Artificial Neural Network (ANN)

- **Outline:**

1. Overview of ANN
2. Components of ANN
3. ANN training (forward and backward propagation)
4. ANN characteristics
5. ANN design

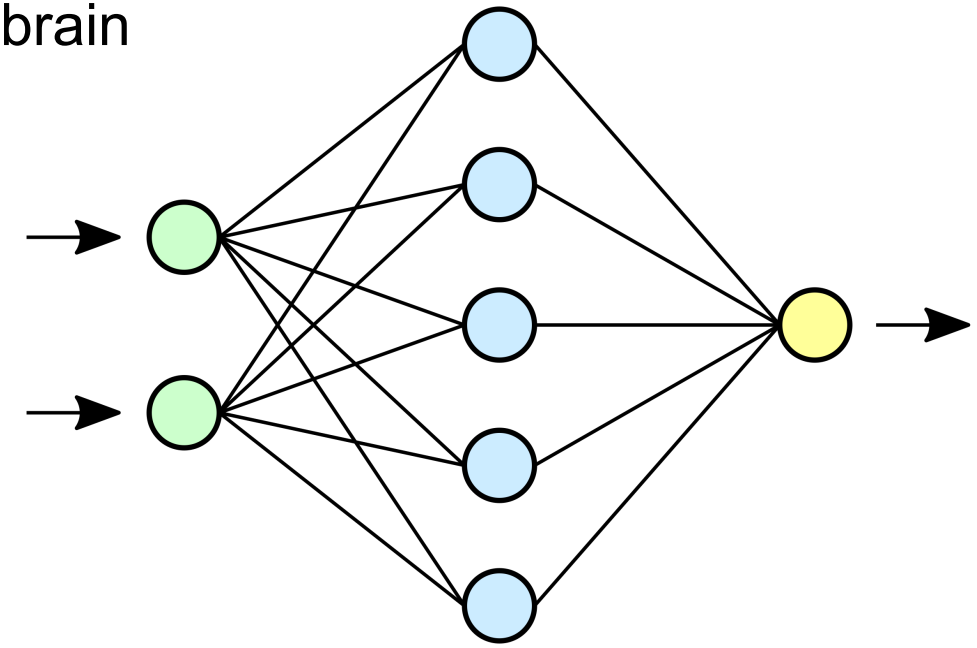
Artificial Neural Network (ANN)

- **Outline:**

1. **Overview of ANN**
2. Components of ANN
3. ANN training (forward and backward propagation)
4. ANN characteristics
5. ANN design

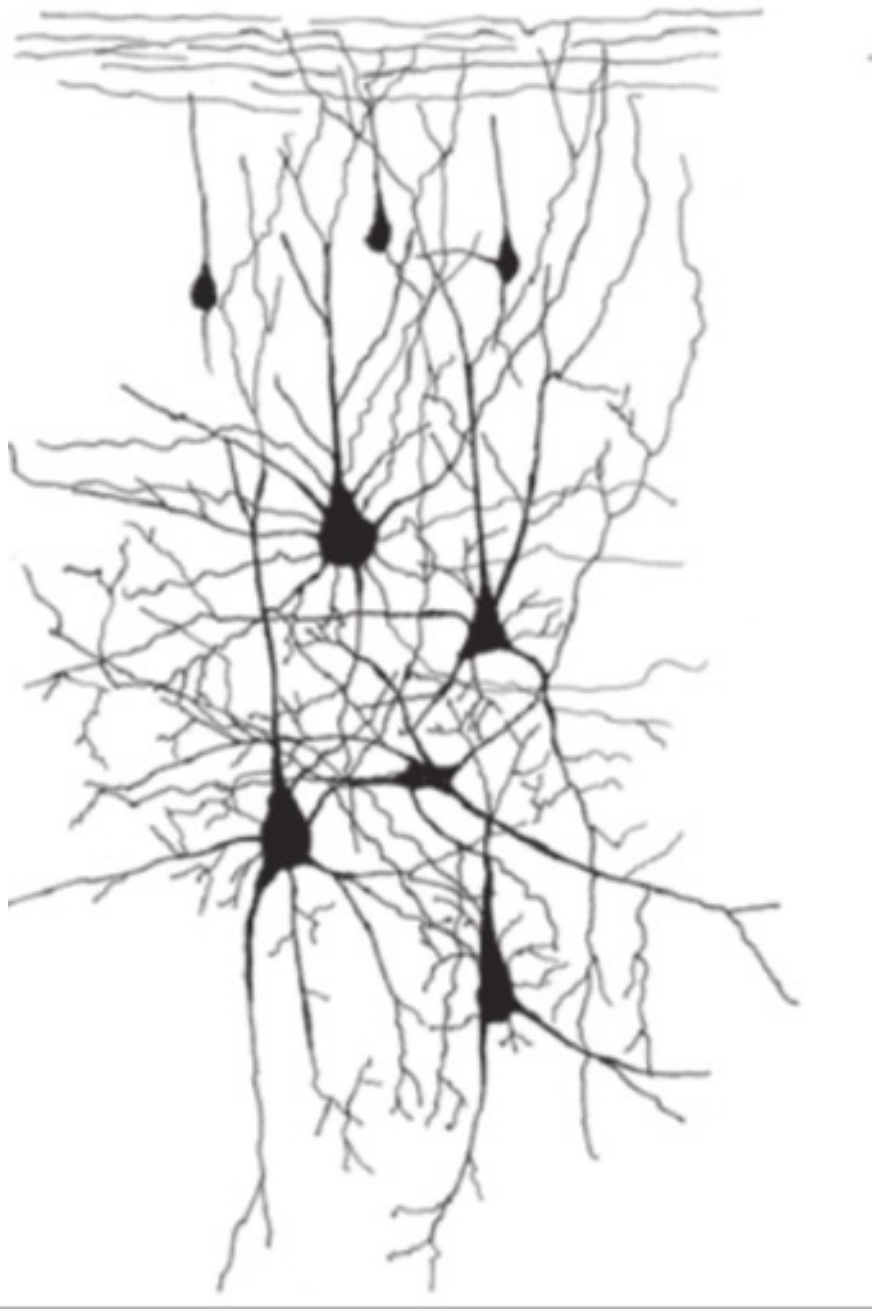
What is an ANN?

