Hãy nói theo cách của bạn

TẬP ĐOÀN CÔNG NGHIỆP – VIỄN THÔNG QUÂN ĐỘI

TỔNG CÔNG TY CÔNG NGHIỆP CÔNG NGHỆ CAO VIETTEL

GIỚI THIỆU

TỔNG QUAN MACHINE LEARNING VÀ MỘT SỐ BÀI TOÁN IoT ANALYTICS

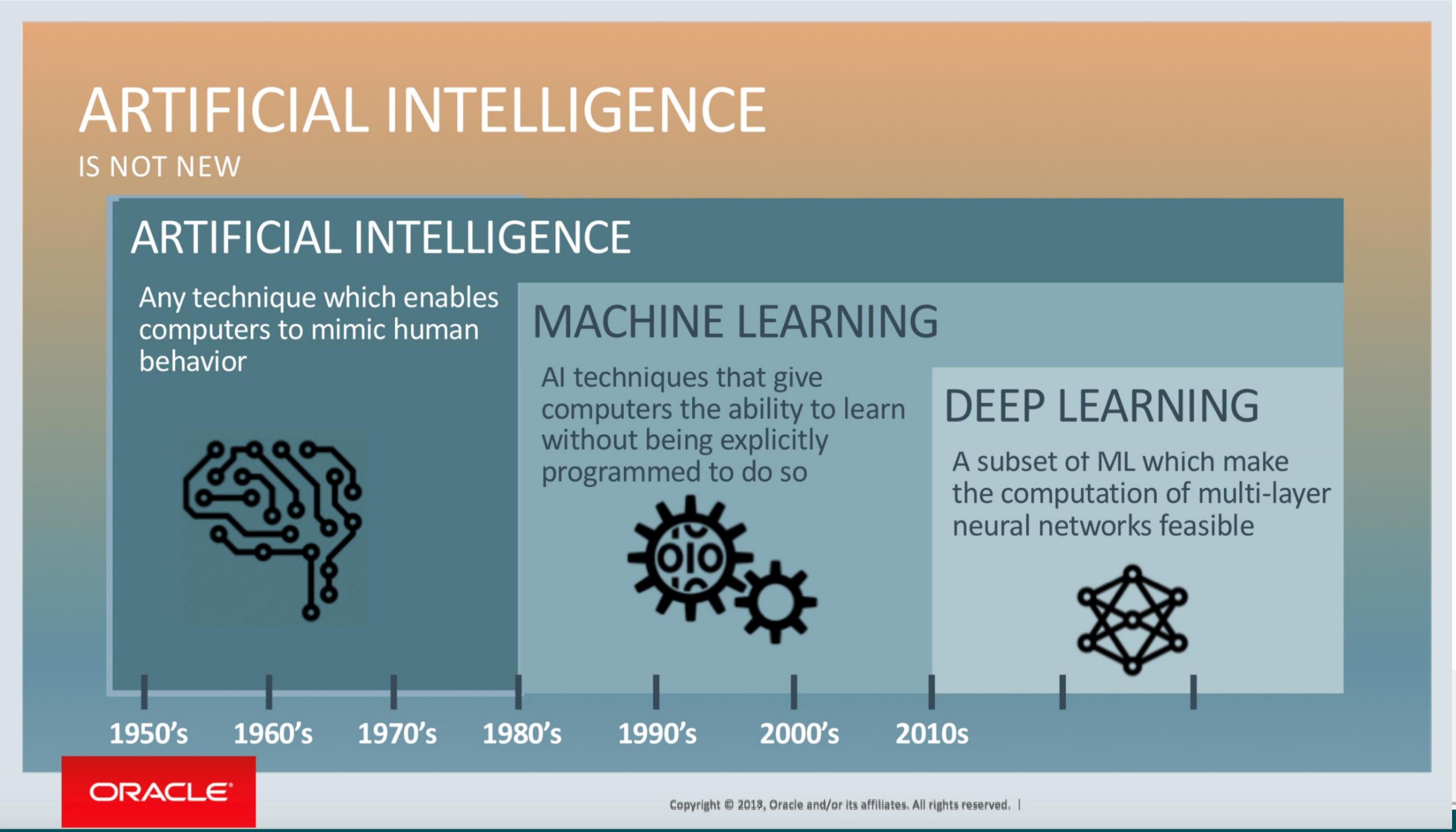
In This Presentation

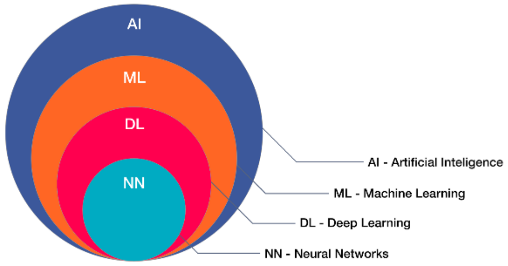
A. Tổng quan về Machine Learning & Deep Learning
B. Một số framework cho ML & DL
C. Một số bài toán, dataset cho IoT Data Analytics

A. TỔNG QUAN VỀ MACHINE LEARNING, DEEP LEARNING

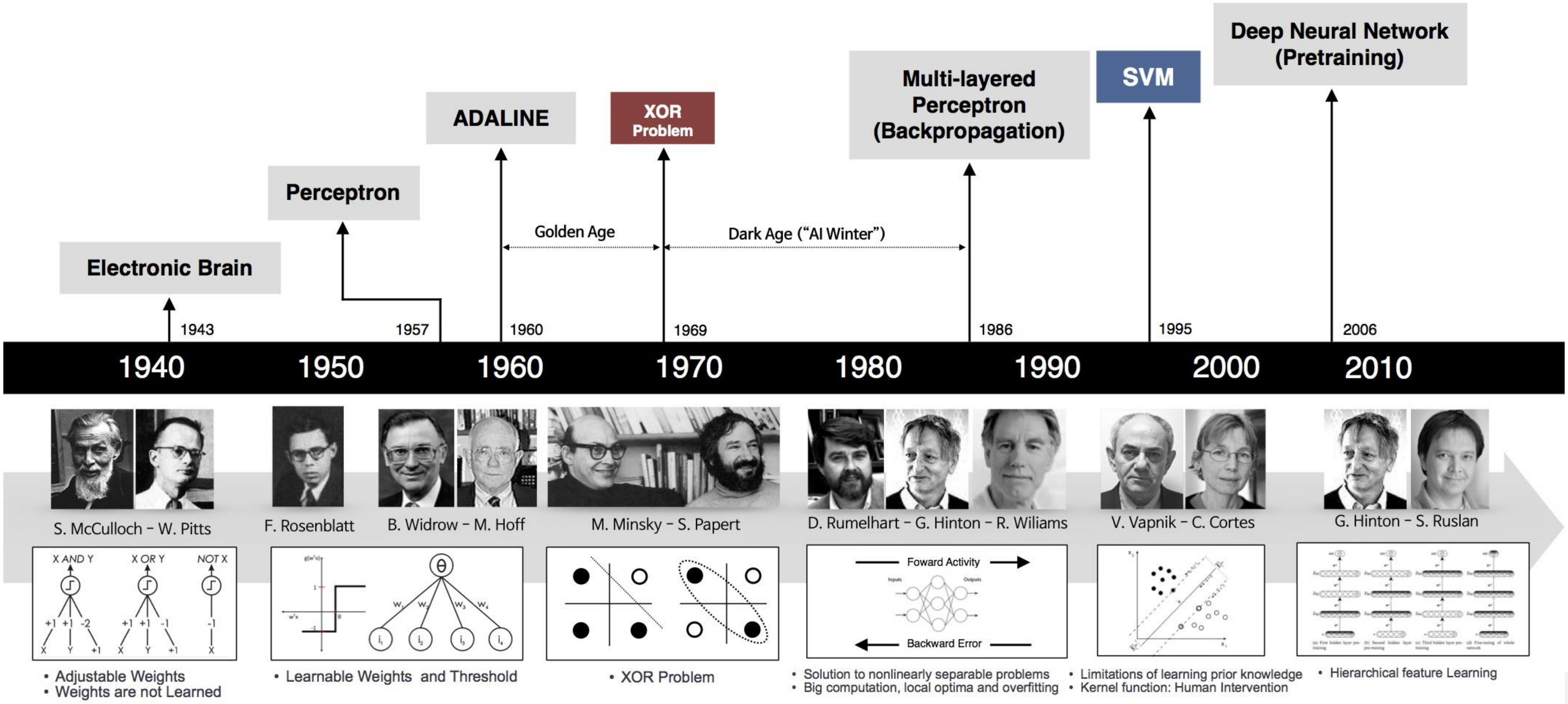
BIG PICTURE

THE BIG PICTURE





HISTORY OF MACHINE LEARNING, DEEP LEARNING



TỔNG QUAN VỀ MACHINE LEARNING

WHAT IS MACHINE LEARNING?

- Machine learning **teaches computers** to do what comes naturally to humans and animals: **learn from experience.**

- Machine learning algorithms **use computational methods to "learn" information directly from data without relying on a predetermined equation.**

Real-World Applications

With the rise in big data, machine learning has become particularly important for solving problems in areas like these:

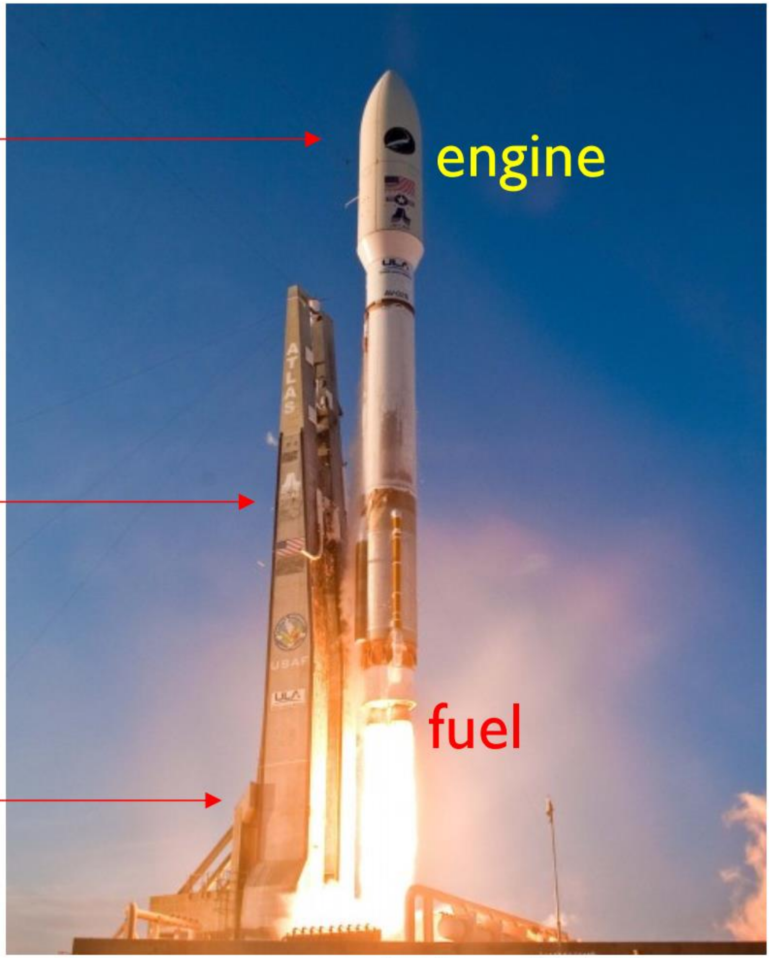
- Computational finance, for credit scoring and algorithmic trading
- Image processing and computer vision, for face recognition, motion detection, and object detection
- Computational biology, for tumor detection, drug discovery, and DNA sequencing
- Energy production, for price and load forecasting
- Automotive, aerospace, and manufacturing, for predictive maintenance
- Natural language processing

KEY FACTORS OF MACHINE LEARNING

Computation models,
e.g., deep learning/AI

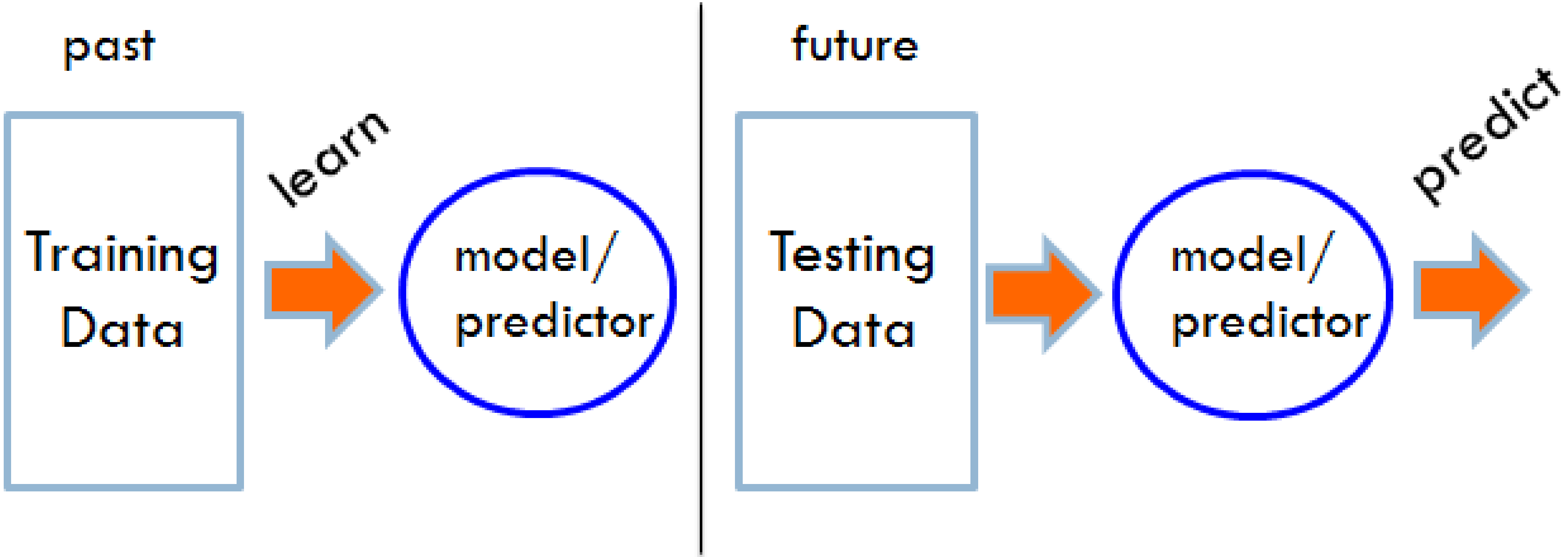
Computation infrastructure,
AI talent,
environment and enterprise policy
as the ramp

