

Bài 1: Giới thiệu về học sâu

Thế nào là học sâu?

- Là phương pháp học máy sử dụng mạng nơ-ron nhân tạo để trích xuất đặc trưng tự động từ dữ liệu

ARTIFICIAL INTELLIGENCE

Any technique that enables computers to mimic human behavior



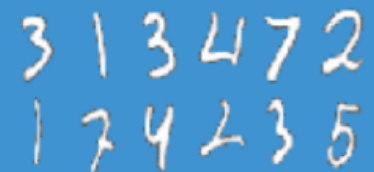
MACHINE LEARNING

Ability to learn without explicitly being programmed



DEEP LEARNING

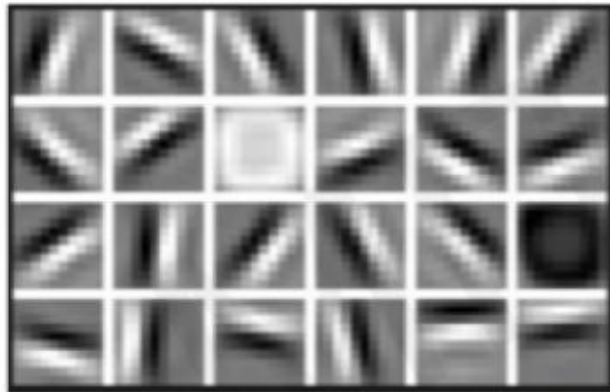
Extract patterns from data using neural networks



Tại sao cần học sâu?

- Phương pháp học máy truyền thống đòi hỏi trích xuất đặc trưng một cách thủ công, đòi hỏi kinh nghiệm và phụ thuộc từng bài toán cụ thể
- Học sâu cho phép trích chọn đặc trưng tự động từ dữ liệu

Low Level Features



Lines & Edges

Mid Level Features



Eyes & Nose & Ears

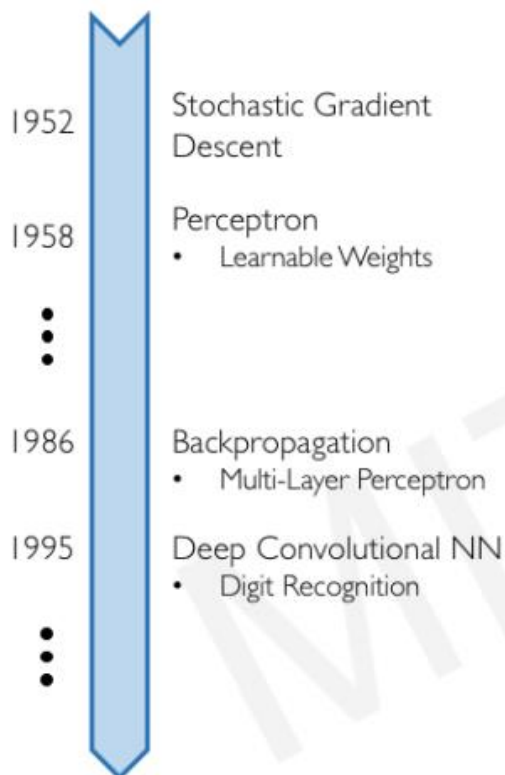
High Level Features



Facial Structure

Tại sao giờ mới bùng nổ học sâu?

Neural Networks date back decades, so why the resurgence?



1. Big Data

- Larger Datasets
- Easier Collection & Storage

IMAGENET



2. Hardware

- Graphics Processing Units (GPUs)
- Massively Parallelizable



3. Software

- Improved Techniques
- New Models
- Toolboxes



Học máy có giám sát

Functions \mathcal{F}

$$f : \mathcal{X} \rightarrow \mathcal{Y}$$

Training data

$$\{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}$$

LEARNING

$$\begin{array}{l} \text{find } \hat{f} \in \mathcal{F} \\ \text{s.t. } y_i \approx \hat{f}(x_i) \end{array}$$



