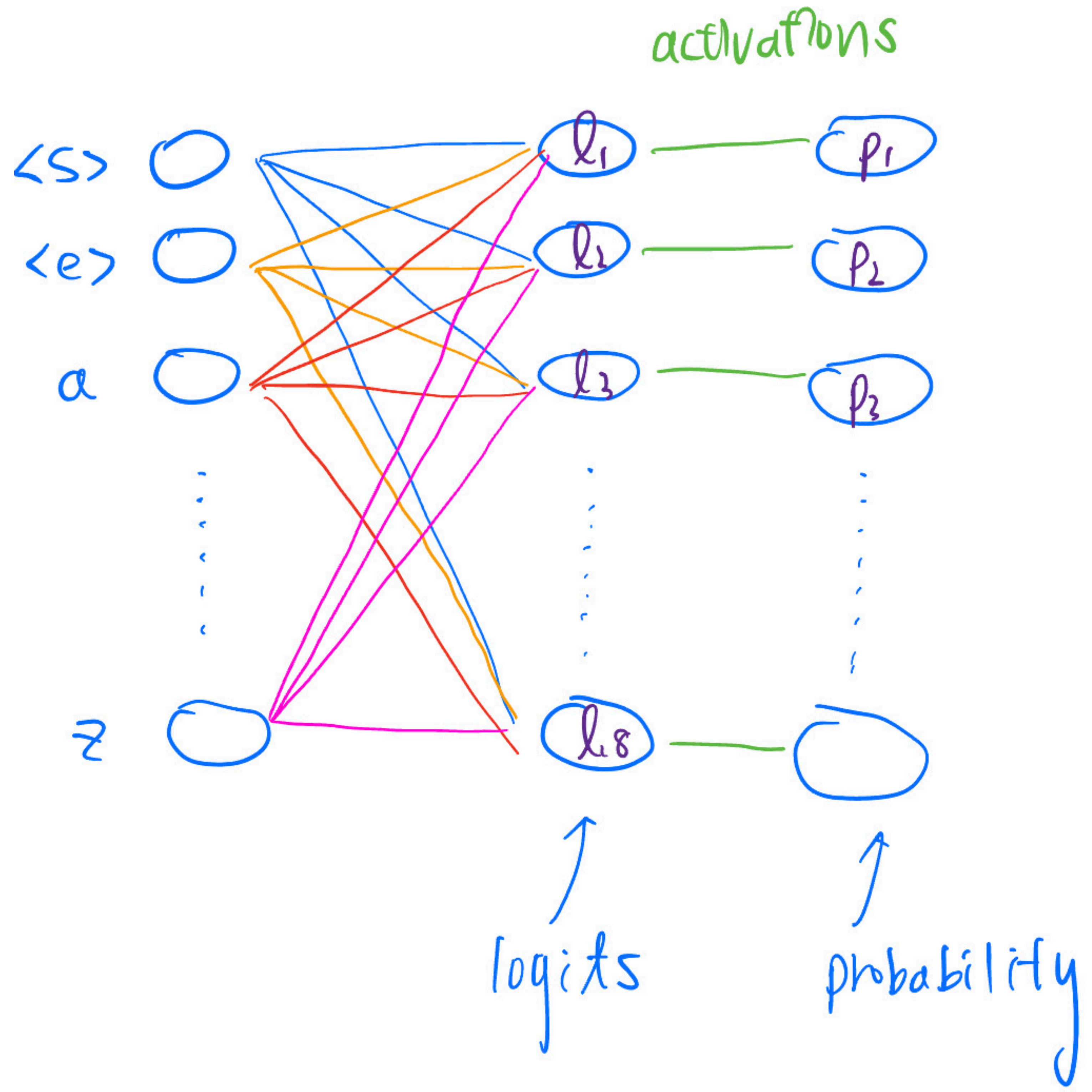
A Very Simple Language Model

We want to train the following neural network

One-hot Encoding

$$v_{<s>} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$v_{<z>} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$



$$p_i = \frac{e^{l_i}}{\sum_{i=1}^{28} e^{l_i}}$$

Let's also do this in Colab!

Questions

- What if having too large/small learning rate (r)?
- What if we cannot cover all the cases so that some conditional probability is zero?
- What if we also consider the penalty of the loss of predicting the class that does not contain the ground truth label?