Data is energy.
Big data is the
huge energy source



Andrew Ng's Analogy

TRAINING AND TESTING PHASE

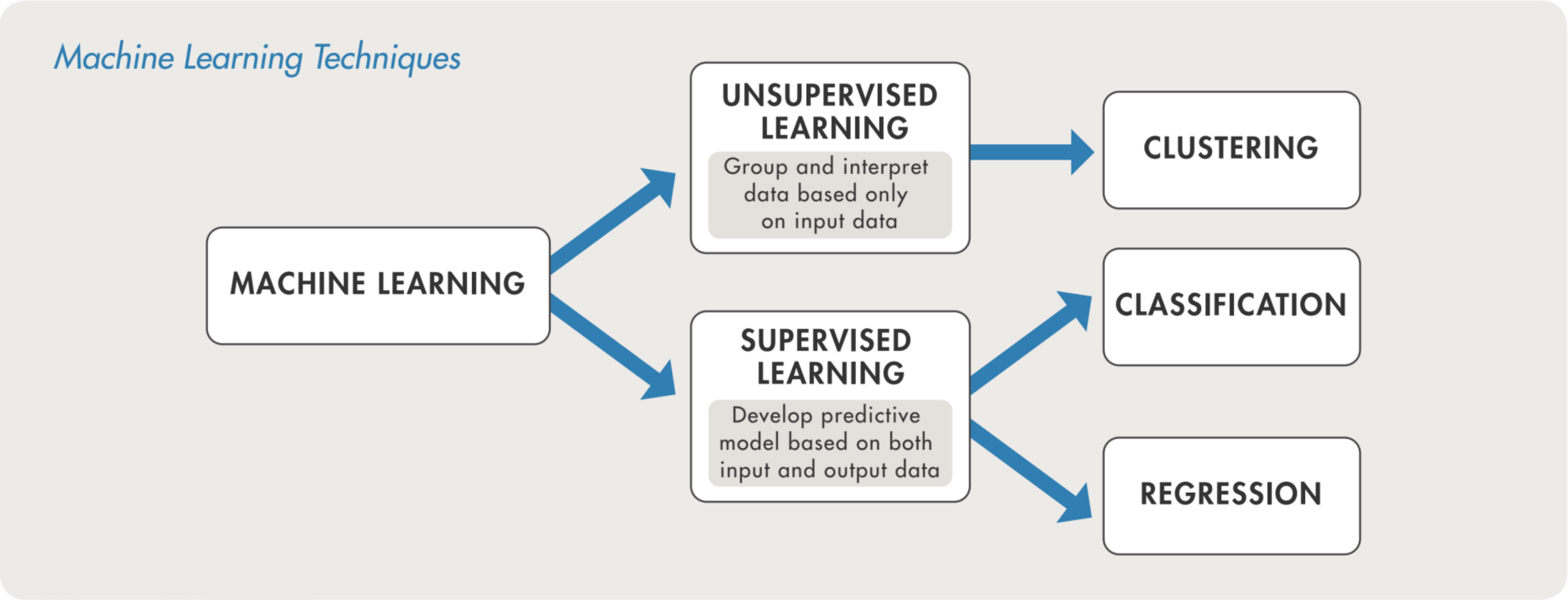


TRAINING PHASE

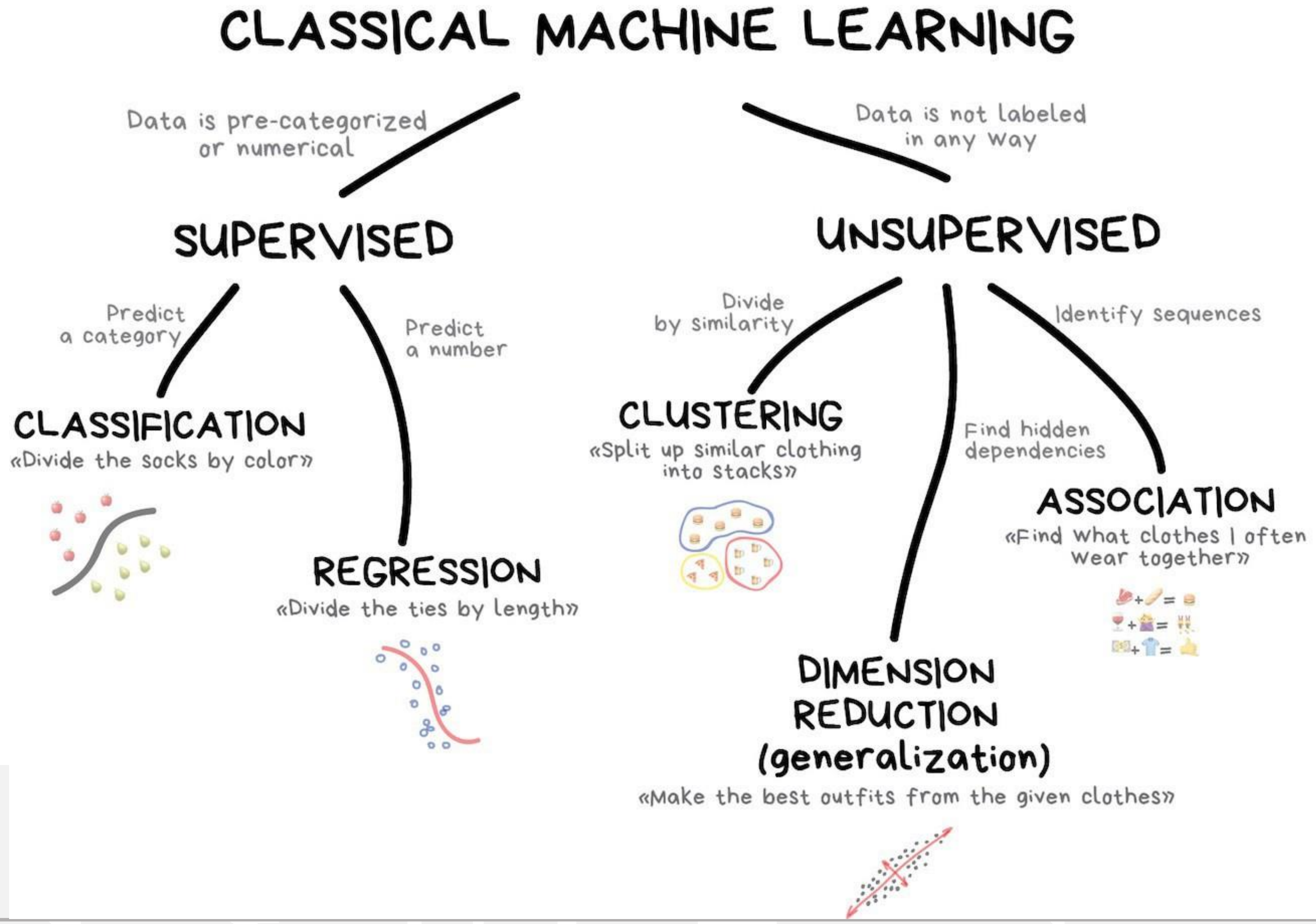
TESTING (INFERENCE) PHASE

MACHINE LEARNING TECHNIQUES

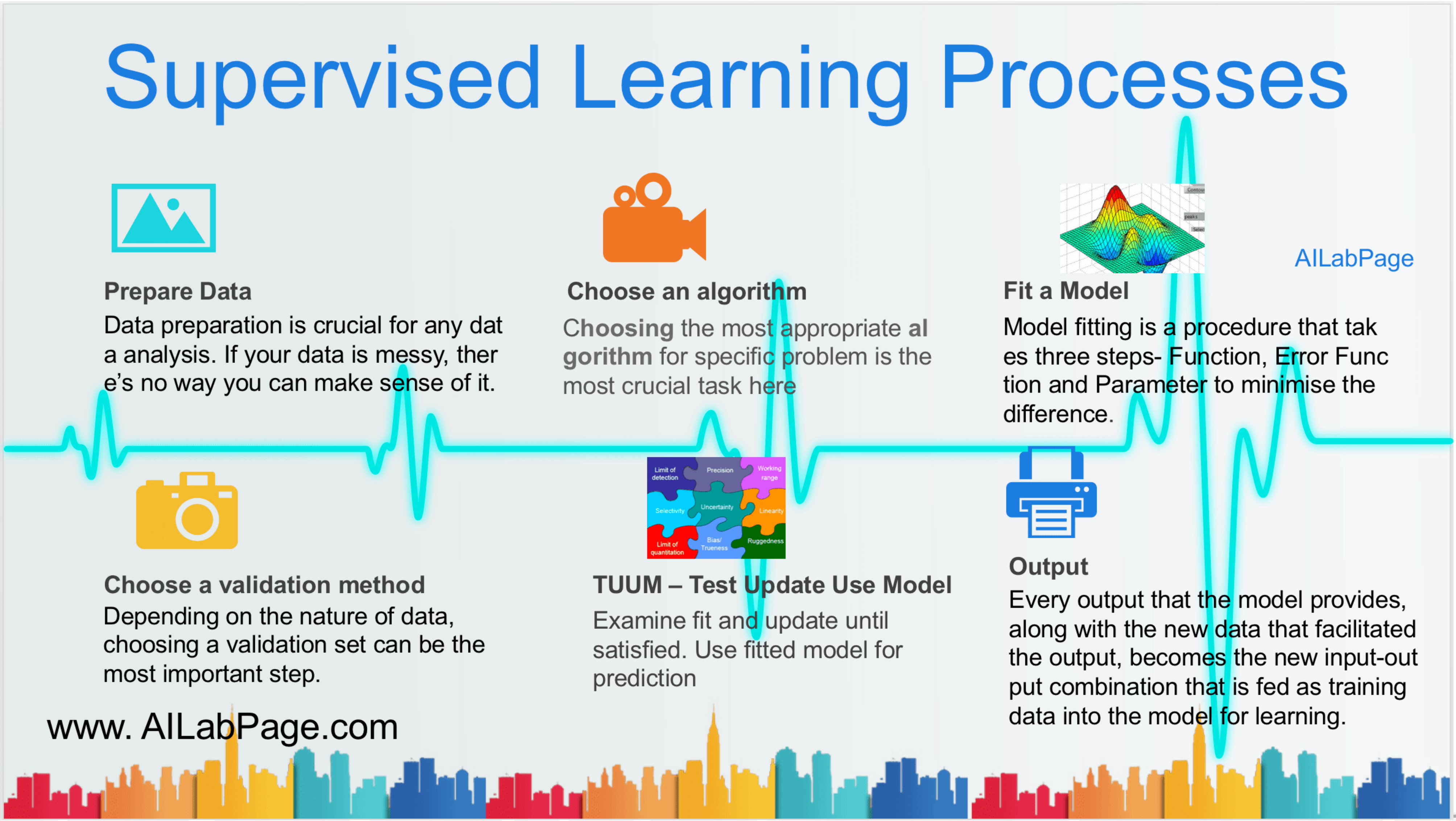
- **Supervised learning:** takes a known set of **input data** and **known responses to the data (output)** and builds a model that makes predictions based on evidence in the presence of uncertainty
- **Unsupervised learning:** finds hidden patterns or intrinsic structures in data. It is used to draw inferences from datasets consisting of input data **without labeled responses.**
- **Semi-supervised learning:** big data**, part of them is labeled**



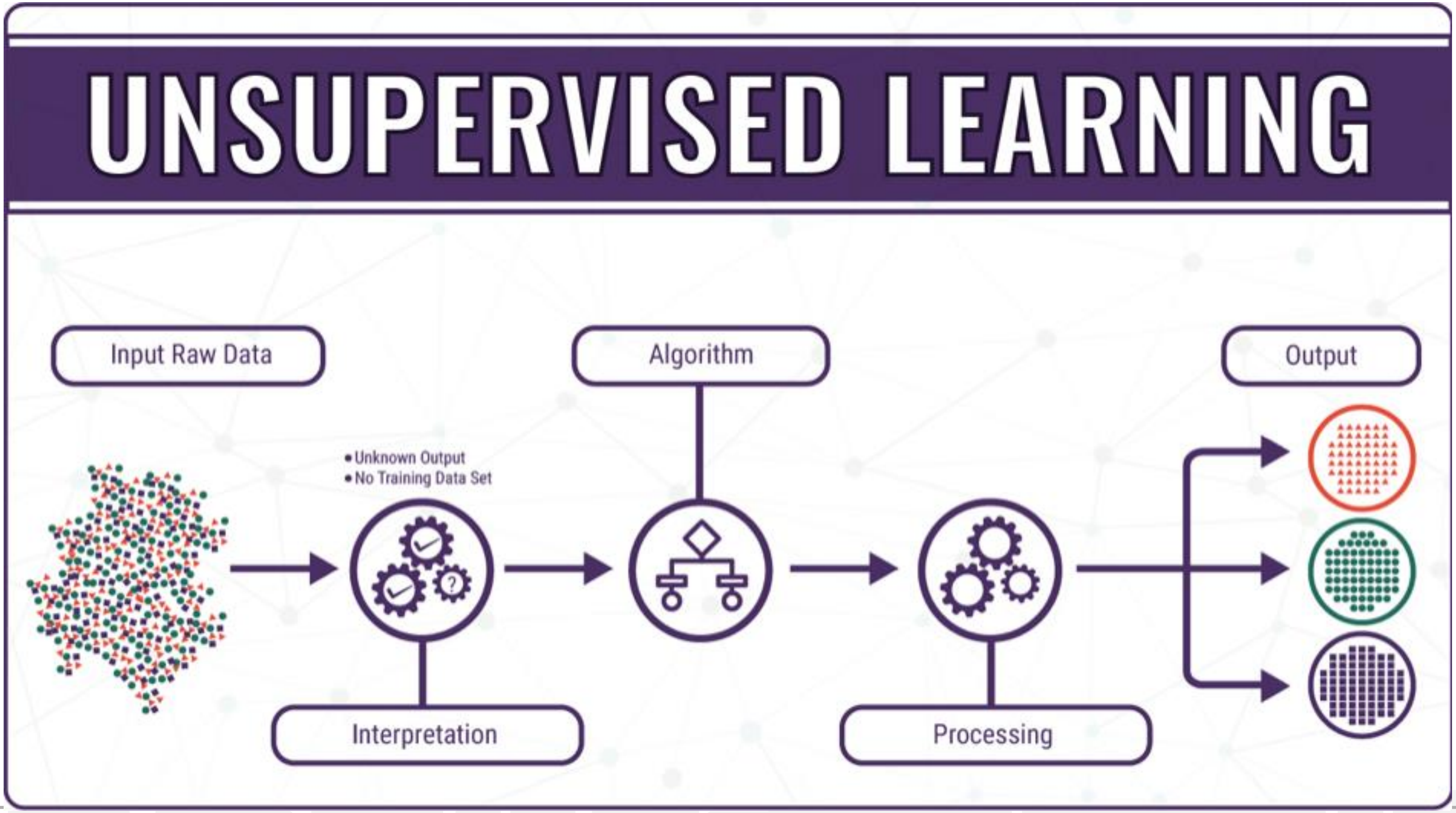
MACHINE LEARNING TECHNIQUES TYPES



SUPERVISED LEARNING PROCESS

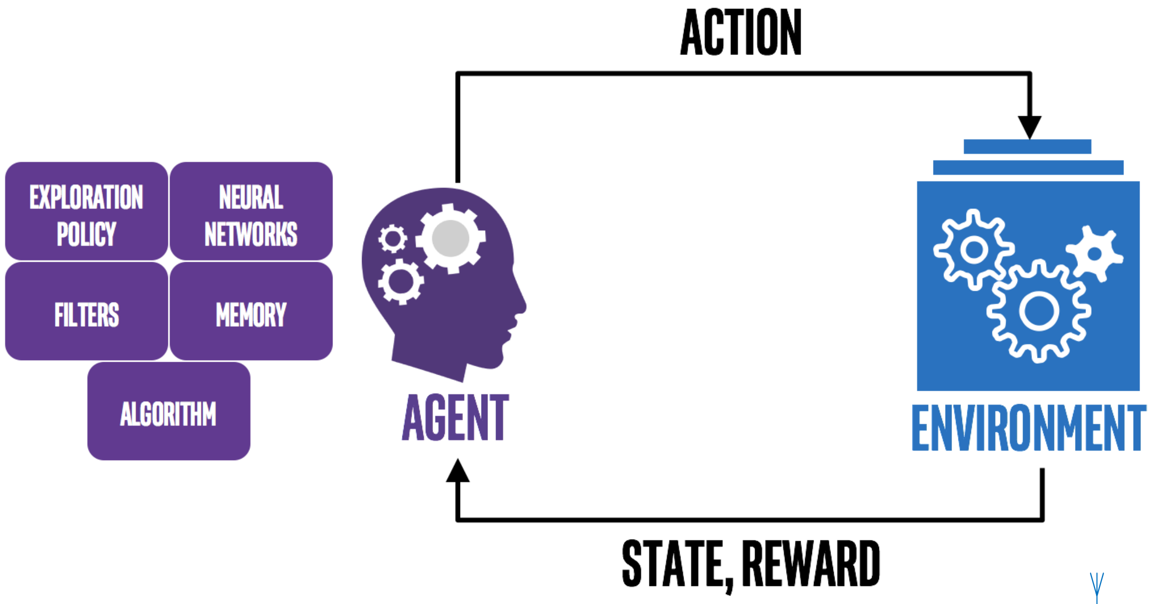


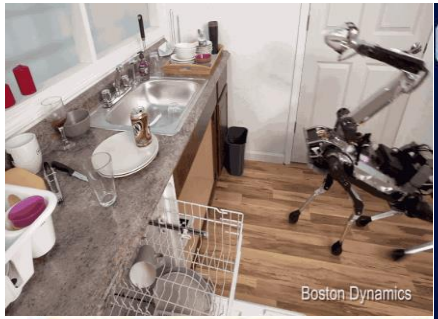
UNSUPERVISED LEARNING PROCESS

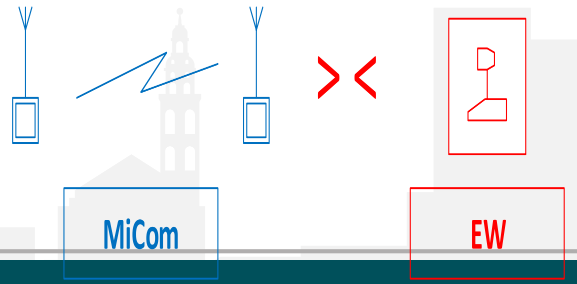


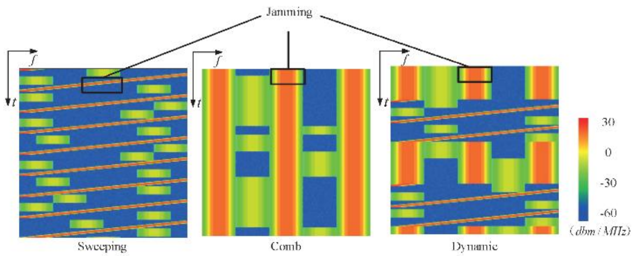
MACHINE LEARNING TECHNIQUES TYPES

- **Reinforcement learning:** different type, but still very important; algorithm for **decision making**





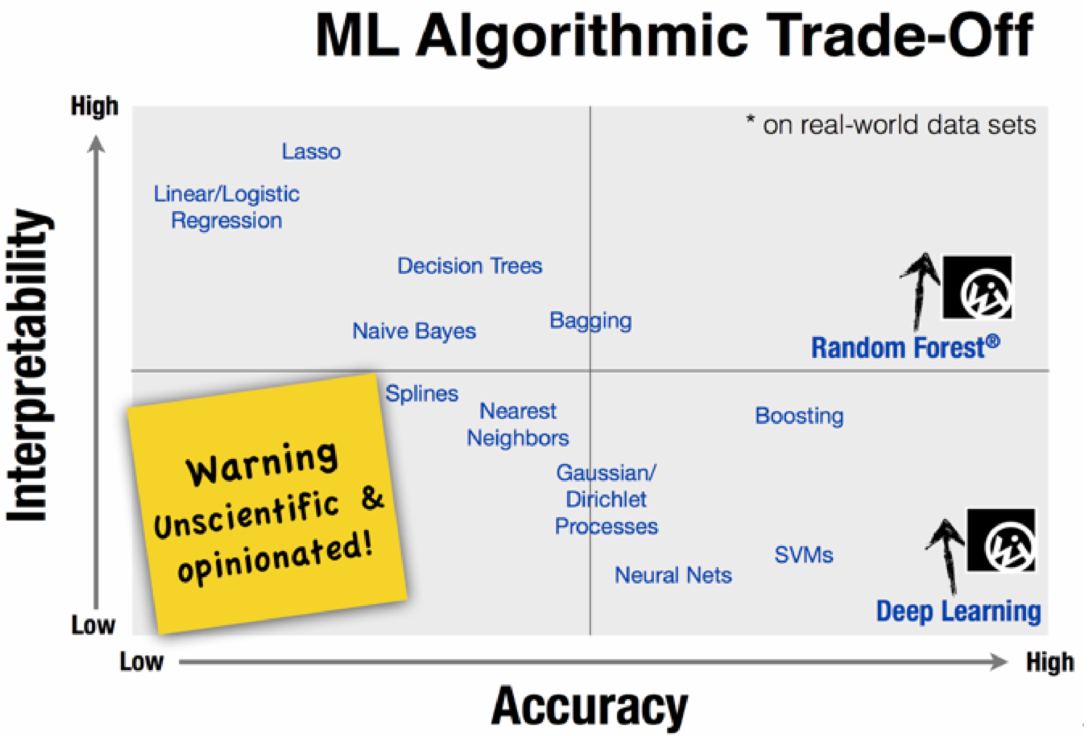




(a) Illustration of sweeping, comb and dynamic jamming modes.

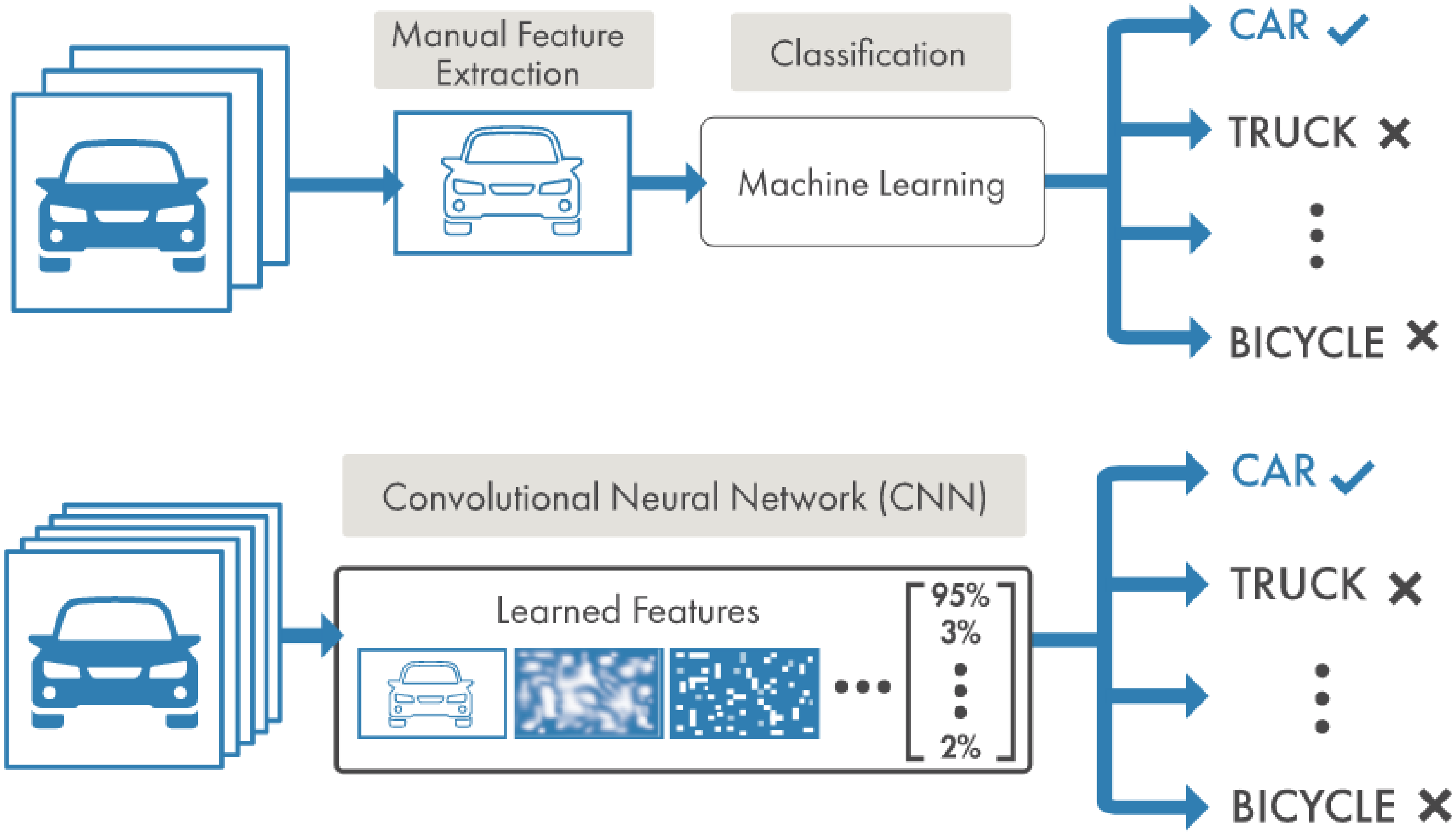
HOW DO YOU DECIDE WHICH ML ALGORITHM TO USE?