Learning machine

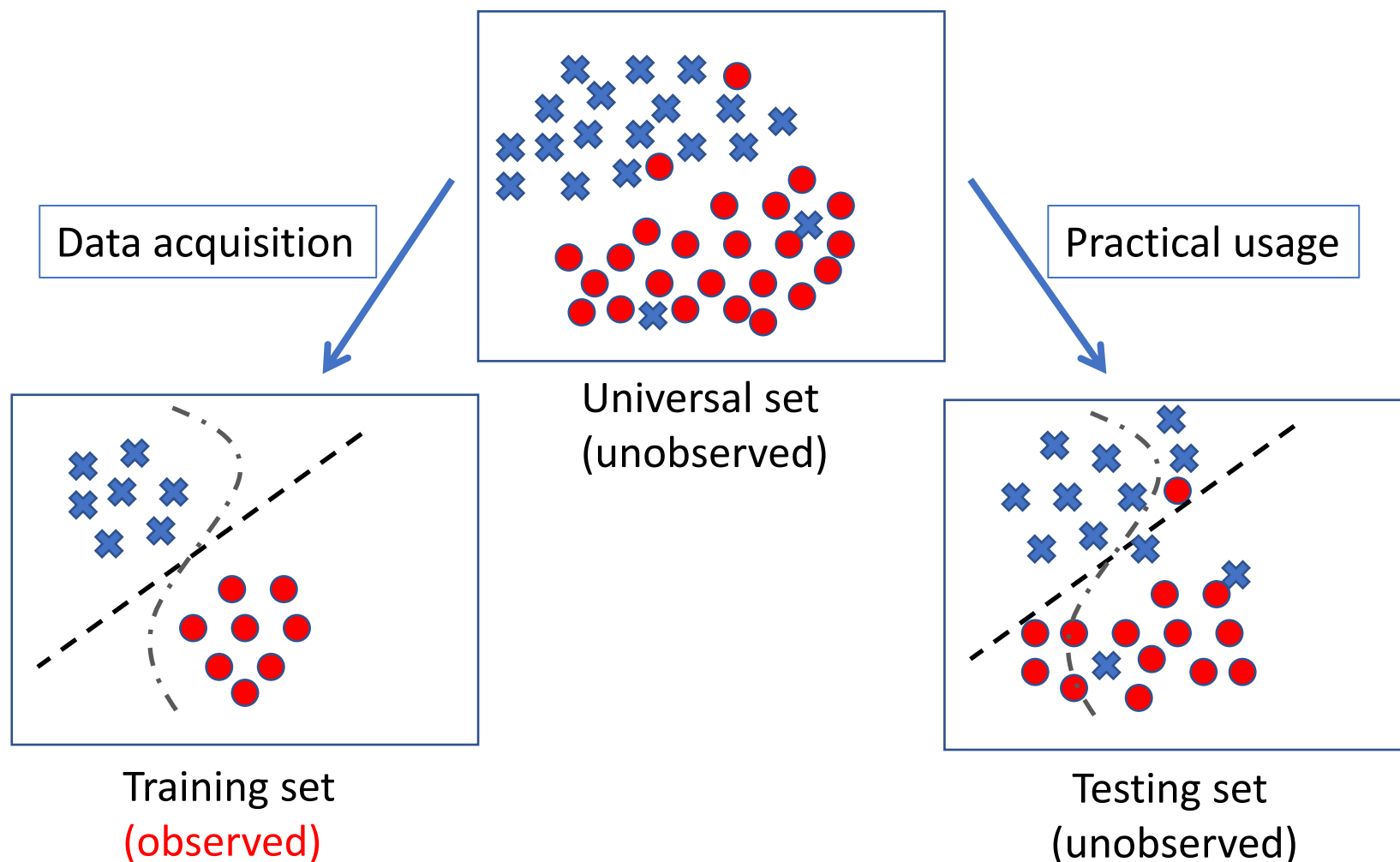
PREDICTION

$$y = \hat{f}(x)$$

New data

$$x$$

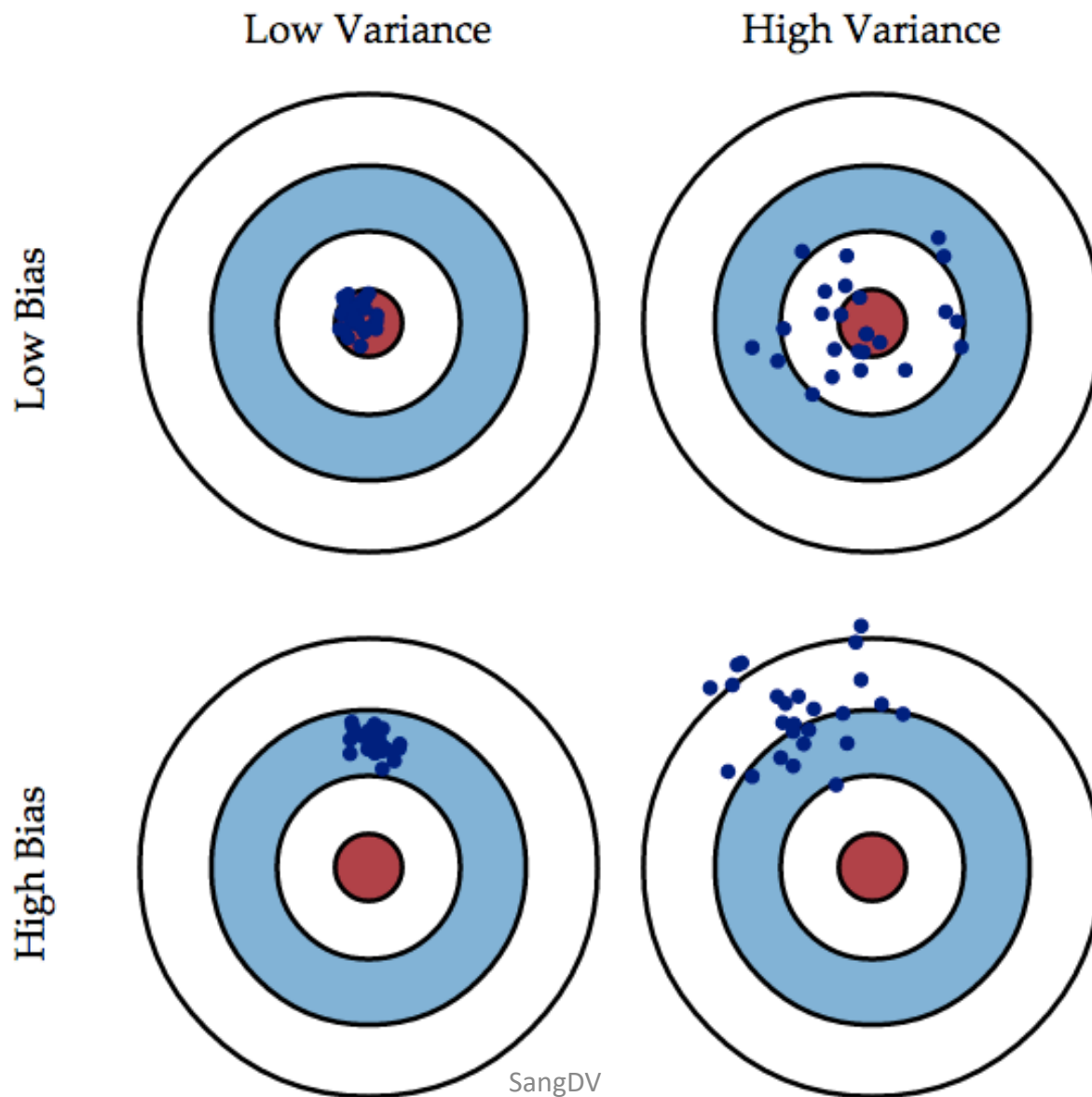
Tập huấn luyện và tập kiểm tra



Hiện tượng overfit và underfit

- Underfitting: mô hình quá “đơn giản” để biểu diễn các tính chất của dữ liệu
 - Bias cao và variance thấp
 - Sai số cao trên tập huấn luyện và tập kiểm tra
- Overfitting: mô hình quá “phức tạp” dẫn tới học cả nhiễu trong dữ liệu
 - Bias thấp và variance cao
 - Sai số thấp trên tập huấn luyện và sai số cao trên tập kiểm tra