- Computing system inspired by the biological neural network
- Connection of units (or nodes) called artificial neurons
  - Each node ~ biological neuron
  - Each connection ~ synapse in brain

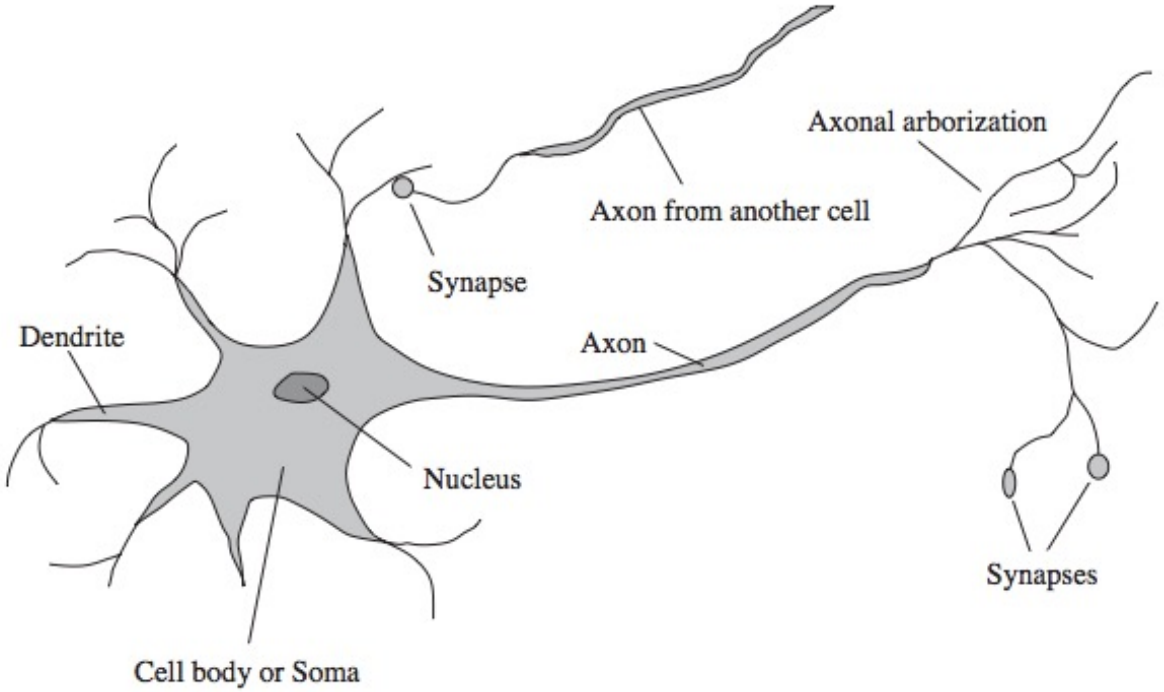


Biological neural networks

- Our brain consists of ~ 100 billion neurons
- A neuron may connect to as many as 100,000 other neurons
- Signals “move” via electrochemical signals
- **Biological neural network:** interconnected network of billions of neurons with trillions of interconnections between them