- **No best method** or **one size fits all**
- Finding the right algorithm is partly just **trial and error** even highly experienced data scientists can't tell whethe an algorithm will work without trying it out.
- Depends on the size and type of data; the insights want t get from the data
- Consider using ML: complex task + a large amount of data and lots of variables, but no existing formula or equation



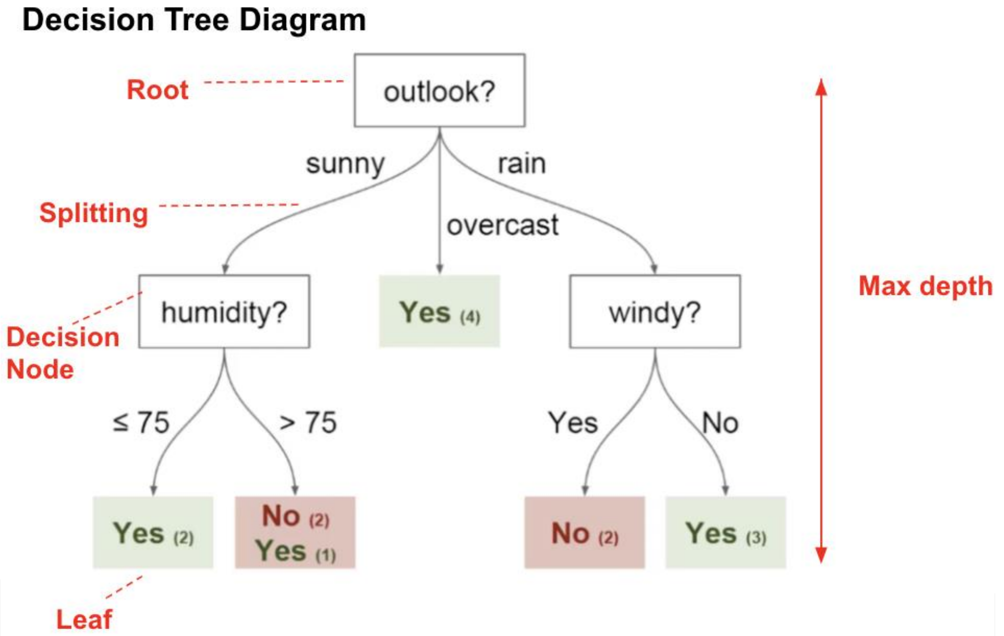
DEEP LEARNING & NEURAL NETWORK

TRADITIONAL MACHINE LEARNING vs DEEP LEARNING



DEEP LEARNING IS NOT IF…THEN…ELSE…

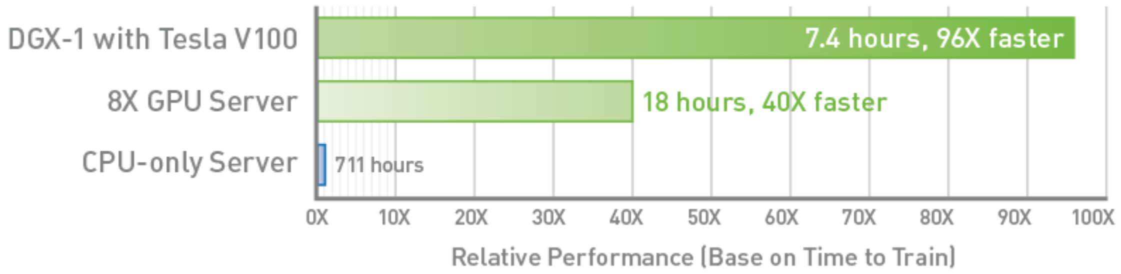
- If then Else is just the simplest form of AI (rule-based engine)
- But, how about for a very complex problem? -> really really hard to write all those complicated rules; hard to update it when a new issue arises as well



WHY IS DEEP LEARNING SUDDENLY IN EVERYWHERE?

- **GPU – The AI Engine Computational**
- **Huge Datasets (labelled) are available in everywhere in every areas**: 2,5 triệu tỷ byte/ngày; 90% lượng dữ liệu tạo ra trong 2 năm qua (số liệu 2017 theo Forbes)
- **DL Opensource platform/framework/tools**
- **Others:**
  - New activation function (Relu)
  - New regularization techniques
  - New optimization techniques

NVIDIA DGX-1 Delivers 96X Faster Training

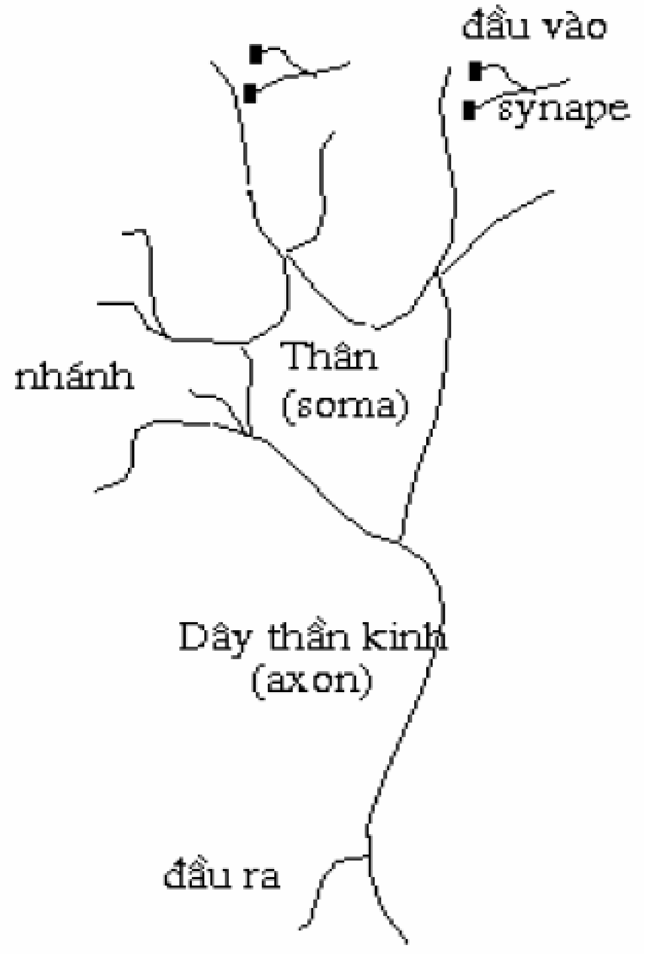


Workload: ResNet50, 90 epochs to solution | CPU Server: Dual Xeon E5-2699 v4, 2.6GHz

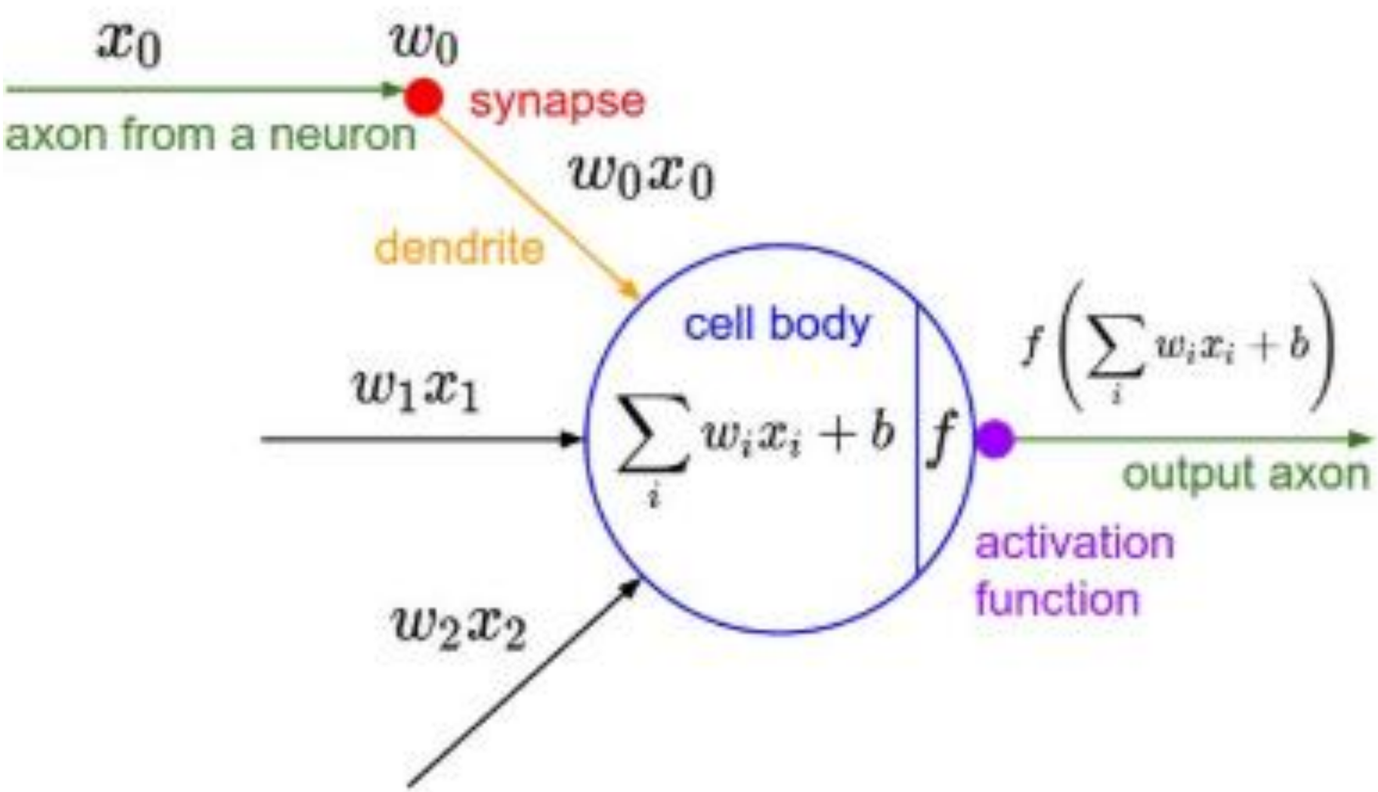
ImageNet:
~5M labeled images
~22k categories

Caffe
Microsoft CNTK
DEEPLEARNING4J
dmlc mxnet
TensorFlow
theano
torch

BIOLOGICAL vs ARTIFICIAL NEURON

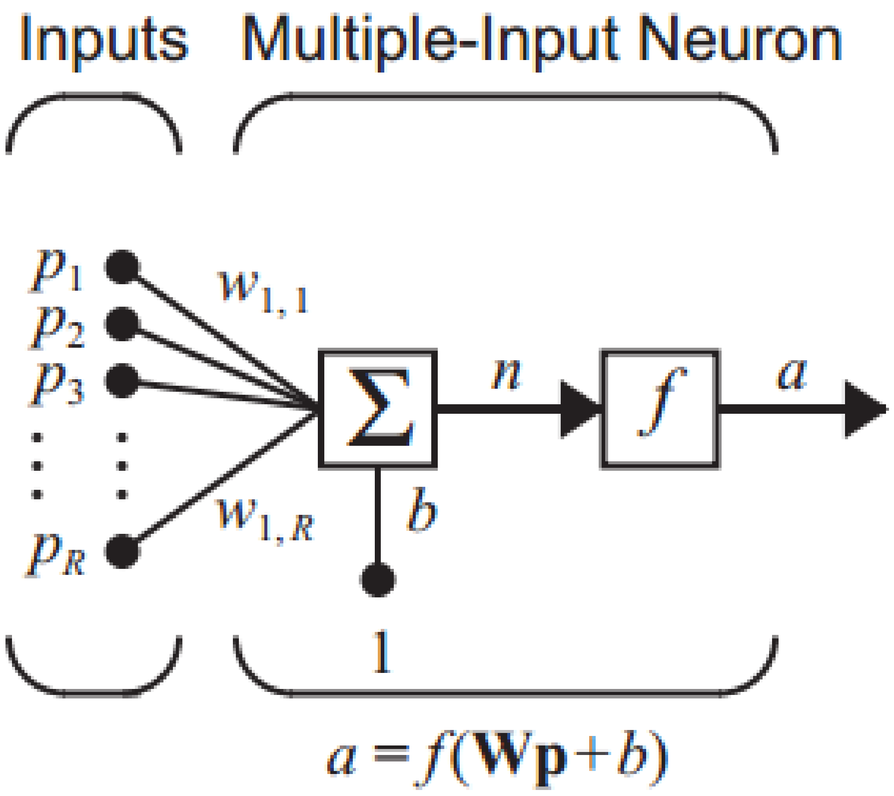


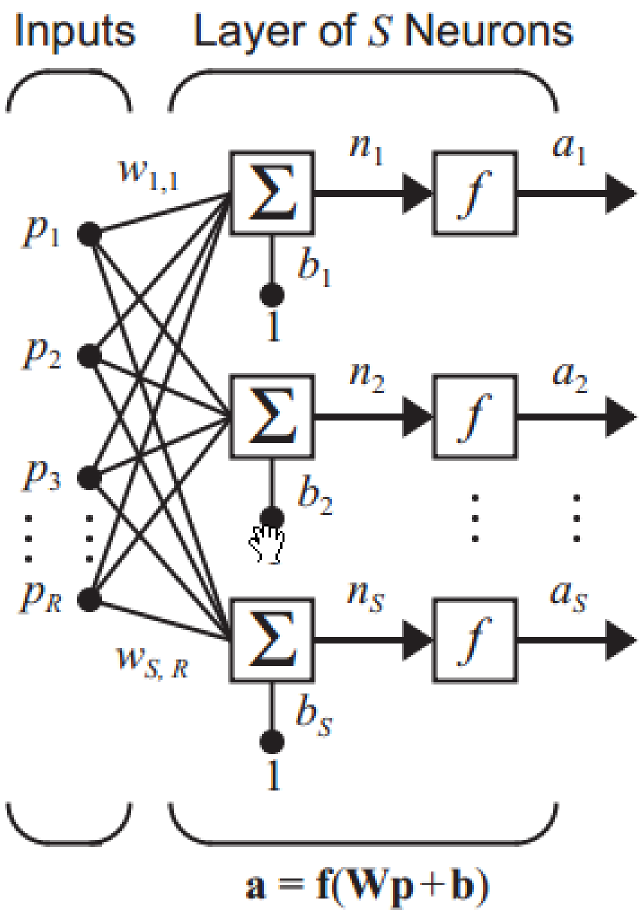
a,



Human brains consist of approximately 10^{11} neurons, each has app. 10^4 connections

ARTIFICIAL NEURON NETWORK (ANN)





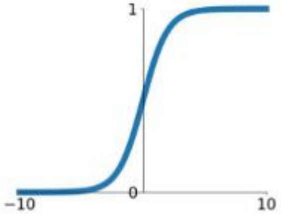
Neuron

Layer

POPULAR ACTIVATION FUNCTIONS IN NEURAL NETWORK

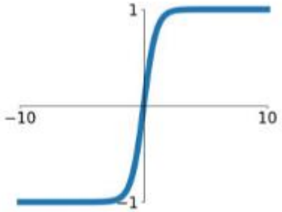
Sigmoid

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



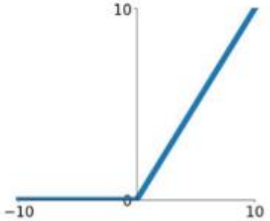
tanh

$\tanh(x)$



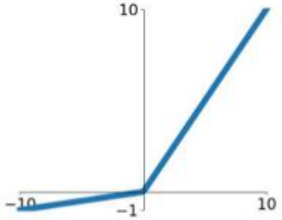
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