Minh họa Bias-Variance



Phân lớp tuyến tính

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



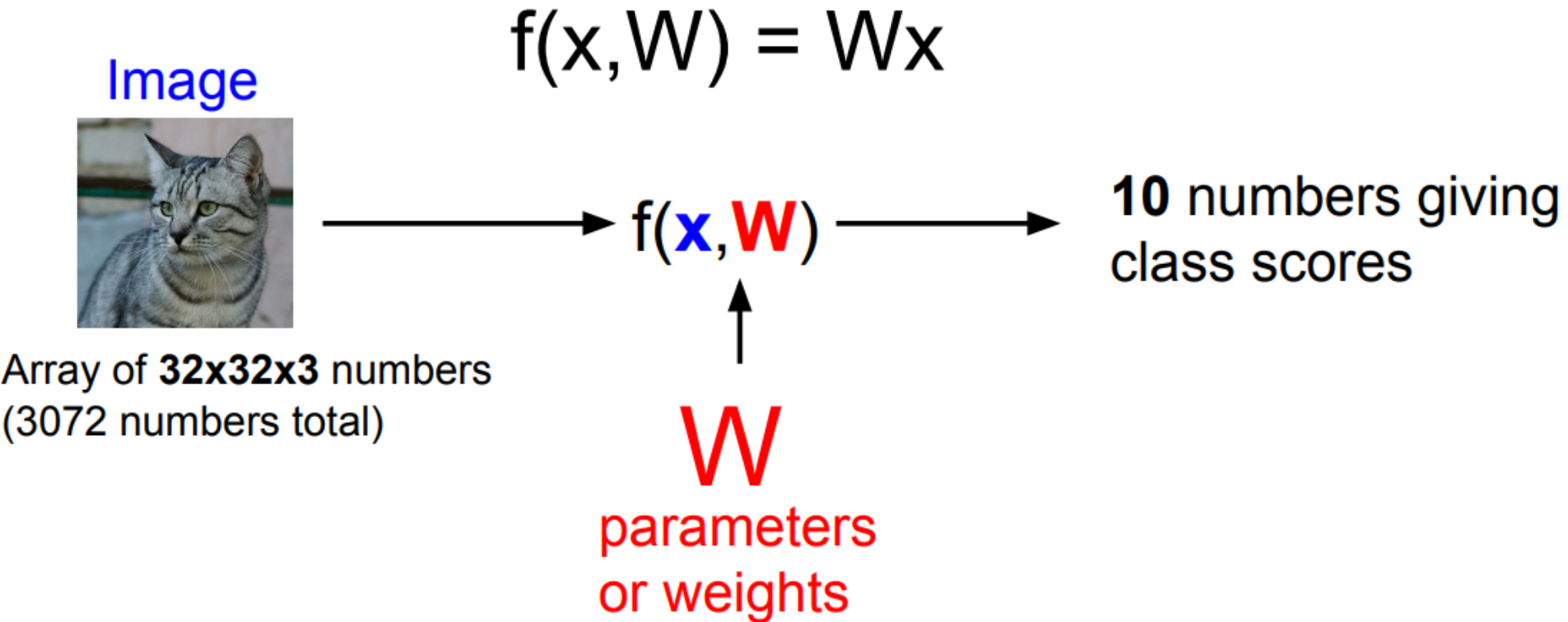
truck



50,000 training images
each image is **32x32x3**

10,000 test images.

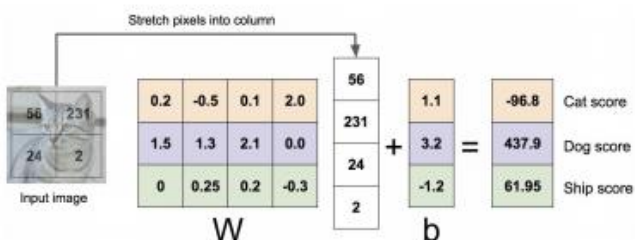
Phân lớp tuyến tính



Phân lớp tuyến tính: 3 góc nhìn

Algebraic Viewpoint

$$f(x, W) = Wx$$



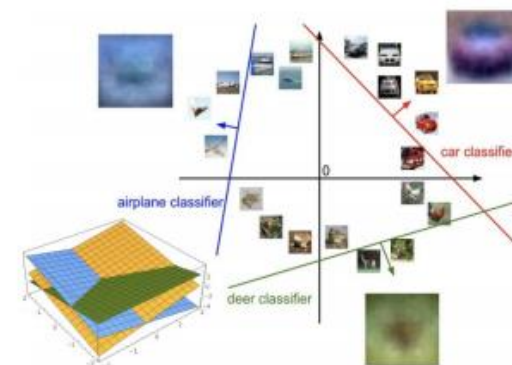
Visual Viewpoint

One template
per class



Geometric Viewpoint

Hyperplanes
cutting up space



Hàm mục tiêu

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



| | | | |
|------|------------|------------|-------------|
| cat | 3.2 | 1.3 | 2.2 |
| car | 5.1 | 4.9 | 2.5 |
| frog | -1.7 | 2.0 | -3.1 |

A **loss function** tells how good our current classifier is

Given a dataset of examples

$$\{(x_i, y_i)\}_{i=1}^N$$

Where x_i is image and
 y_i is (integer) label

Loss over the dataset is a
average of loss over examples:

$$L = \frac{1}{N} \sum_i L_i(f(x_i, W), y_i)$$

Hàm mục tiêu

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



| | | | |
|------|------------|------------|-------------|
| cat | 3.2 | 1.3 | 2.2 |
| car | 5.1 | 4.9 | 2.5 |
| frog | -1.7 | 2.0 | -3.1 |

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the
scores vector: $s = f(x_i, W)$

the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \begin{cases} 0 & \text{if } s_{y_i} \geq s_j + 1 \\ s_j - s_{y_i} + 1 & \text{otherwise} \end{cases}$$

$$= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Điều khiển quá trình huấn luyện



λ = regularization strength
(hyperparameter)

$$L(W) = \underbrace{\frac{1}{N} \sum_{i=1}^N L_i(f(x_i, W), y_i)}_{\text{Data loss}} + \underbrace{\lambda R(W)}_{\text{Regularization}}$$

Data loss: Model predictions should match training data

Regularization: Prevent the model from doing *too* well on training data

Simple examples

L2 regularization: $R(W) = \sum_k \sum_l W_{k,l}^2$

L1 regularization: $R(W) = \sum_k \sum_l |W_{k,l}|$

Elastic net (L1 + L2): $R(W) = \sum_k \sum_l \beta W_{k,l}^2 + |W_{k,l}|$

More complex:

Dropout

Batch normalization

Stochastic depth, fractional pooling, etc

Bộ phân loại softmax



Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

$$P(Y = k|X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}} \quad \text{Softmax Function}$$