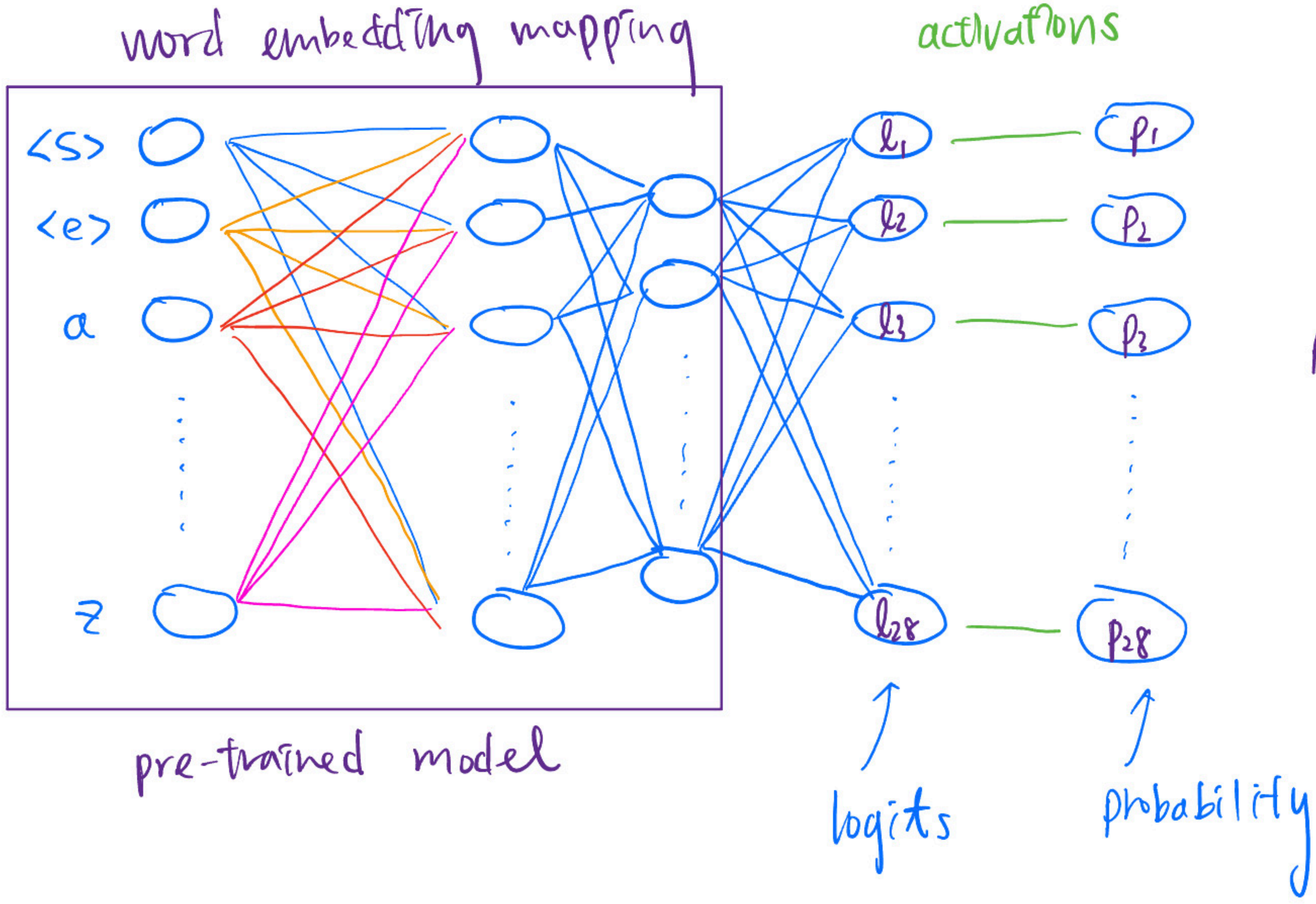
Wrap Up

What We Have Gone Through

- Applications in machine learning
- Training a machine learning model
- Gradient descent
- Bi-gram language model

What's Next

- Deep Neural Network
- Word Embedding
- Transfer Learning



$$p_i = \frac{e^{l_i}}{\sum_{i=1}^{28} e^{l_i}}$$

Q & A