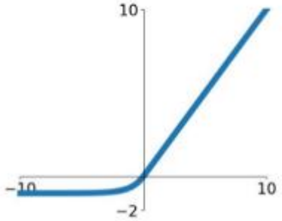


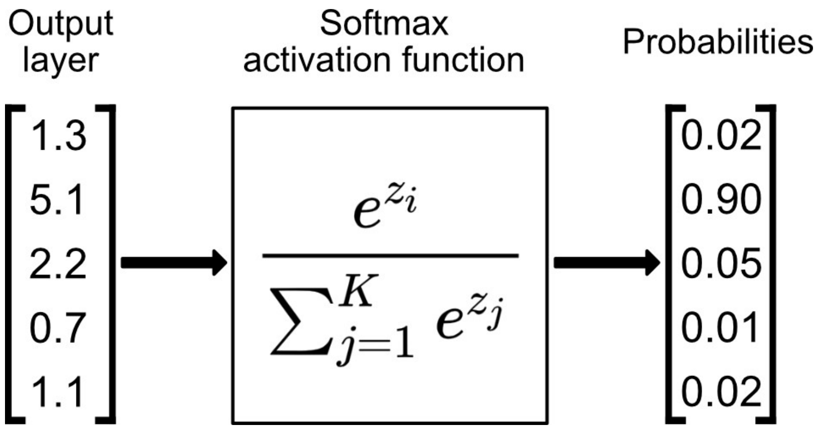
Maxout

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

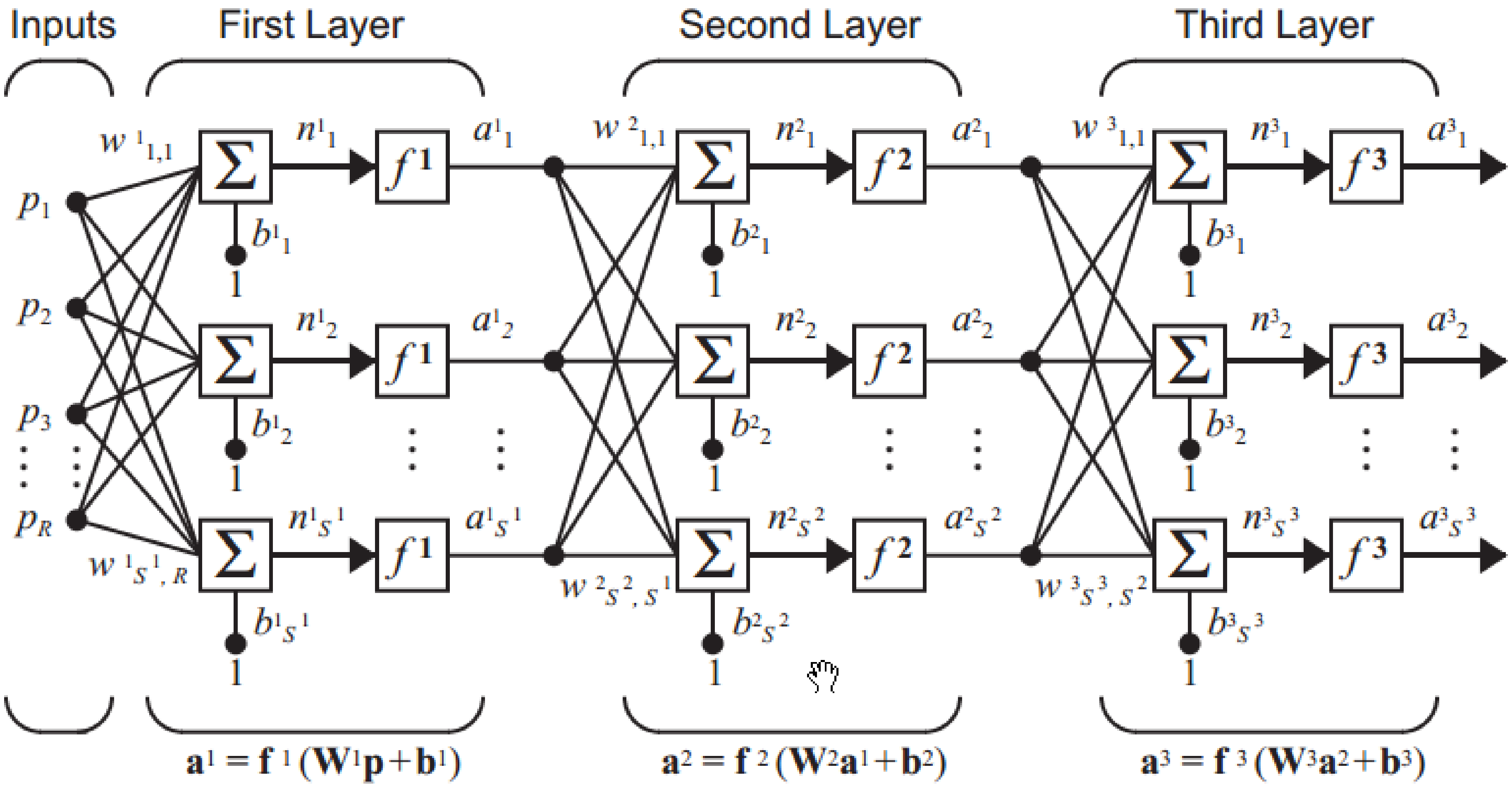
ELU

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





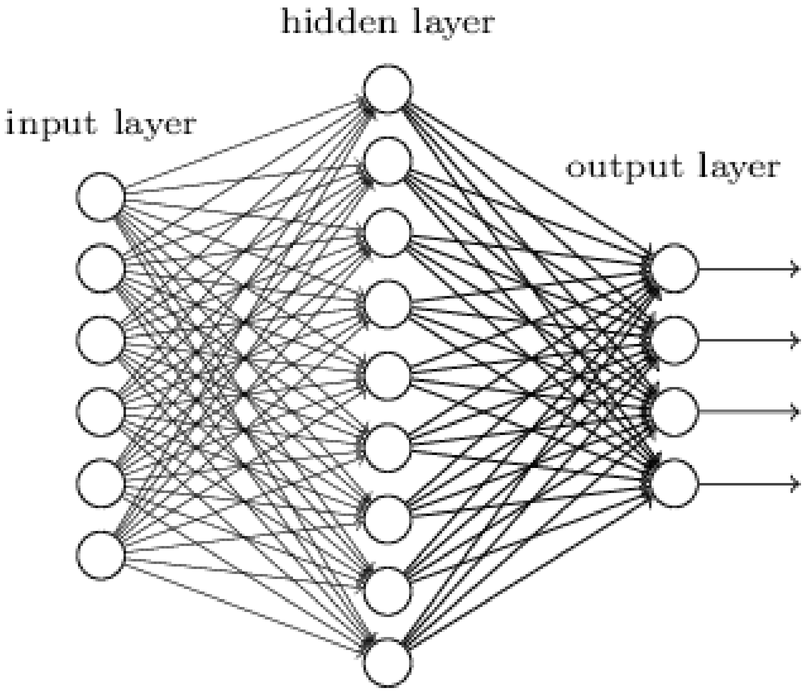
MULTI-LAYER NEURAL NETWORK



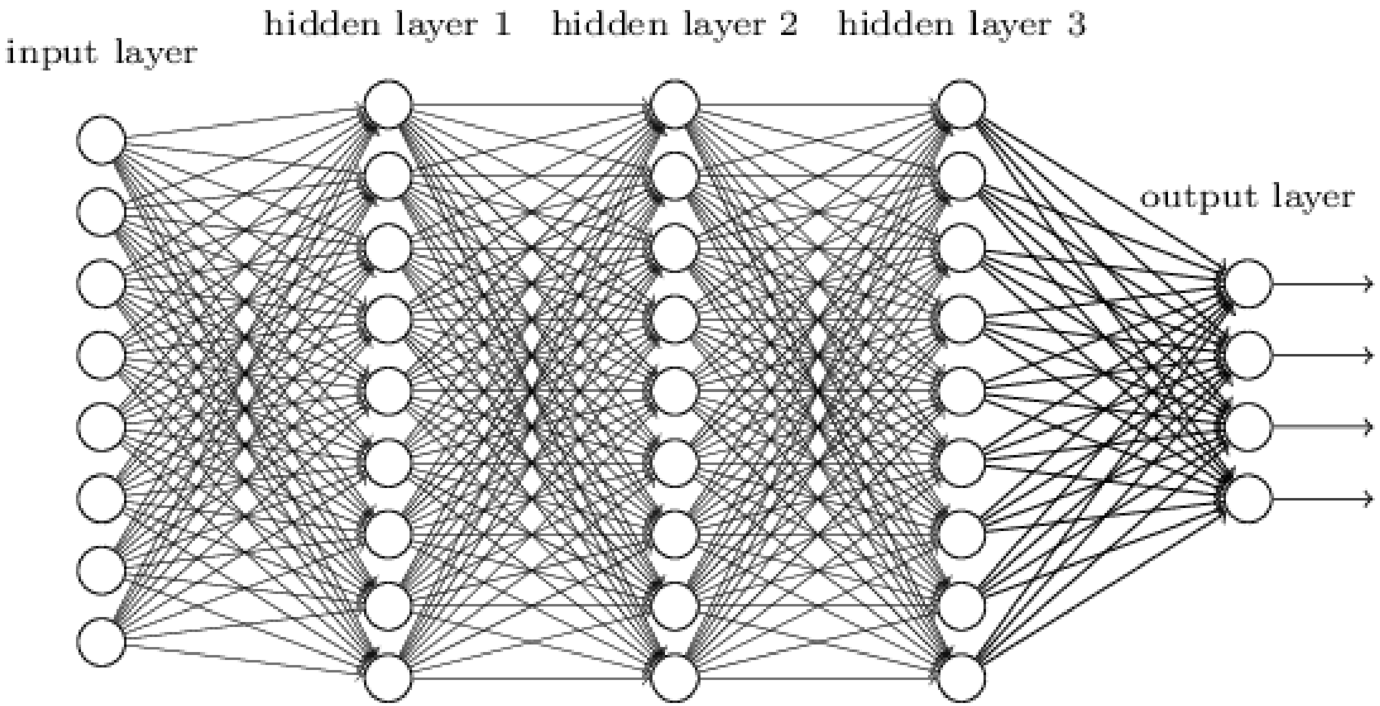
$$\mathbf{a}^3 = \mathbf{f}^3(\mathbf{W}^3\mathbf{f}^2(\mathbf{W}^2\mathbf{f}^1(\mathbf{W}^1\mathbf{p}+\mathbf{b}^1)+\mathbf{b}^2)+\mathbf{b}^3)$$

DEEP vs "NON-DEEP" (SHALLOW) LEARNING

"Non-deep" feedforward
neural network



Deep neural network



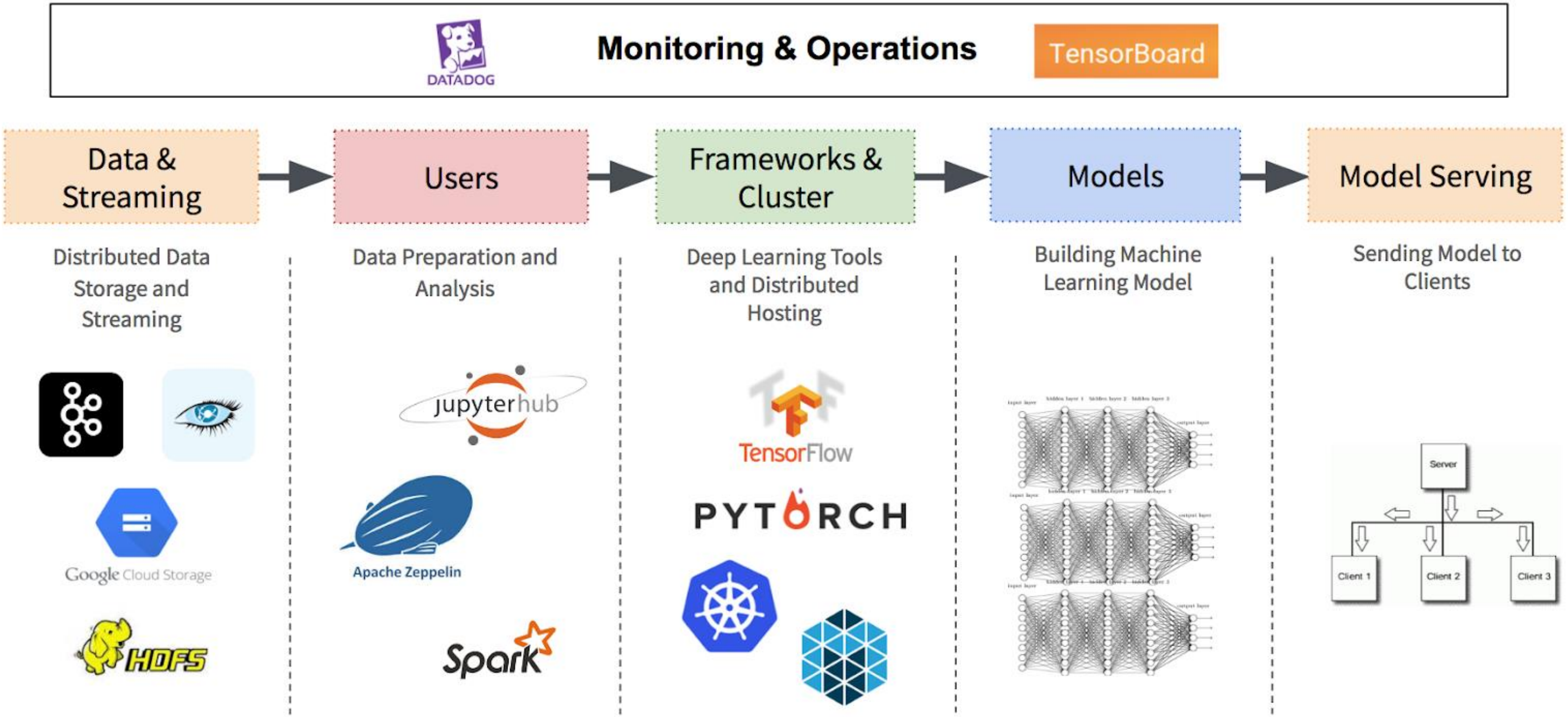
DEEP LEARNING PROCESS FOR A PROBLEM

- Thu thập dữ liệu, áp dụng các kỹ thuật xử lý dữ liệu trước huấn luyện (loại bỏ dữ liệu bất thường, chuẩn hóa, làm giàu thêm dữ liệu…)
- Xây dựng mô hình kiến trúc mạng neural
- Xây dựng loss function
- Thực hiện huấn luyện mô hình với tập dữ liệu đầu vào; tối ưu các tham số mô hình (hyperparameter tunning), áp dụng các kỹ thuật tránh overfitting

Mục tiêu: tìm được bộ weights **tối thiểu hóa loss function** & **good-fitting**

DEEP LEARNING PIPELINE

Deep Learning Pipeline



TRAINING PROCESS

Training set:

$$(p_1, t_1)$$
$$(p_2, t_2)$$
$$\vdots$$
$$(p_N, t_N)$$

Performance index:
Mean Squared Errors

$$F(\mathbf{w}) = E[\mathbf{e}^2] = E[(\mathbf{t} - \mathbf{a})^2]$$

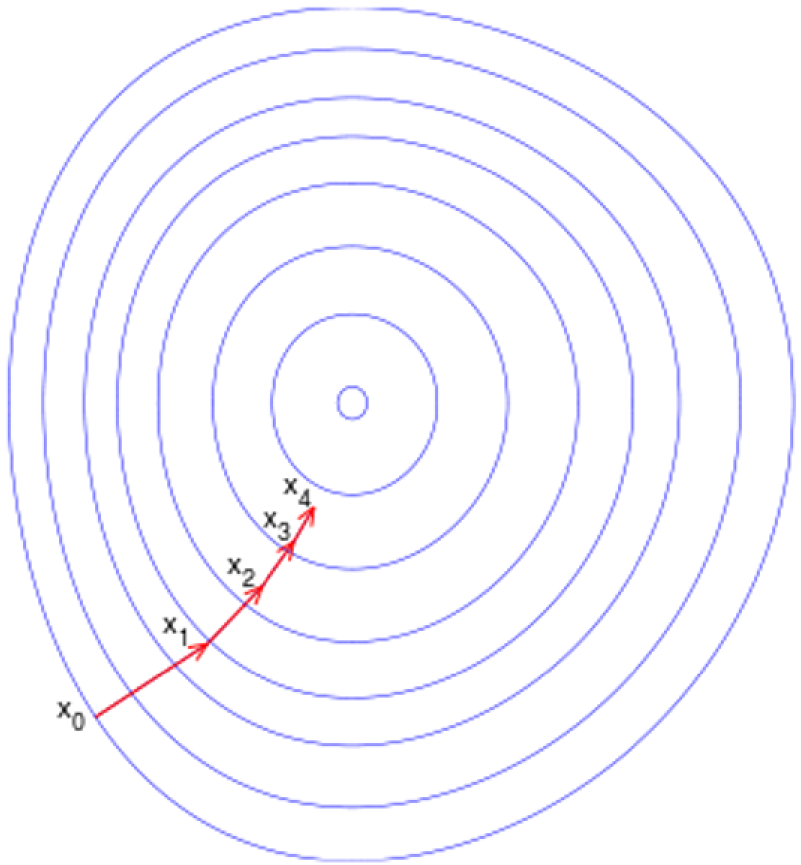
Need to find a set of weights $\mathbf{w}$ that minimizes $F(\mathbf{w})$

Weight update:

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \alpha \frac{\partial F}{\partial \mathbf{w}}$$

Use backpropagation algorithm

to calculate the $\dfrac{\partial F}{\partial \mathbf{w}}$



MỘT SỐ THUẬT TOÁN, KHÁI NIỆM SỬ DỤNG TRONG DEEP LEARNING

THUẬT TOÁN GRADIENT DESCENT

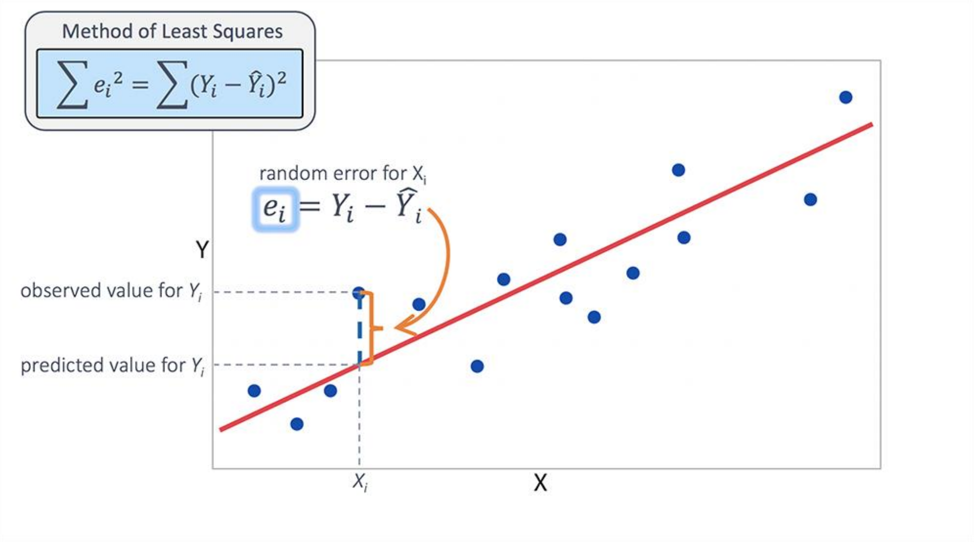
- Bài toán Linear Regression đơn giản nhất: Ước lượng giá nhà cho 1 căn nhà rộng x_1 (m^2), có x_2 phòng ngủ và cách trung tâm thành phố x_3 (km)?

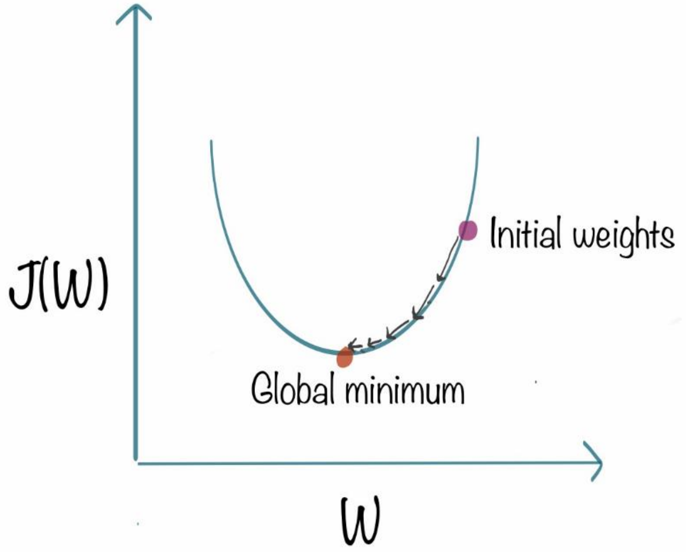
$$y \approx f(x) = \hat{y} = w_1 x_1 + w_2 x_2 + w_3 x_3 + w_0$$

- Minimize loss function (MSE) (*L2*-Norm): $\mathrm{L}(w)$

$= \frac{1}{2N}\sum_{i=1}^{N}(y_i - \bar{x}_i w)^2$; với *N* là số mẫu trong tập training

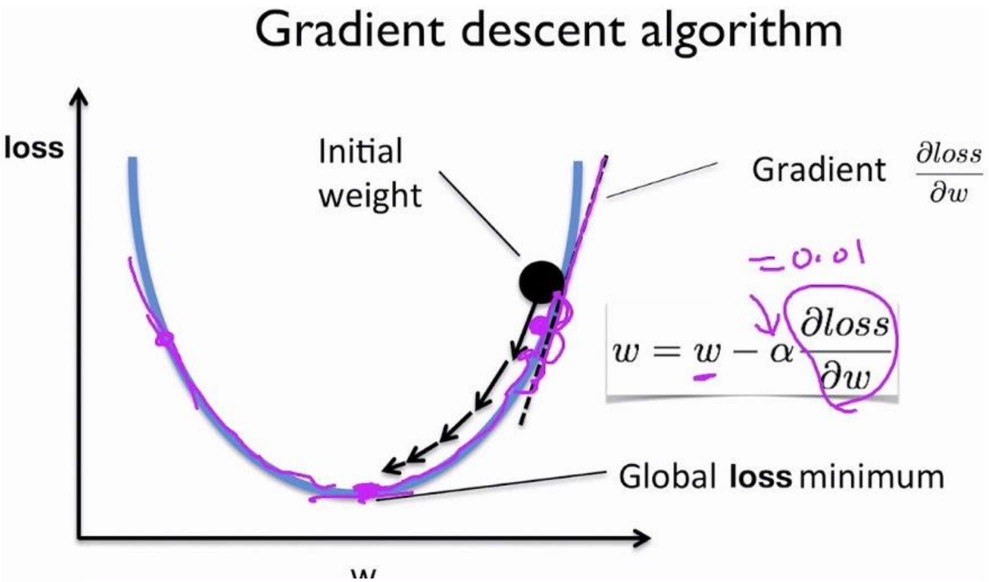
- Bài toán trở thành tìm điểm làm nghiệm phương trình đạo hàm = 0
- Trong thực tế, sử dụng ***phép toán lặp*** để tiến dần đến điểm cần tìm (***đạo hàm bằng 0***) -> đó là ý tưởng cơ bản của thuật toán ***Gradient Descent***.