Probabilities
must be ≥ 0

Probabilities
must sum to 1

$$L_i = -\log P(Y = y_i|X = x_i)$$

cat
car
frog

3.2
5.1
-1.7

Unnormalized
log-probabilities / logits

exp

24.5
164.0
0.18

unnormalized
probabilities

normalize

0.13
0.87
0.00

probabilities

compare

Kullback-Leibler
divergence

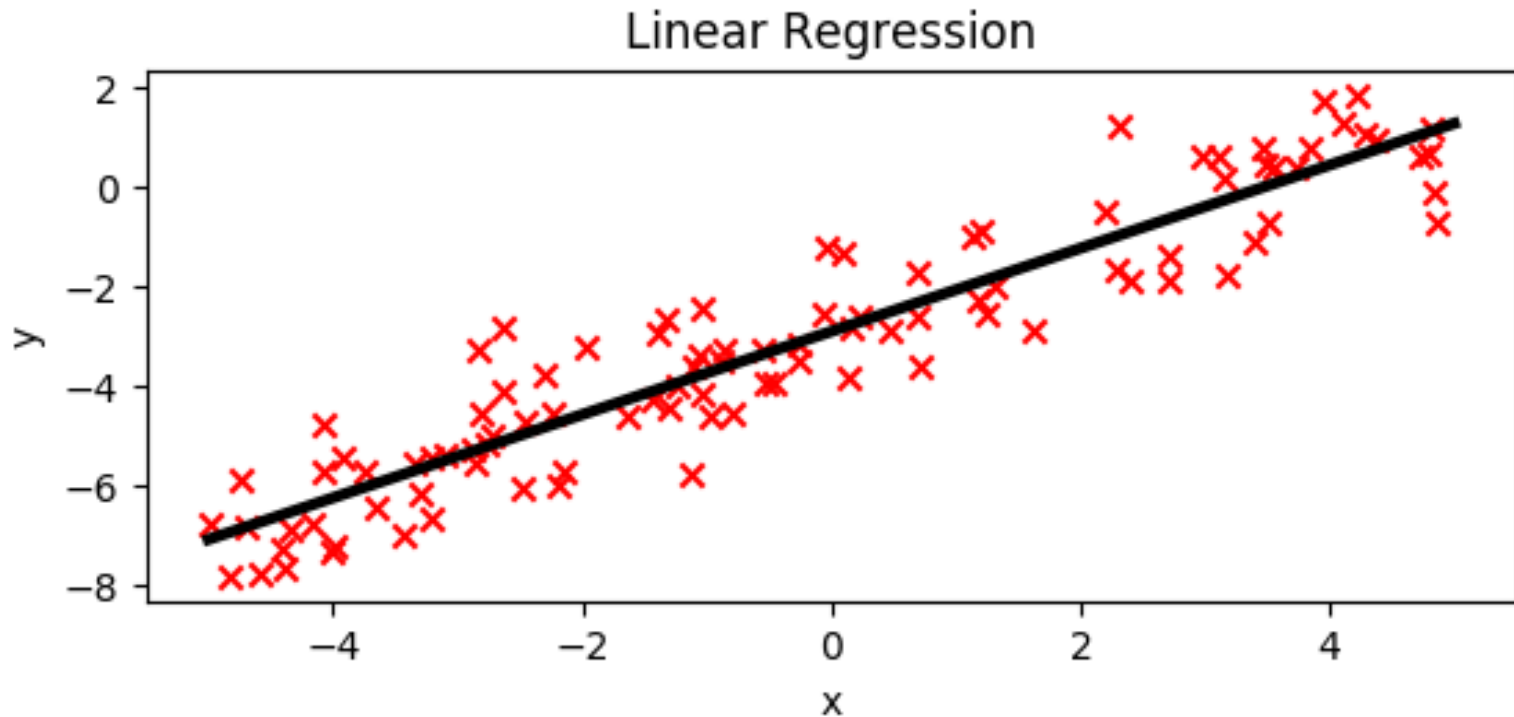
$$D_{KL}(P||Q) = \sum_y P(y) \log \frac{P(y)}{Q(y)}$$

1.00
0.00
0.00

Correct
probs

Hồi quy tuyến tính

- $f(x; w) = w_0 + \sum_{i=1}^d w_i x_i = w^T x'$
- Below: $d = 1$. $w^T x'$ is graphed.



Hồi quy tuyến tính

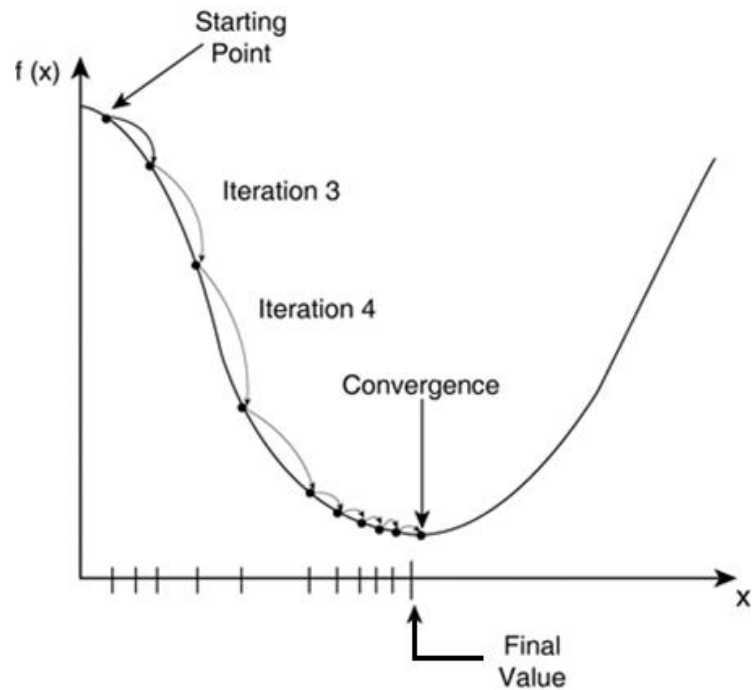
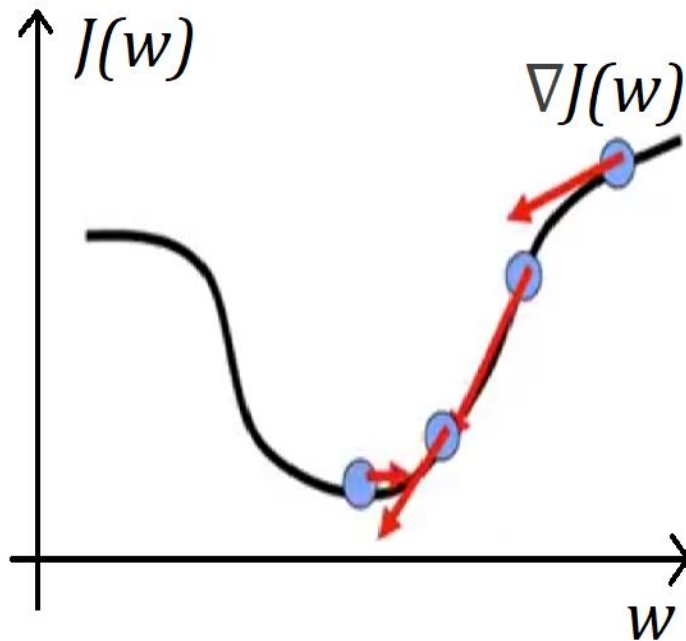
- Nên chọn hàm mục tiêu nào?
 - Mỗi $y^{(i)}$ là một số thực
 - Bình phương tối thiểu là một lựa chọn tốt ☺

$$\begin{aligned} \bullet J(w; \mathbf{X}, \mathbf{Y}) &= \frac{1}{N} \sum_{i=1}^N [f(x^{(i)}; w) - y^{(i)}]^2 \\ &= \frac{1}{N} \sum_{i=1}^N [w^T x^{(i)'} - y^{(i)}]^2 \\ &= \frac{1}{N} (w^T \mathbf{X}' - \mathbf{Y})^T (w^T \mathbf{X}' - \mathbf{Y}) \end{aligned}$$

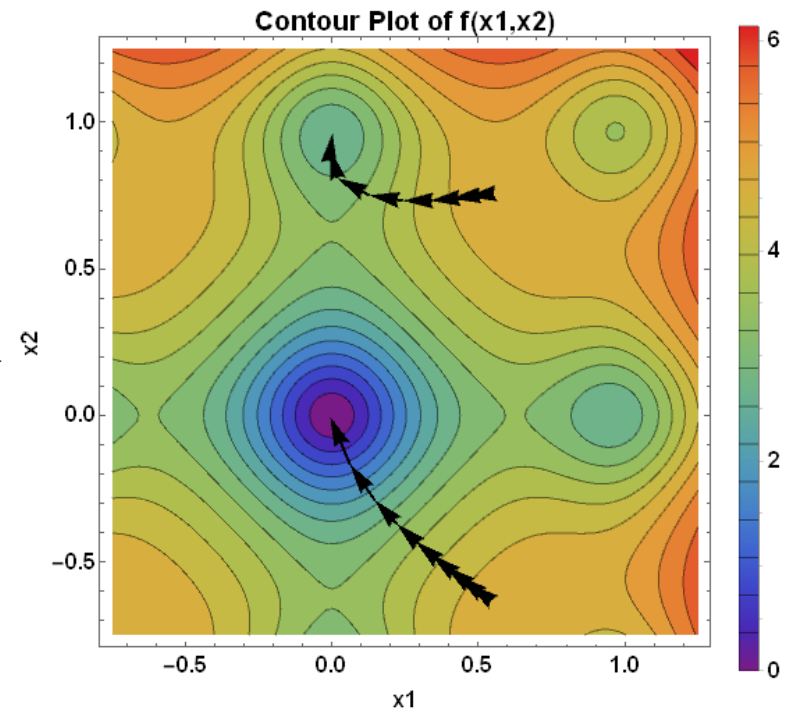
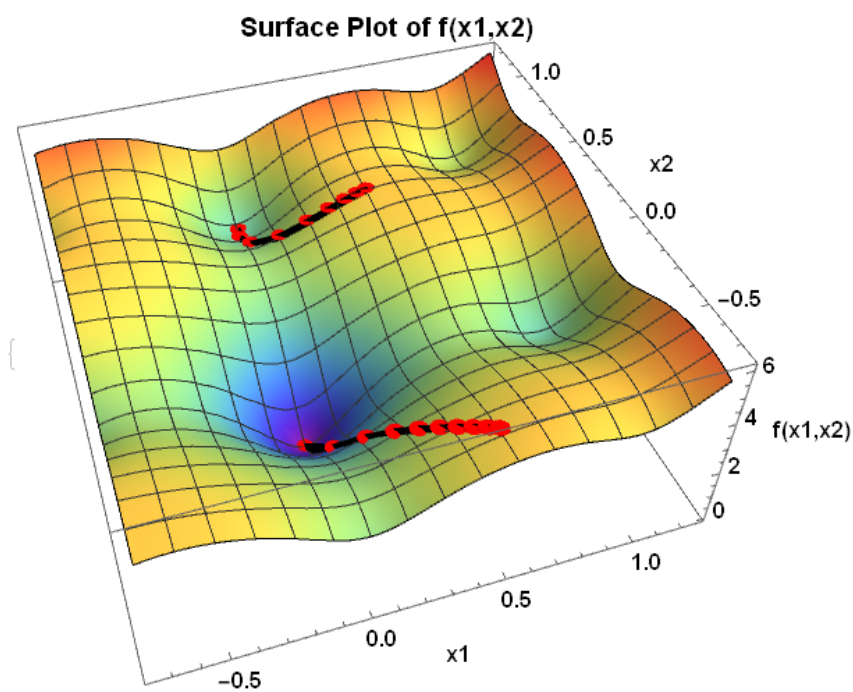
Tối ưu hàm mục tiêu