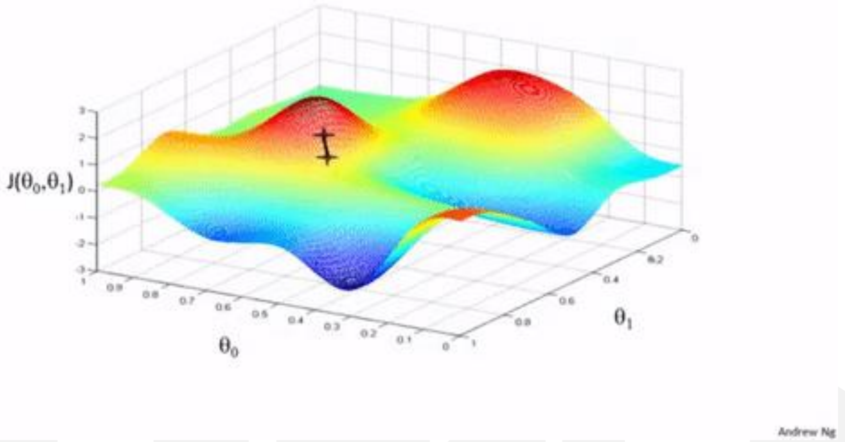


- Giả sử w_t là điểm tìm được sau lần lặp t; cần tìm một thuật toán để đưa w_t về càng gần $w*$.
  - $f'(x_t) > 0$ thì x_t đang ở bên phải so với điểm tối ưu và ngược lại -> cần di chuyển x_t về phía ngược lại dấu của đạo hàm một lượng Δ: $x_{t+1} = x_t + \Delta$. Đó là ý nghĩa của Gradient Descent (*đi ngược của đạo hàm*).
  - Quy tắc cập nhật của thuật toán Gradient Descent là:

$\boldsymbol{x_{t+1} = x_t - \eta * f'(x_t)}$, với $\eta > 0$ gọi là tốc độ học (learning rate).



- Đối với trường hợp hàm nhiều biến, việc áp dụng thuật toán Gradient Descent tương tự như đối với trường hợp hàm một biến bằng việc áp dụng nguyên tắc tính đạo hàm từng phần (partial derivative)

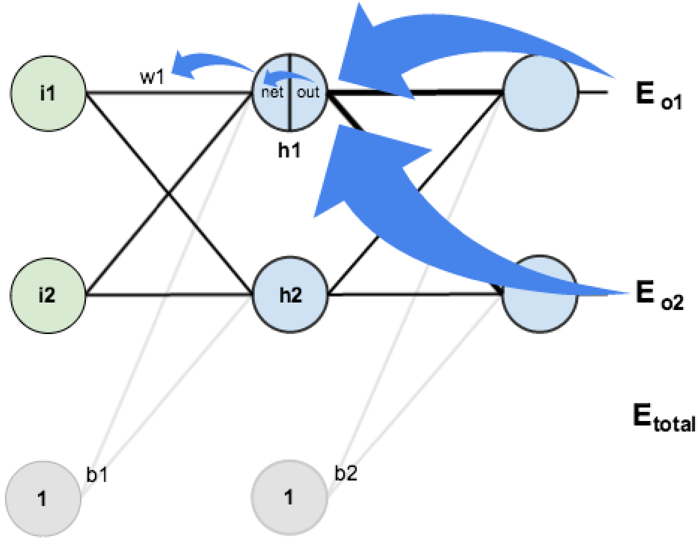


THUẬT TOÁN BACKPROPAGATION

- **Backpropagation (Lan truyền ngược)** là một thuật toán dùng để tính toán đạo hàm thành phần cho weights của từng layer kể từ layer cuối cùng (kỹ thuật nhanh chóng tính được đạo hàm); là 1 dạng biểu diễn của *chain-rule*: quy tắc tính đạo hàm của 1 hàm hợp
- Là **giải thuật cốt lõi** giúp cho các mô hình học sâu có thể dễ dàng thực thi tính toán được, nhanh hơn rất nhiều so với các phương pháp truyền thống

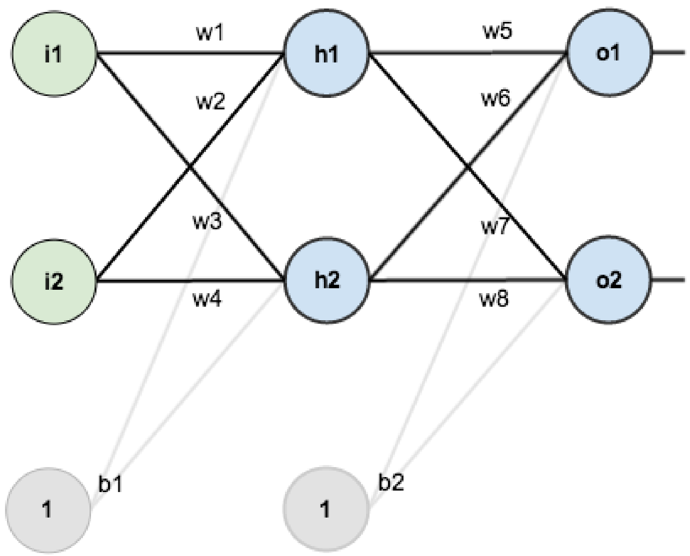
$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

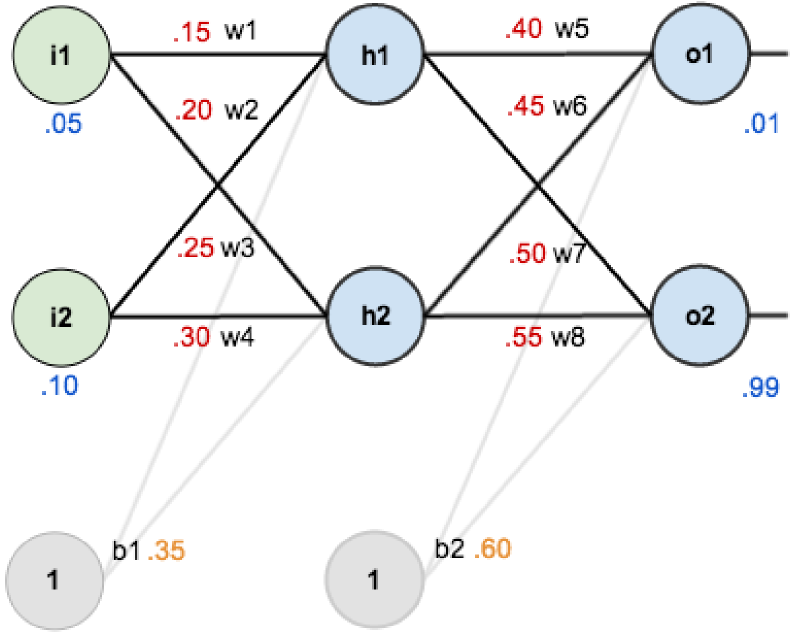
$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}}$$



$E_{total} = E_{o1} + E_{o2}$

➢ **Step 1**: **Forward Pass**, assume activation function is *sigmoid*, calculate total error





$$net_{h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$

$$net_{h1} = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$$

$$out_{h1} = \frac{1}{1+e^{-net_{h1}}} = \frac{1}{1+e^{-0.3775}} = 0.593269992$$

$$out_{h2} = 0.596884378$$

$$net_{o1} = w_5 * out_{h1} + w_6 * out_{h2} + b_2 * 1$$

$$net_{o1} = 0.4 * 0.593269992 + 0.45 * 0.596884378 + 0.6 * 1 = 1.105905967$$

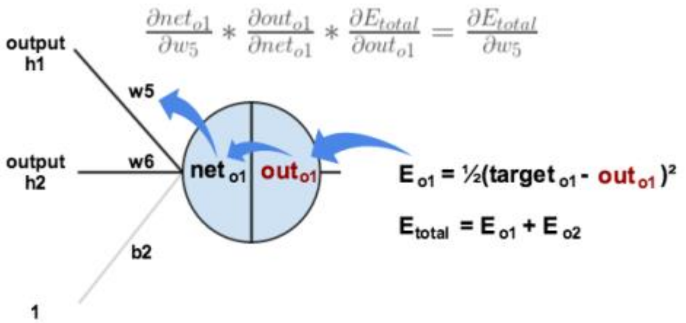
$$out_{o1} = \frac{1}{1+e^{-net_{o1}}} = \frac{1}{1+e^{-1.105905967}} = 0.75136507$$

$$out_{o2} = 0.772928465$$

$$E_{o1} = \frac{1}{2}(target_{o1} - out_{o1})^2 = \frac{1}{2}(0.01 - 0.75136507)^2 = 0.274811083$$

$$E_{total} = E_{o1} + E_{o2} = 0.274811083 + 0.023560026 = 0.298371109$$

- **Step 2**: **Backward propagation**: Goal is to **update each of the weights in the network** so that they cause the actual output to be closer the target output, thereby minimizing the error for each output neuron and the network
  - *2.1. Output layer:* Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

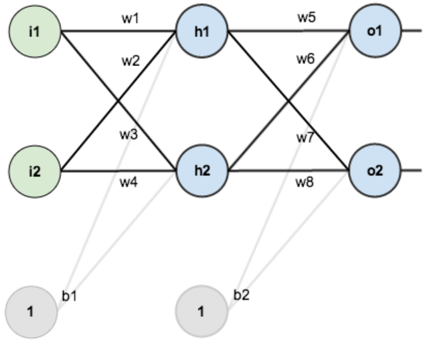


$$E_{total} = \frac{1}{2}(target_{o1} - out_{o1})^2 + \frac{1}{2}(target_{o2} - out_{o2})^2$$

$$\frac{\partial E_{total}}{\partial out_{o1}} = 2 * \frac{1}{2}(target_{o1} - out_{o1})^{2-1} * -1 + 0$$

$$\frac{\partial E_{total}}{\partial out_{o1}} = -(target_{o1} - out_{o1}) = -(0.01 - 0.75136507) = 0.74136507$$

$$out_{o1} = \frac{1}{1+e^{-net_{o1}}}$$



$$\frac{\partial out_{o1}}{\partial net_{o1}} = out_{o1}(1 - out_{o1}) = 0.75136507(1 - 0.75136507) = 0.186815602$$

$$net_{o1} = w_5 * out_{h1} + w_6 * out_{h2} + b_2 * 1$$

$$\frac{\partial net_{o1}}{\partial w_5} = 1 * out_{h1} * w_5^{(1-1)} + 0 + 0 = out_{h1} = 0.593269992$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

$$\frac{\partial E_{total}}{\partial w_5} = 0.74136507 * 0.186815602 * 0.593269992 = 0.082167041$$

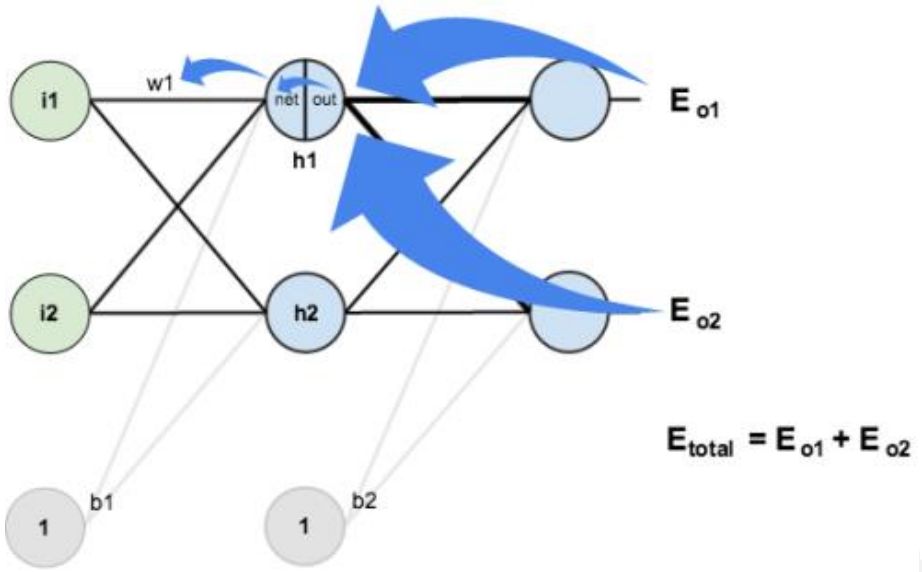
$$w_5^+ = w_5 - \eta * \frac{\partial E_{total}}{\partial w_5} = 0.4 - 0.5 * 0.082167041 = 0.35891648$$

- *2.2. Hidden layer:*

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\downarrow$$

$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}}$$



$$\frac{\partial E_{o1}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial out_{h1}}$$

$$\frac{\partial E_{o1}}{\partial net_{o1}} = \frac{\partial E_{o1}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} = 0.74136507 * 0.186815602 = 0.138498562$$

$$net_{o1} = w_5 * out_{h1} + w_6 * out_{h2} + b_2 * 1$$

$$\frac{\partial net_{o1}}{\partial out_{h1}} = w_5 = 0.40$$

$$\frac{\partial E_{o1}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial out_{h1}} = 0.138498562 * 0.40 = 0.055399425$$

$$\frac{\partial E_{o2}}{\partial out_{h1}} = -0.019049119$$

$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}} = 0.055399425 + -0.019049119 = 0.036350306$$

$$out_{h1} = \frac{1}{1+e^{-net_{h1}}}$$

$$\frac{\partial out_{h1}}{\partial net_{h1}} = out_{h1}(1 - out_{h1}) = 0.59326999(1 - 0.59326999) = 0.241300709$$

$$net_{h1} = w_1 * i_1 + w_3 * i_2 + b_1 * 1$$

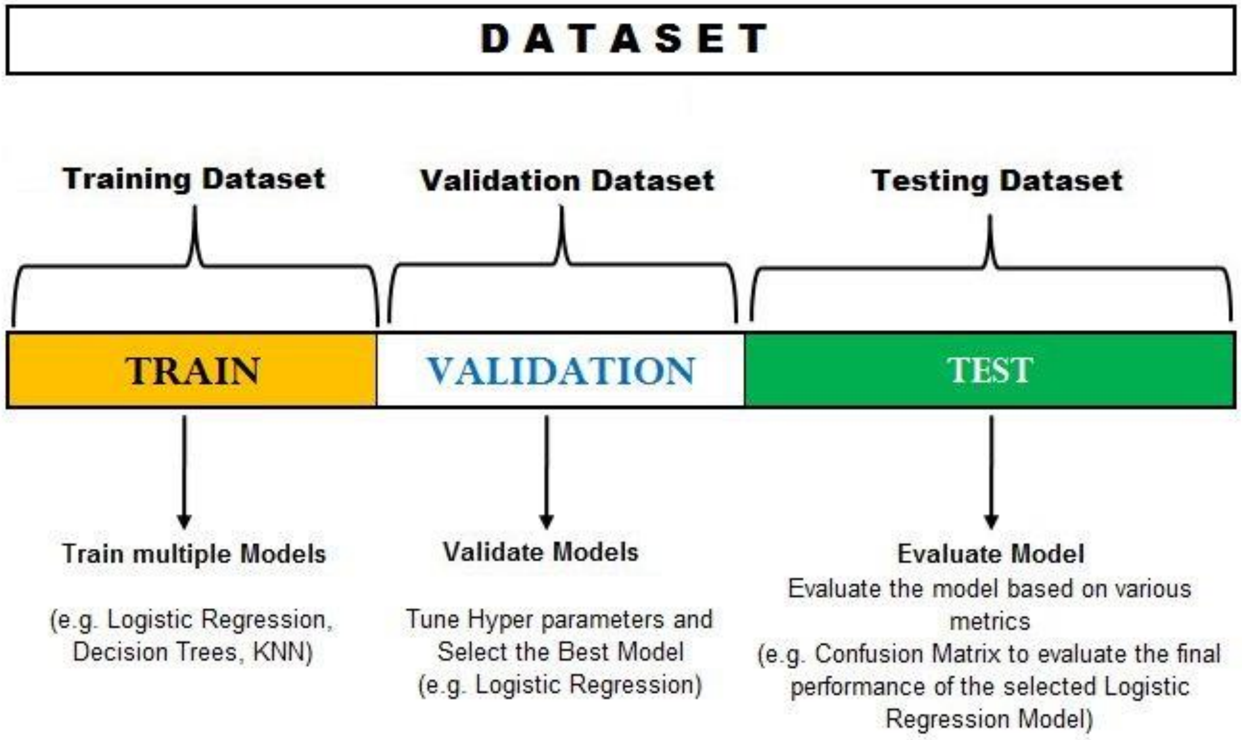
$$\frac{\partial net_{h1}}{\partial w_1} = i_1 = 0.05$$

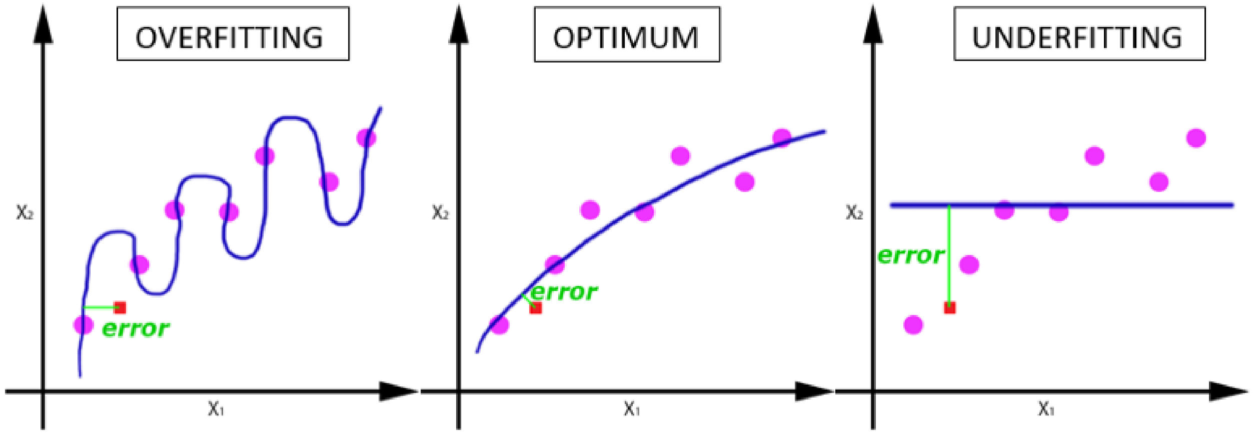
$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\frac{\partial E_{total}}{\partial w_1} = 0.036350306 * 0.241300709 * 0.05 = 0.000438568$$

$$w_1^+ = w_1 - \eta * \frac{\partial E_{total}}{\partial w_1} = 0.15 - 0.5 * 0.000438568 = 0.149780716$$

- Training set/Testing set/Validation set
- Overfitting / Underfitting





TRAINING TERMINOLOGIES

- **Các kỹ thuật chống overfitting:** Cross Validation (k-fold validation); Early stopping; Batch Normalization; Regularization; Dropout…

Train				Validation	Kết quả
Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	a1
Fold 1	Fold 2	Fold 3	Fold 5	Fold 4	a2
Fold 1	Fold 2	Fold 5	Fold 4	Fold 3	a3
Fold 1	Fold 5	Fold 3	Fold 4	Fold 2	a4
Fold 5	Fold 2	Fold 3	Fold 4	Fold 1	a5

Batch Normalization

Sunday, 24 March 2019 6:19 PM

Step-1 Calculate Batch Statistics

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$$ Batch mean

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i-\mu_B)^2$$ Batch Variance

Step2 Normalize Layer Inputs

$$\bar{x}_i = \frac{x_i-\mu_B}{\sqrt{\sigma_B^2+\epsilon}}$$

Step3 Scaling and Shifting

$$\boxed{y_i = \gamma \bar{x}_i + \beta}$$

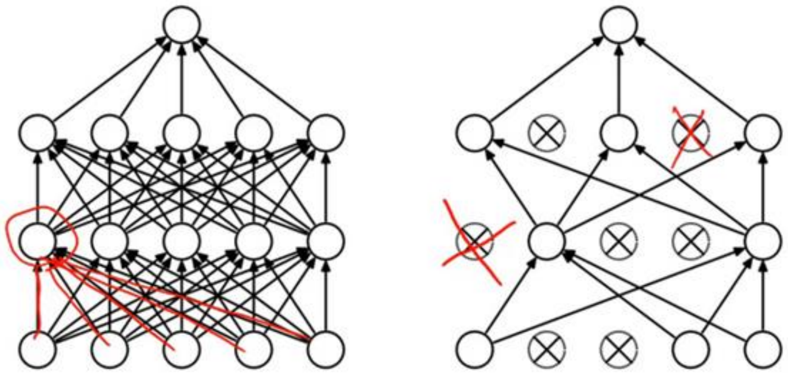
* γ and β are learnt during training.