Gradient Descent



Gradient Descent



Gradient Descent

- Chọn tốc độ học learning rate η
- Khởi tạo w ngẫu nhiên
 - Khởi tạo w từ các phân bố thường gặp như phân bố đều hoặc phân bố chuẩn (gauss)
- Chừng nào w vẫn chưa hội tụ
 - Cập nhật $w \leftarrow w - \eta \nabla J(w; \mathbf{X}, \mathbf{Y})$

Stochastic Gradient Descent (SGD)



$$L(W) = \frac{1}{N} \sum_{i=1}^N L_i(x_i, y_i, W) + \lambda R(W)$$

Full sum expensive
when N is large!

$$\nabla_W L(W) = \frac{1}{N} \sum_{i=1}^N \nabla_W L_i(x_i, y_i, W) + \lambda \nabla_W R(W)$$

Approximate sum
using a **minibatch** of
examples
32 / 64 / 128 common

```
# Vanilla Minibatch Gradient Descent
```

```
while True:
```

```
    data_batch = sample_training_data(data, 256) # sample 256 examples
```

```
    weights_grad = evaluate_gradient(loss_fun, data_batch, weights)
```

```
    weights += - step_size * weights_grad # perform parameter update
```


Giới thiệu công cụ và môi trường

- Google Colab: <https://colab.research.google.com/>
- Miễn phí GPU (Tesla T4/P100)
- Dùng liên tục 12 tiếng mỗi session

The screenshot shows the Google Colab interface with the command `!nvidia-smi` executed in the terminal. The output displays the NVIDIA-SMI version, driver version, CUDA version, and a table of GPU information for a Tesla P100-PCIe...

```

Wed Feb 19 17:02:58 2020

+-----+
| NVIDIA-SMI 440.48.02      Driver Version: 418.67      CUDA Version: 10.1      |
+-----+-----+
| GPU   Name                Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan   Temp   Perf   Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
+-----+-----+
|    0   Tesla P100-PCIE...    Off      | 00000000:00:04:0 Off |              0      |
| N/A   38C    P0      28W / 250W |      0MiB / 16280MiB |           0%      Default |
+-----+-----+

+-----+
| Processes:                                     GPU Memory |
|  GPU       PID    Type    Process name                     Usage      |
+-----+-----+
| No running processes found                  |
+-----+
  
```

Google Colab



- Thiết lập GPU: Edit / Notebook settings

Notebook settings

Runtime type

Python 3



Hardware accelerator

GPU

☐

Omit code cell output when saving this notebook

CANCEL

SAVE

Google Colab

- Trick giúp colab chạy liên tục: Ấn F12, chọn Console

```
function ClickConnect(){
  console.log("Working");
  document.querySelector("colab-toolbar-button#connect").click()
}

setInterval(ClickConnect,60000)
```