L1 Regularization

$$\text{Cost} = \sum_{i=0}^{N}\Big(y_i - \sum_{j=0}^{M} x_{ij}W_j\Big)^2 + \lambda\sum_{j=0}^{M}|W_j|$$

L2 Regularization

$$\text{Cost} = \underbrace{\sum_{i=0}^{N}\Big(y_i - \sum_{j=0}^{M} x_{ij}W_j\Big)^2}_{\text{Loss function}} + \underbrace{\lambda\sum_{j=0}^{M} W_j^2}_{\text{Regularization Term}}$$



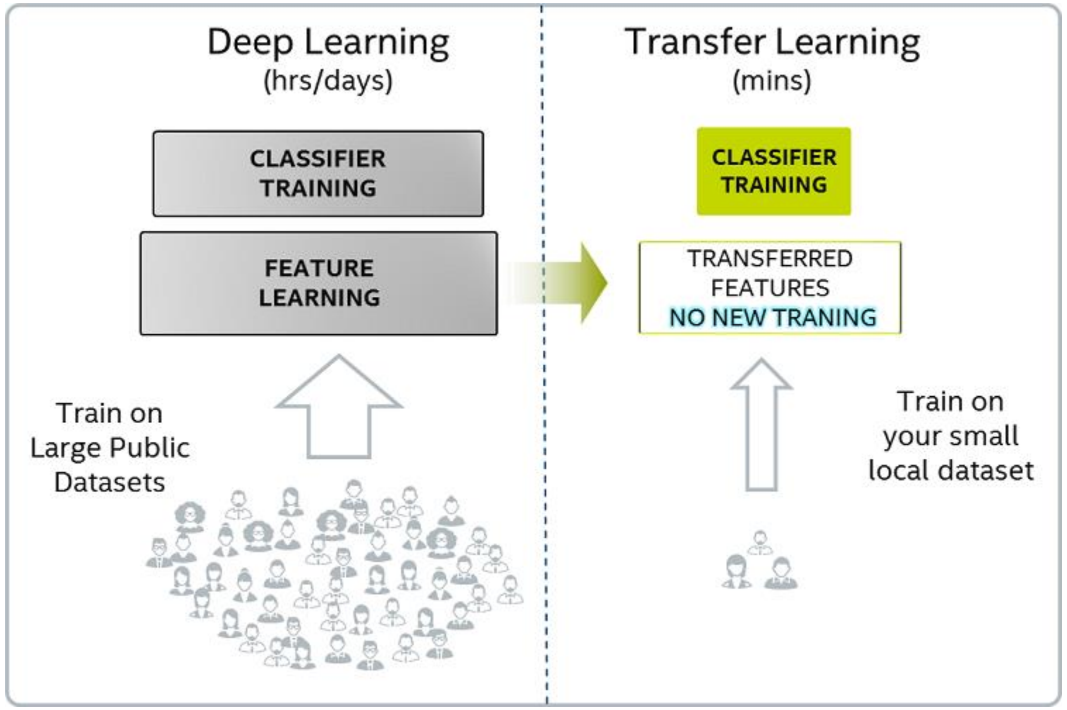
(a) Standard Neural Net (b) After applying dropout.

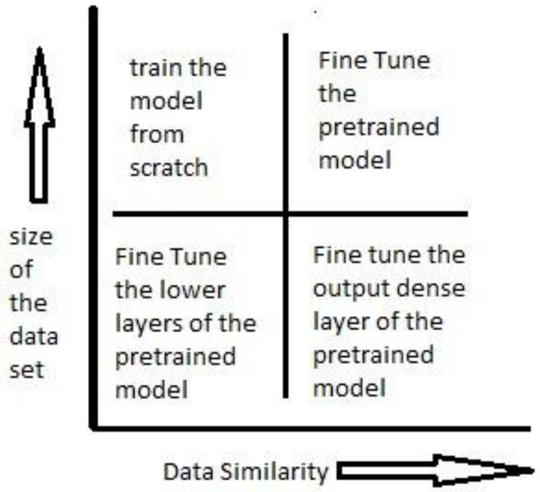
Epoch/Batch/Iteration

- Cần phải chia nhỏ tập dữ liệu thành các **batch** (size nhỏ hơn) và đưa vào train lần lượt nhiều lần để tránh overfitting (**Batch size**: số lượng mẫu trong 1 batch)
- **Epoch**: được tính khi đưa tất cả dữ liệu vào mạng neural network 1 lần (quá trình train gồm nhiều epoch)
- **Iterations** là số lượng batchs cần để hoàn thành 1 epoch

Ví dụ: cả tập dữ liệu gồm 20,000 mẫu, batch size: 500, vậy chúng ta cần 40 lần lặp (iteration) để hoàn thành 1 epoch.

TRANSFER LEARNING





Freeze or fine-tune?

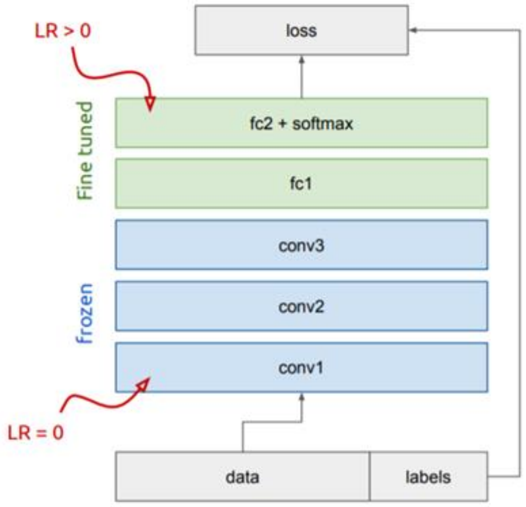
Bottom *n* layers can be frozen or fine tuned.

- **Frozen**: not updated during backprop
- **Fine-tuned**: updated during backprop

Which to do depends on target task:

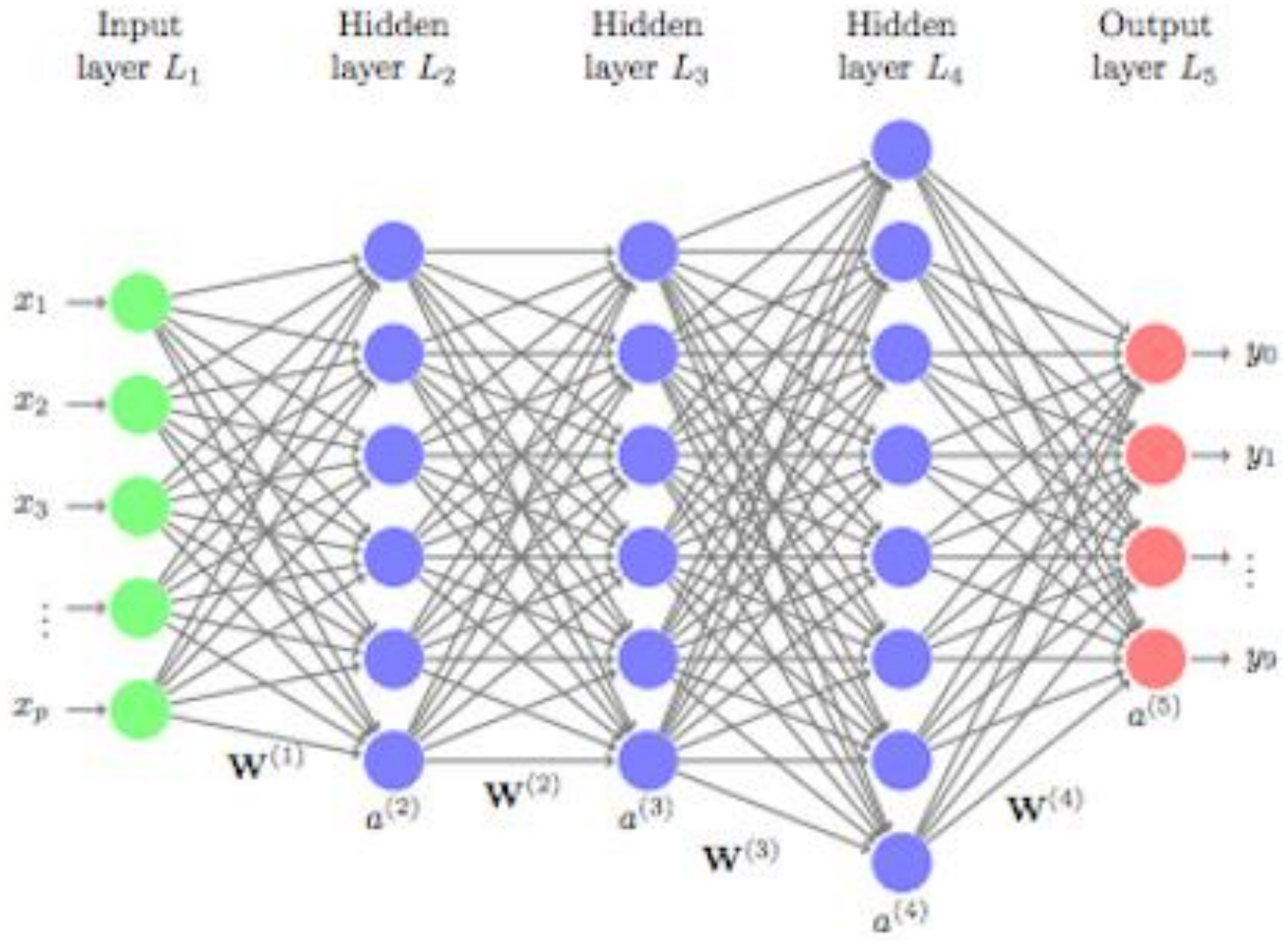
- **Freeze**: target task labels are scarce, and we want to avoid overfitting
- **Fine-tune**: target task labels are more plentiful

In general, we can set learning rates to be different for each layer to find a tradeoff between freezing and fine tuning

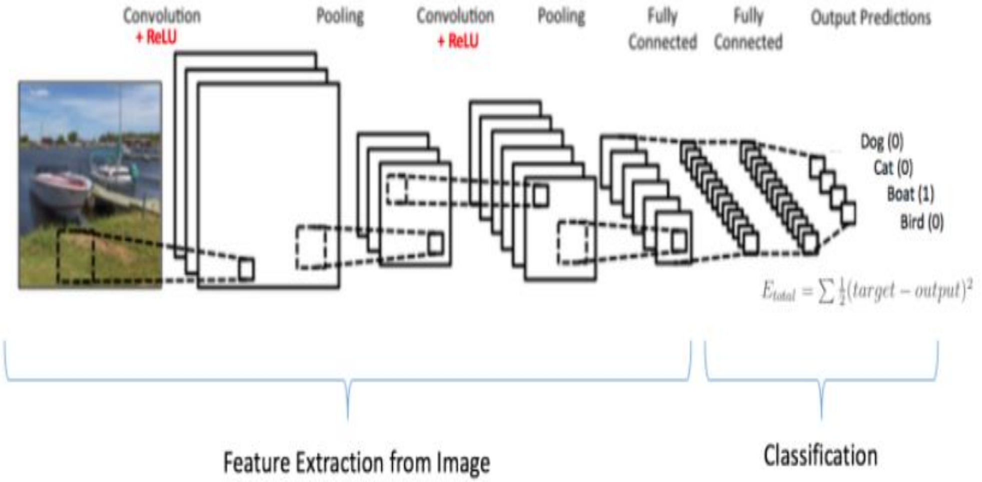


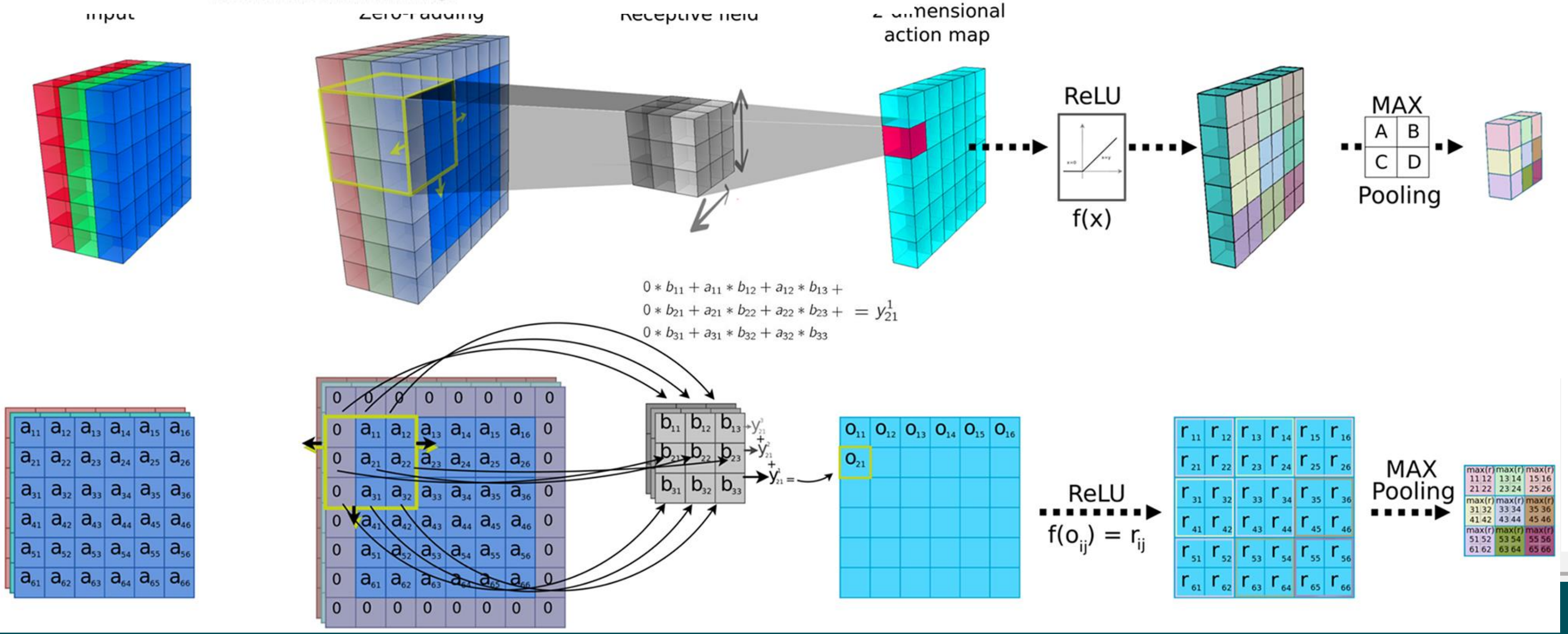
MỘT SỐ KIẾN TRÚC NEURAL NETWORK

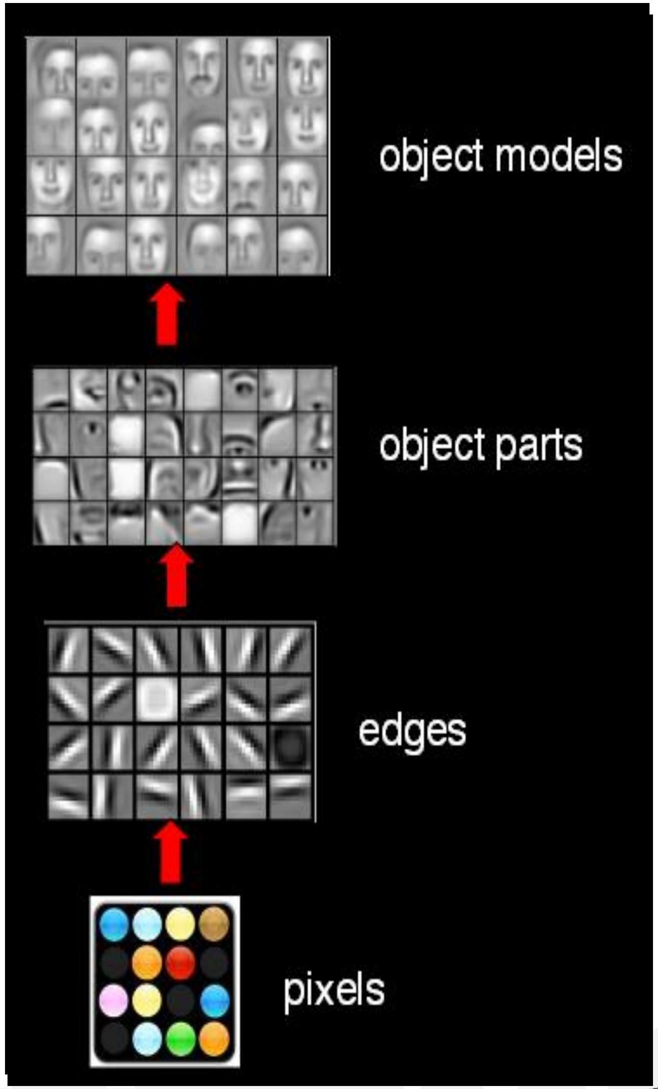
FEEDFORWARD NEURAL NETWORK



CONVOLUTIONAL NEURAL NETWORK (CNN)

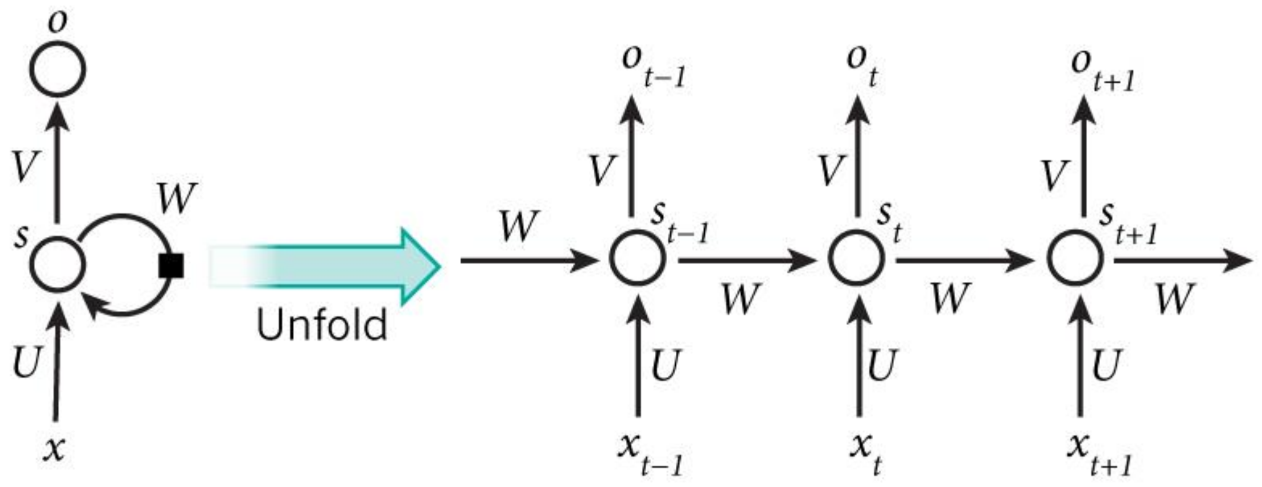






➢ Mạng nơron hồi quy RNN

- Sử dụng chuỗi các thông tin.
- Thực hiện cùng một tác vụ cho tất cả các phần tử của một chuỗi với đầu ra phụ thuộc vào cả các phép tính trước đó (có khả năng nhớ các thông tin được tính toán trước đó)
- **Ứng dụng:** xử lý ngôn ngữ tự nhiên, dự đoán từ tiếp theo, time series, tạo mô tả, nhận dạng giọng nói…

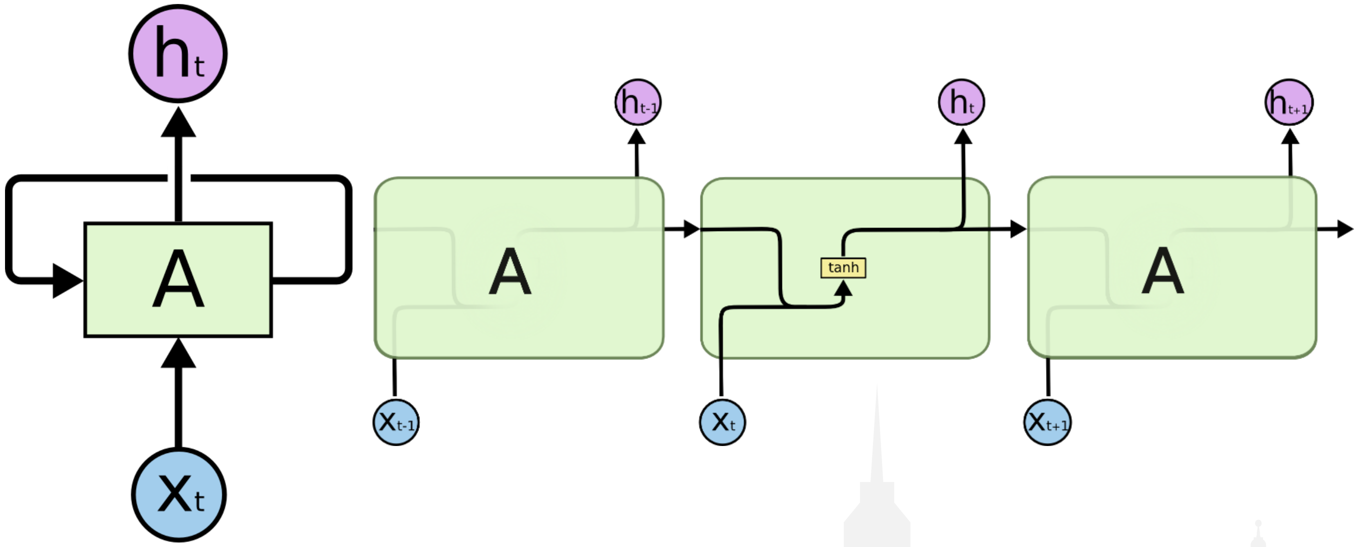


- s_t là trạng thái ẩn tại bước t. Nó chính là **bộ nhớ** của mạng. s_t được tính toán dựa trên cả các trạng thái ẩn phía trước và đầu vào tại bước đó: $s_t = f(Ux_t + Ws_{t-1})$. Hàm f thường là một hàm phi tuyến tính như tang

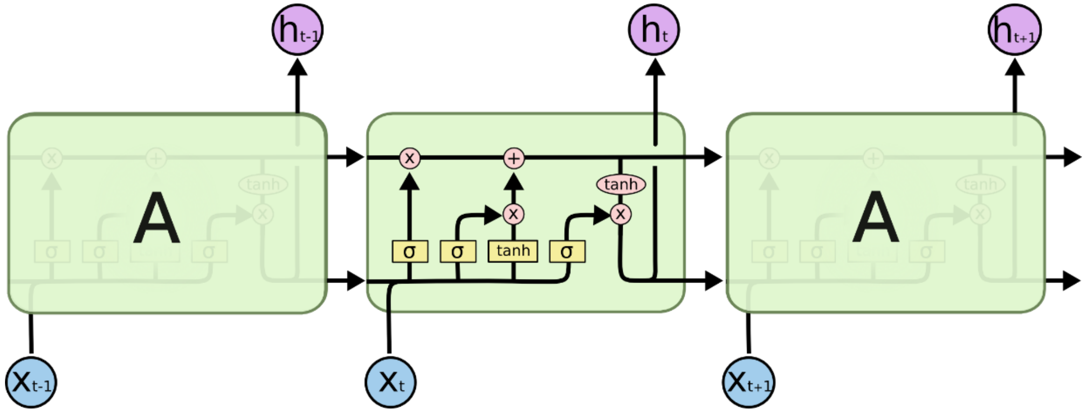
LONG-SHORT TERM MEMORY (LSTM)

➢ Mạng LSTM

- Là một dạng biến thể của RNN (1997), khắc phục vấn đề phụ thuộc xa (long-dependencies) trong chuỗi dữ liệu cần học



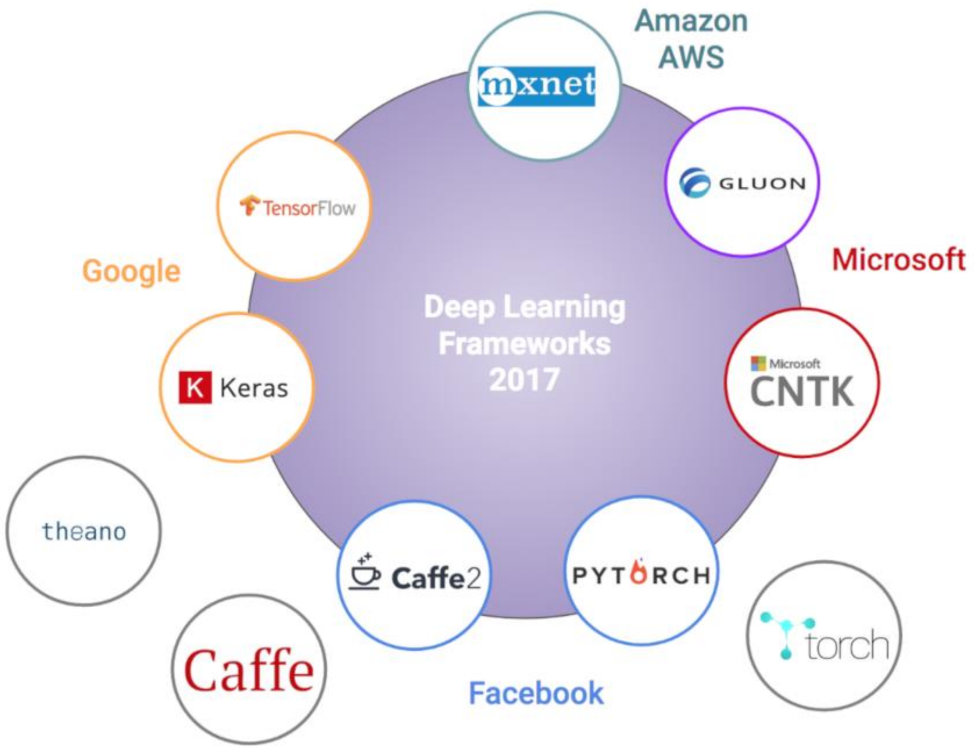
RNN

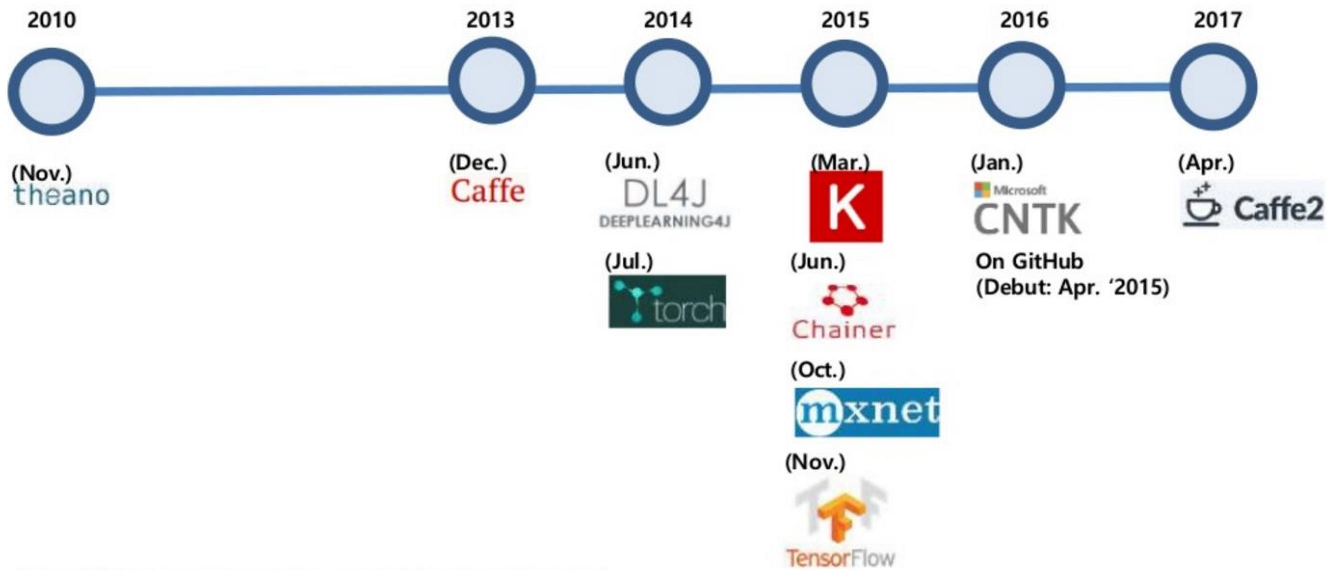


LSTM

B. MỘT SỐ FRAMEWORK DEEP LEARNING

TỔNG QUAN





AWS supports almost all popular deep learning frameworks: "You can quickly launch Amazon EC2 instances pre-installed with popular deep learning frameworks such as **Apache MXNet and Gluon, TensorFlow, Microsoft Cognitive Toolkit, Caffe, Caffe2, Theano, Torch, PyTorch, Chainer, and Keras** to train sophisticated, custom AI models, experiment with new algorithms, or to learn new skills and techniques"

https://docs.aws.amazon.com/whitepapers/latest/aws-overview/machine-learning.html

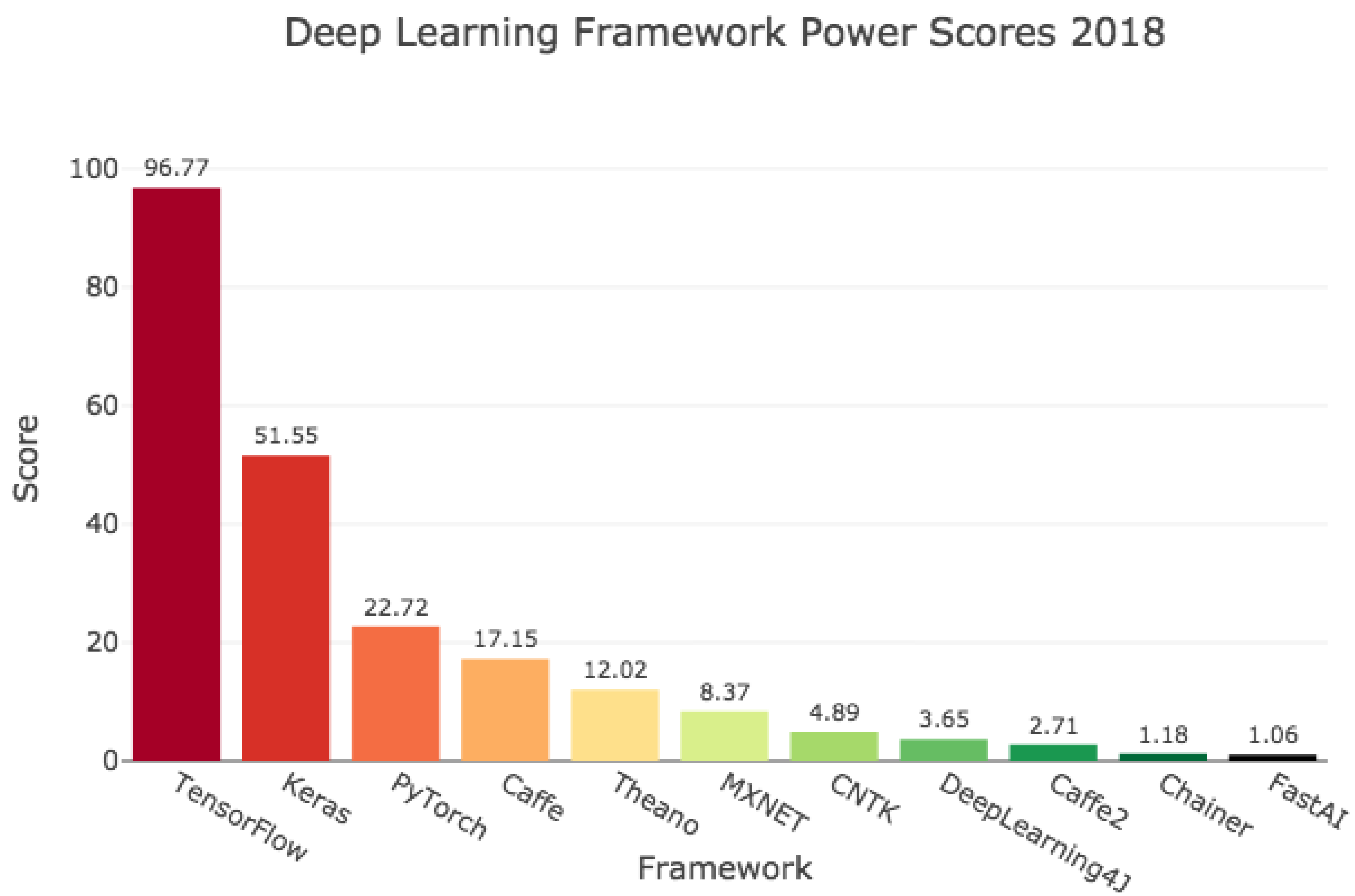
SO SÁNH CÁC FRAMEWORK DEEP LEARNING

	Languages	Tutorials and training materials	CNN modeling capability	RNN modeling capability	Architecture: easy-to-use and modular front end	Speed	Multiple GPU support	Keras compatible
Theano	Python, C++	++	++	++	+	++	+	+
Tensor-Flow	Python	+++	+++	++	+++	++	++	+
Torch	Lua, Python (new)	+	+++	++	++	+++	++	
Caffe	C++	+	++		+	+	+	
MXNet	R, Python, Julia, Scala	++	++	+	++	++	+++	
Neon	Python	+	++	+	+	++	+	
CNTK	C++	+	+	+++	+	++	+	

Each framework has its own strength based on its supported DL architectures, optimization algorithms, and ease of development and deployment

https://www.kdnuggets.com/2017/03/getting-started-deep-learning.html

SO SÁNH CÁC FRAMEWORK DEEP LEARNING



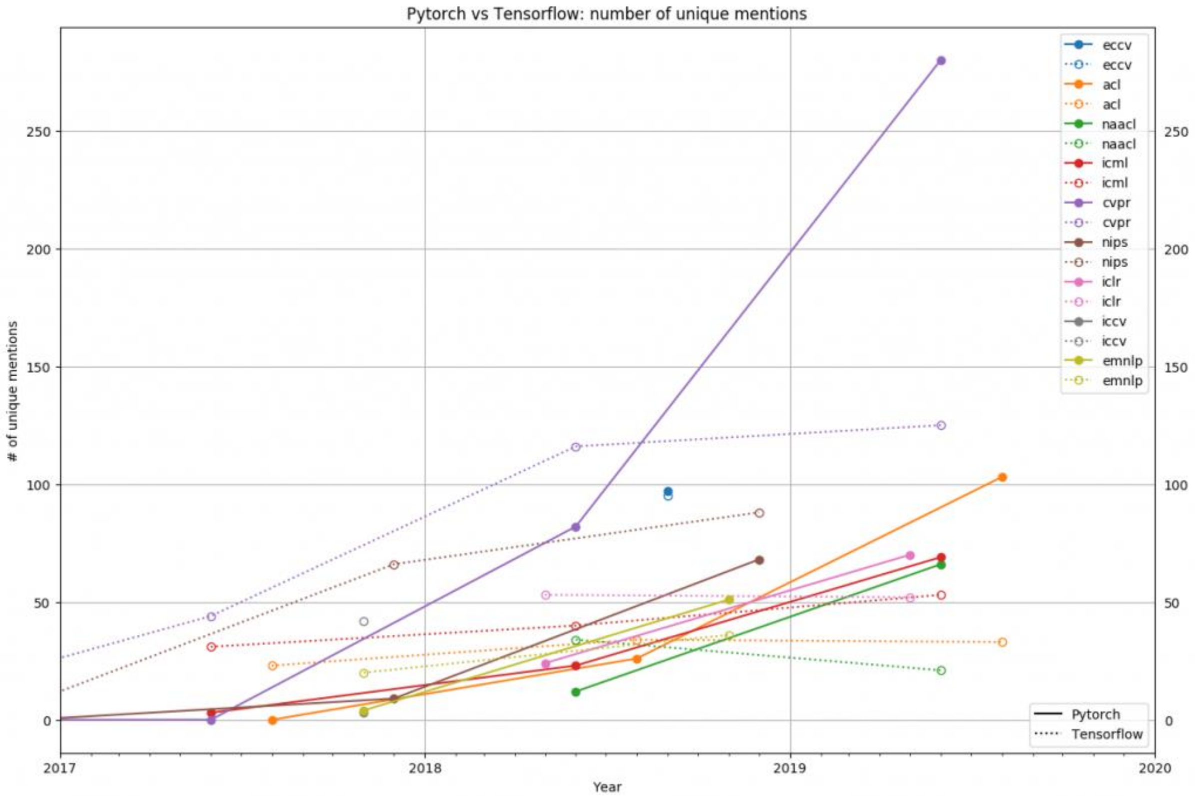
The evaluation categories are:

- **Online Job Listings**
- **KDnuggets Usage Survey**
- **Google Search Volume**
- **Medium Articles**
- **Amazon Books**
- **ArXiv Articles**
- **GitHub Activity**

https://towardsdatascience.com/deep-learning-framework-power-scores-2018-23607ddf297a

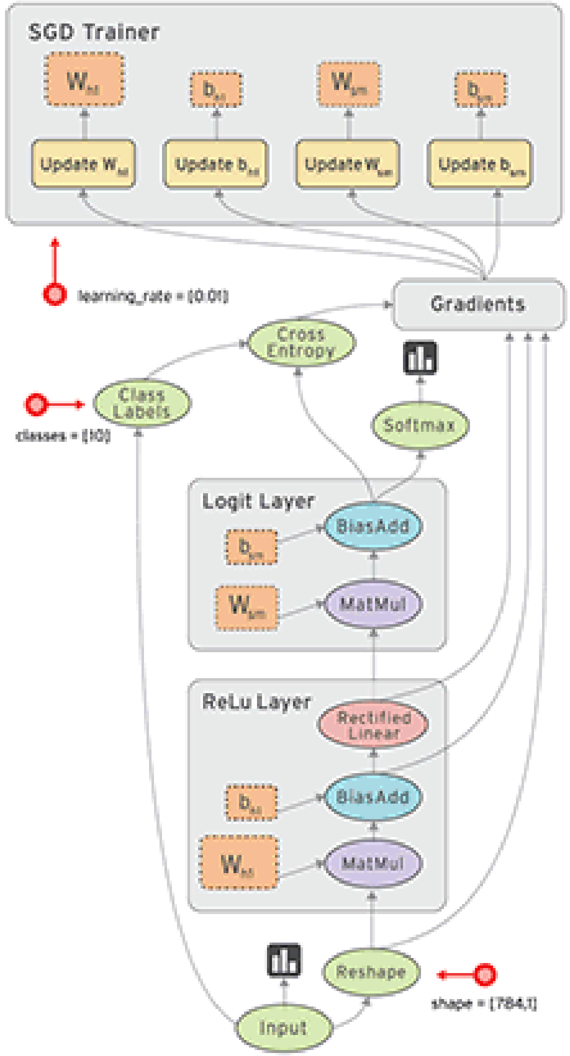
TENSORFLOW v.s Pytorch (2019)