File Edit View Insert Runtime Tools Help Last edited on February 20

+ Code + Text

Connect

Editing

!nvidia-smi

Wed Feb 19 17:02:58 2020

```
+-----+
| NVIDIA-SMI 440.48.02    Driver Version: 418.67      CUDA Version: 10.1     |
+-----+-----+
| GPU   Name               Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
+-----+-----+
|  0    Tesla P100-PCIE...    Off      | 00000000:00:04:0 Off |                    0 |
| N/A   38C    P0      28W / 250W |      0MiB / 16280MiB |      0%      Default |
+-----+-----+
```

```
+-----+
| Processes:                                     GPU Memory |
|  GPU       PID    Type    Process name                     Usage      |
+-----+-----+
| No running processes found.                                     |
+-----+
```

30/03/2020

SangDV

The screenshot shows the Chrome DevTools Console with the following content:

- Top panel: Shows the source map for the ClickConnect function, which is a self-referencing loop: `search.google.com/v2/external/js/min-maps/vs/loader.js.map`.
- Bottom panel: Shows the console log with the following messages:
 - Warning: DevTools failed to parse SourceMap: `https://colab.research.google.com/v2/external/js/min-maps/vs/editor/editor.main.js.map`
 - Warning: DevTools failed to parse SourceMap: `https://colab.research.google.com/v2/external/js/min-maps/vs/base/worker/workerMain.js.map`
 - Log: `> function ClickConnect(){ console.log("Working"); document.querySelector("colab-toolbar-button#connect").click() } setInterval(ClickConnect,60000)`

Google Colab

- Tạo nhiều tài khoản google
- Share dữ liệu cho nhiều tài khoản google cùng dùng
- Mount dữ liệu:

```
from google.colab import drive  
drive.mount('/content/drive')
```

- Lưu dữ liệu dưới dạng zip và unzip vào ổ cứng máy ảo Colab để tăng tốc độ xử lý dữ liệu:

```
!unzip -uq "/content/drive/My Drive/Colab Notebooks/data.zip" -d "/content/"
```

Jupyter Notebook

- Hướng dẫn sử dụng và cài đặt:

<https://www.dataquest.io/blog/jupyter-notebook-tutorial/>

The screenshot displays the Jupyter Notebook web interface. At the top, there's a 'jupyter' logo and a 'Logout' button. Below this, a navigation bar shows 'Files', 'Running', and 'Clusters' tabs. The 'Files' tab is active, showing a file browser for the 'Projects / jupyter-notebook' directory. It lists three files: '..' (parent directory), 'my-first-notebook.ipynb' (6 days ago), and 'notebook01.ipynb' (Running 3 minutes ago). The 'notebook01.ipynb' file is selected. Below the file browser, the notebook content is shown. It contains two code cells. The first cell, labeled 'In [1]:', contains the code `print('Hello World')` and has executed, showing the output 'Hello World'. The second cell, labeled 'In [2]:', contains a `while` loop that prints numbers from 1 to 10. The output of this cell is a list of numbers from 1 to 10. At the bottom, there is an input field for the next code cell, labeled 'In []:'.

Tensorflow/Keras/PyTorch

Introduction

Keras



TensorFlow



PyTorch

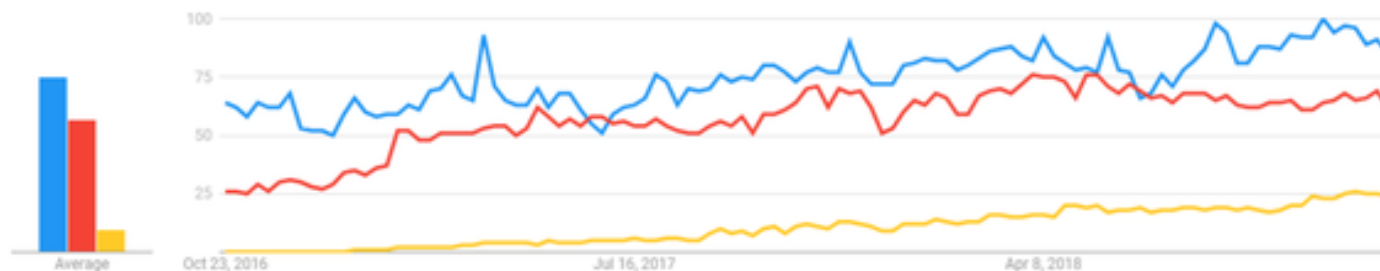


Popularity


● Keras

● TensorFlow

● PyTorch




Tensorflow/Keras/PyTorch


Keras


[+ Follow](#) [+ I use this](#)

| Stacks | Followers | Votes |
|--------|-----------|-------|
| 488 | 415 | 11 |


PyTorch


[+ Follow](#) [+ I use this](#)

| Stacks | Followers | Votes |
|--------|-----------|-------|
| 363 | 363 | 11 |


TensorFlow

[+ Follow](#) [+ I use this](#)

| Stacks | Followers | Votes |
|--------|-----------|-------|
| 1.4K | 1.4K | 62 |

 
488 24.5K

 
1.6K 5.7K

  
3.9K 4.1K 52.8K

   
47K 17.8K 3d

   
36.4K 9.2K 3d

   
141.3K 80K 4d