- PyTorch for **Researchers**, TensorFlow for **Industries:**
  - From 2018 to 2019, TensorFlow had 1541 new job listings vs. 1437 job listings for PyTorch on public job boards, 3230 new TensorFlow Medium articles vs. 1200 PyTorch, 13.7k new GitHub stars for TensorFlow vs 7.2k for PyTorch, etc.



https://blog.exxactcorp.com/pytorch-vs-tensorflow-in-2020-what-you-should-know-about-these-frameworks/

TENSORFLOW, KERAS EXAMPLE



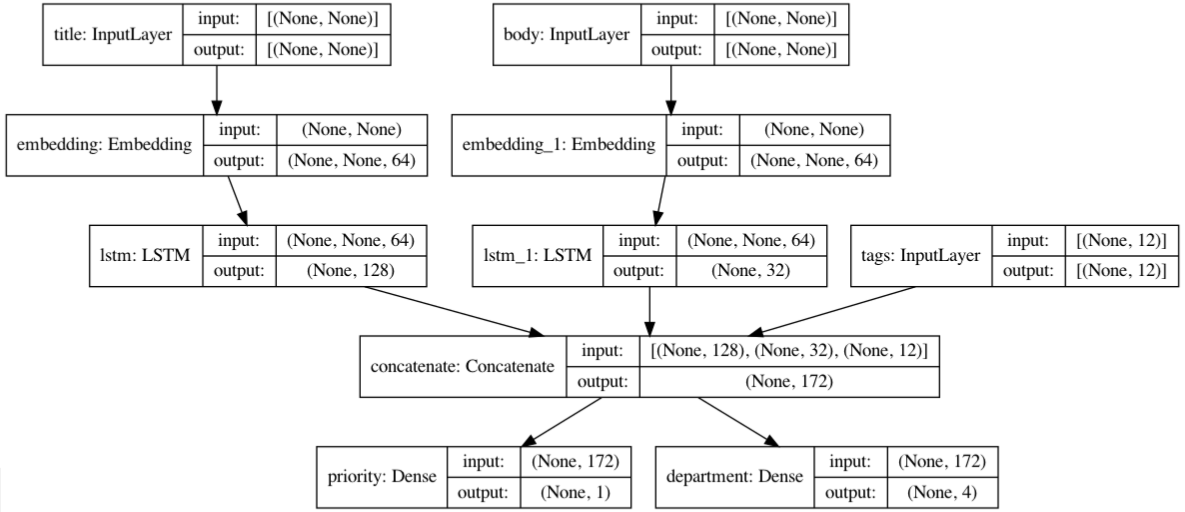
TensorFlow Estimators ← High-level, object-oriented API

tf.layers, tf.losses, tf.metrics ← Reusable libraries for common model components

Python TensorFlow ← Provides Ops, which wrap C++ Kernels

C++ TensorFlow

CPU | GPU | TPU ← Kernels work on one or more platforms

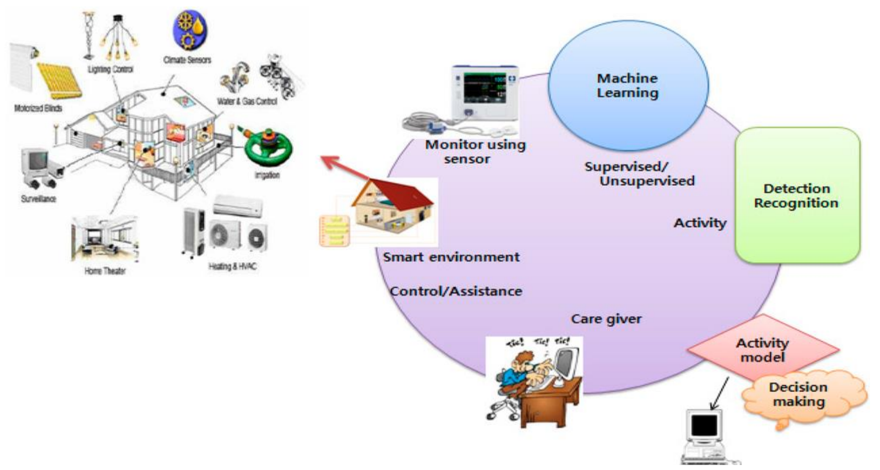


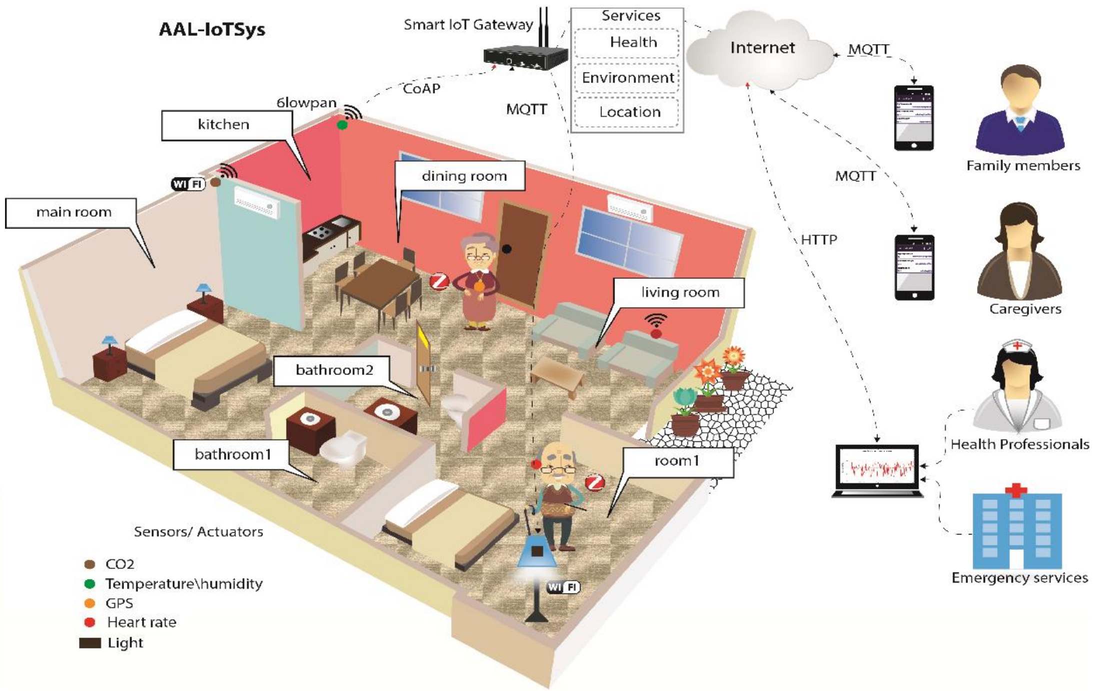
C. MỘT SỐ BÀI TOÁN, DATASET CHO IoT DATA ANALYTIC

HUMAN ACTIVITY DETECTION

ACTIVITY DETECTION IN SMART HOME

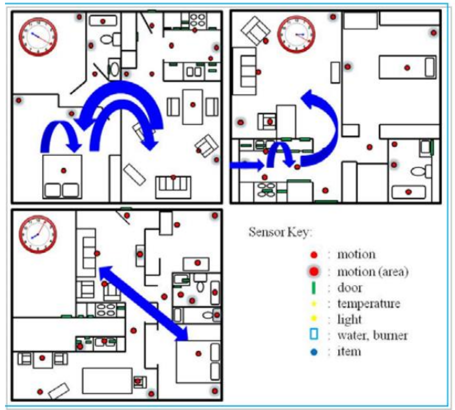
- Activity detection use various different sensor types to monitoring human activities in smart home
- Applications
  - Control HVAC (heating, ventilation, and air conditioning) and lighting systems → Automatic rule like "*turn off lights in case of sleeping activity*"
  - Ambient Assisted Living (AAL) applications which determine the wellness of elderly people, people with disabilities, or people with acute living independently in their home. Example: AbiBird used AWS (*https://aws.amazon.com/solutions/case-studies/abibird/?did=cr_card&trk=cr_card*)





ACTIVITY DETECTION IN SMART HOME

- **Real-time Activity Recognition**
- **Activity-aware applications:** Health assistance and energy-efficiency



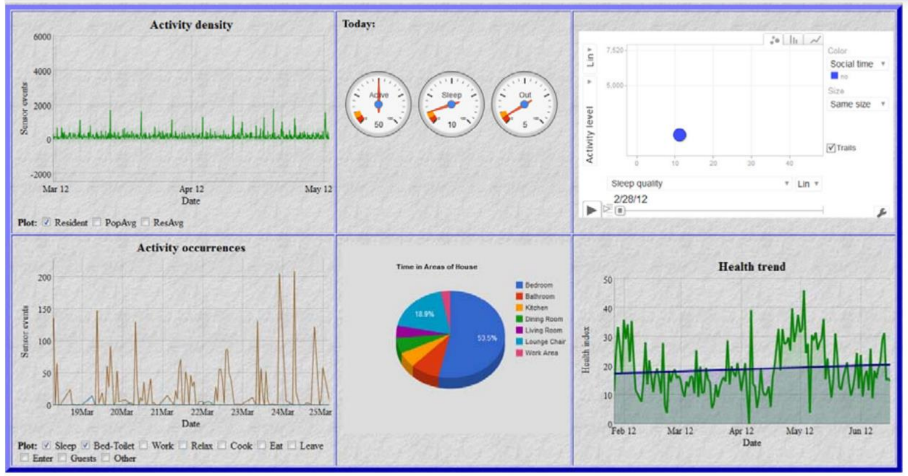


Figure 4. Activity trends for a smart home resident.

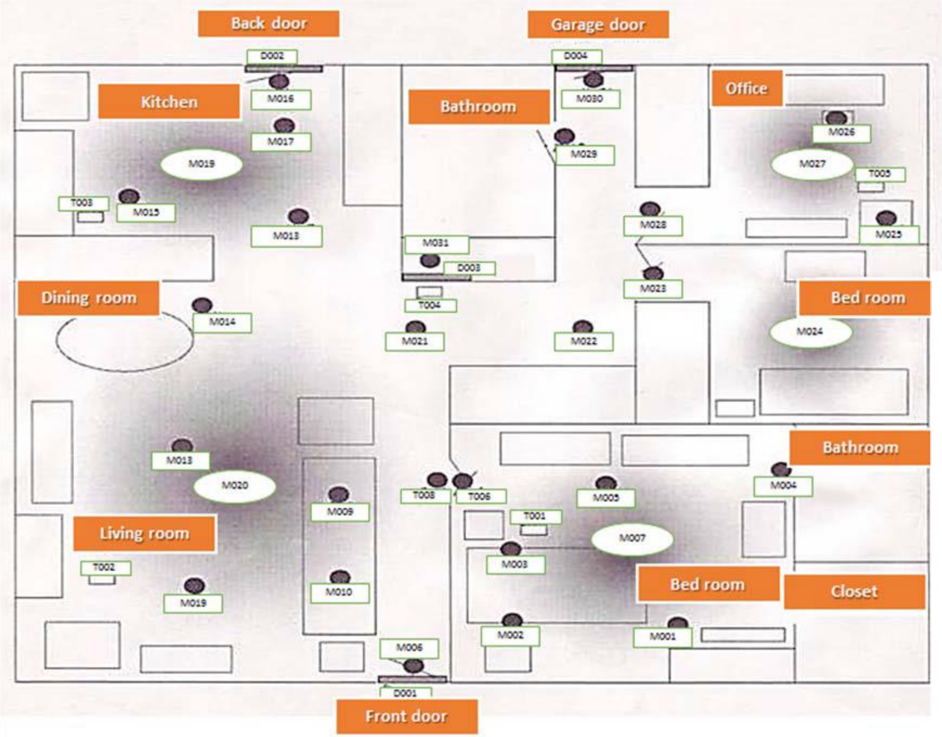
Figure 5. Snapshot of CASAS activity visualizer. The visualizer renders sensor events on a computer or mobile device while plotting usage of resources such as electricity.

- Available datasets for Activity Detection in Smart home
  - **CASAS datasets for activities of daily living** (*Washington State University*): Several public datasets related to Activities of Daily Living (ADL) performance in a two story home, an apartment, and an office settings
  - **ARAS Human Activity** (*Bogazici University*): Human activity recognition datasets collected from two real houses with multiple residents during two months.
  - **ISL** (*University of Amsterdam)*: recorded 3 single subject datasets using 3 different smart home
  - **House_n smart home dataset** (*University of MIT*), collect data by using a set of simple state change sensors. Two datasets are collected for two different subjects. Both individuals live alone in one bedroom apartments
  - …

https://hub.packtpub.com/25-datasets-deep-learning-iot/

CASAS dataset

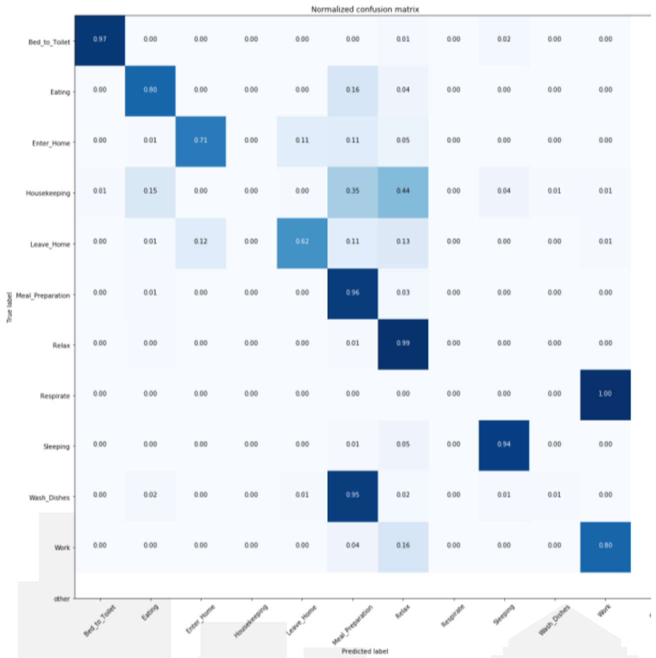
- Example Aruba dataset



https://medium.com/@nickmal/real-time-activity-recognition-in-a-smart-home-using-binary-sensors-efd147ec694

- 3 types of sensors: M (motion), D (door closure), T (temperature)
  - In other datasets, + different sensor types: light switch, light sensor…
- Sensor activations were collected over time
- 11 activities are labelled: Meal_Preparation (1606), Relax (2910), Eating (257), Work (171), Sleeping (401), Wash_Dishes (65), Bed_to_Toilet (157), Enter_Home (431), Leave_Home (431), Housekeeping (33), Resperate (6)

2010-11-04	05:40:51.303739	M004	ON	Bed_to_Toilet	begin
2010-11-04	05:40:52.342105	M005	OFF		
2010-11-04	05:40:57.176409	M007	OFF		
2010-11-04	05:40:57.941486	M004	OFF		
2010-11-04	05:43:24.021475	M004	ON		
2010-11-04	05:43:26.273181	M004	OFF		
2010-11-04	05:43:26.345503	M007	ON		
2010-11-04	05:43:26.793102	M004	ON		
2010-11-04	05:43:27.195347	M007	OFF		
2010-11-04	05:43:27.787437	M007	ON		
2010-11-04	05:43:29.711796	M005	ON		
2010-11-04	05:43:30.279021	M004	OFF	Bed_to_Toilet	end
2010-11-04	05:43:45.7324	M003	ON	Sleeping	begin
2010-11-04	05:43:52.044085	M003	OFF		
2010-11-04	05:43:53.185335	M002	ON		
2010-11-04	05:43:53.253809	M003	ON		
2010-11-04	05:43:59.493281	M002	OFF		
2010-11-04	05:44:04.048766	M003	OFF		
2010-11-04	05:44:06.14204	M003	ON		
2010-11-04	05:44:11.229146	M003	OFF		

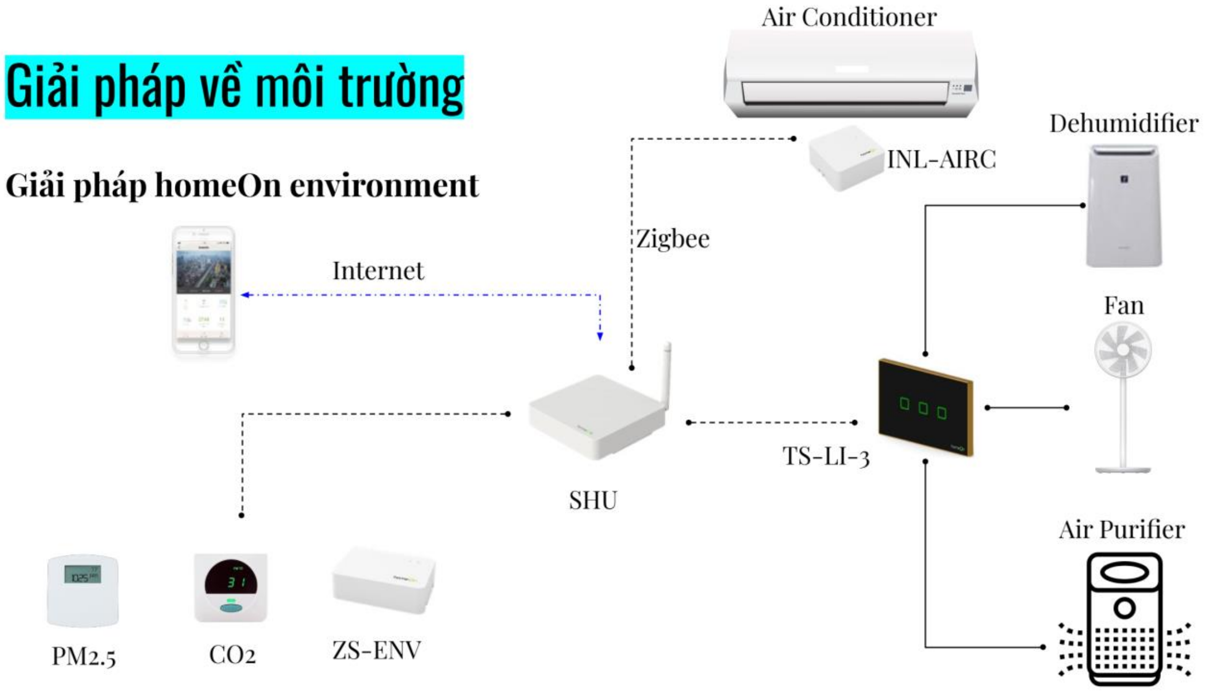


INDOOR AIR QUALITY MONITORING

➢ Nhu cầu của công ty smarthome homeOn (Việt Nam)

Giải pháp về môi trường

Giải pháp homeOn environment



+ SHU tiếp nhận các thông số môi trường như chỉ số PM 2.5, CO_2, nhiệt độ, độ ẩm.
+ Dựa trên các thông số môi trường và kịch bản được cài đặt, SHU sẽ ra lệnh điều khiển tới các thiết bị điều hòa nhiệt độ, máy hút ẩm, quạt thông gió, máy tạo không khí tươi... để giữ cho điều kiện môi trường ở trạng thái cân bằng và tươi sạch nhất.

- ➢ Problem 1: **Phát hiện các điểm bất thường trong data các sensor gửi lên (bài toán abnormal detection) -> Mục đích thống kê**
  - Sample in AWS: *Amazon SageMaker Random Cut Forest (RCF)* is an unsupervised algorithm for detecting anomalous data points within a data set
- ➢ Problem 2: Dựa trên dữ liệu từ các sensor để đưa ra đánh giá về mức độ ô nhiễm (**bài toán classification**)
  - *https://figshare.com/articles/Pollutant_Recognition_Based_on_Supervised_Machine_Learning_for_Indoor_Air_Quality_Monitoring_Systems/5306506*

Xin cảm ơn đã lắng nghe!