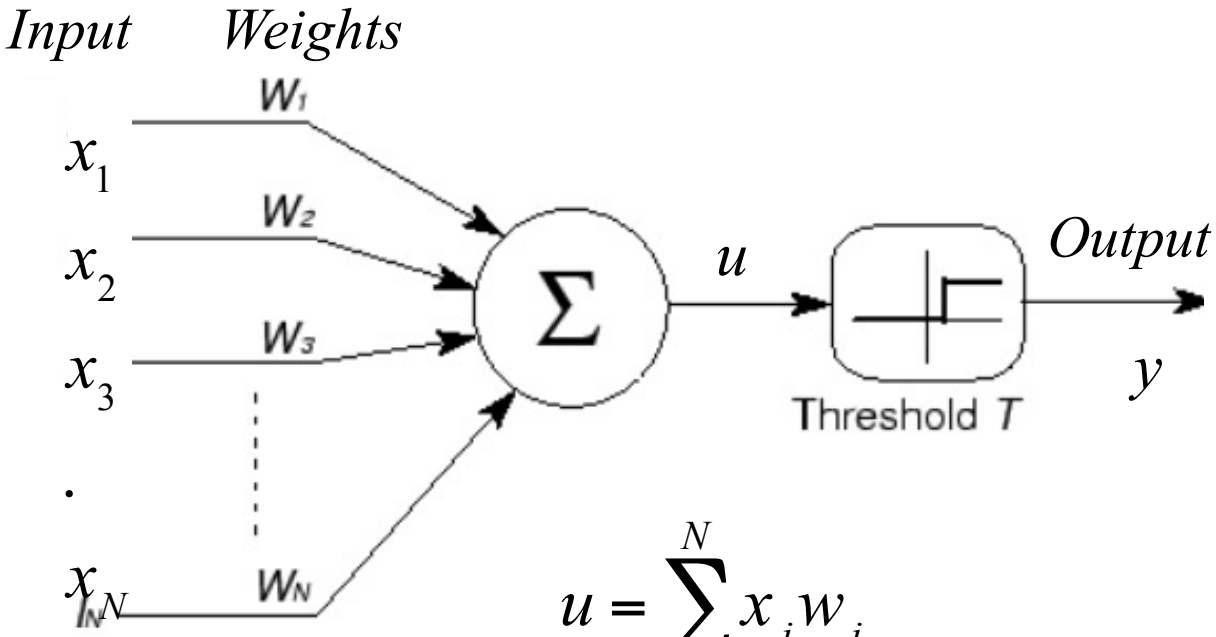
Structure of a biological neuron



- **Dendrite:** receives signals from other neurons
- **Cell body:** sums all the incoming signals to generate input
- **Axon:** transfers signals to the other neurons when the neuron “fires” (sum reaches a threshold)
- **Synapses:** points of interconnection of one neuron with other neurons

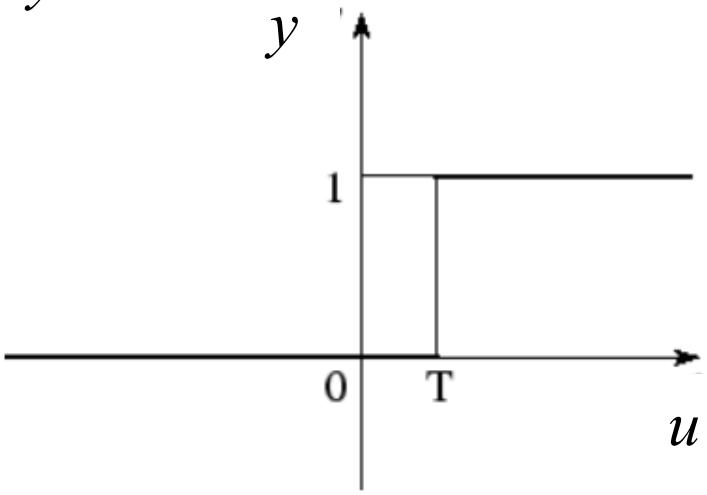
McCulloch and Pitts neuron model (1943)

- A mathematic computing paradigm that models the human neuron



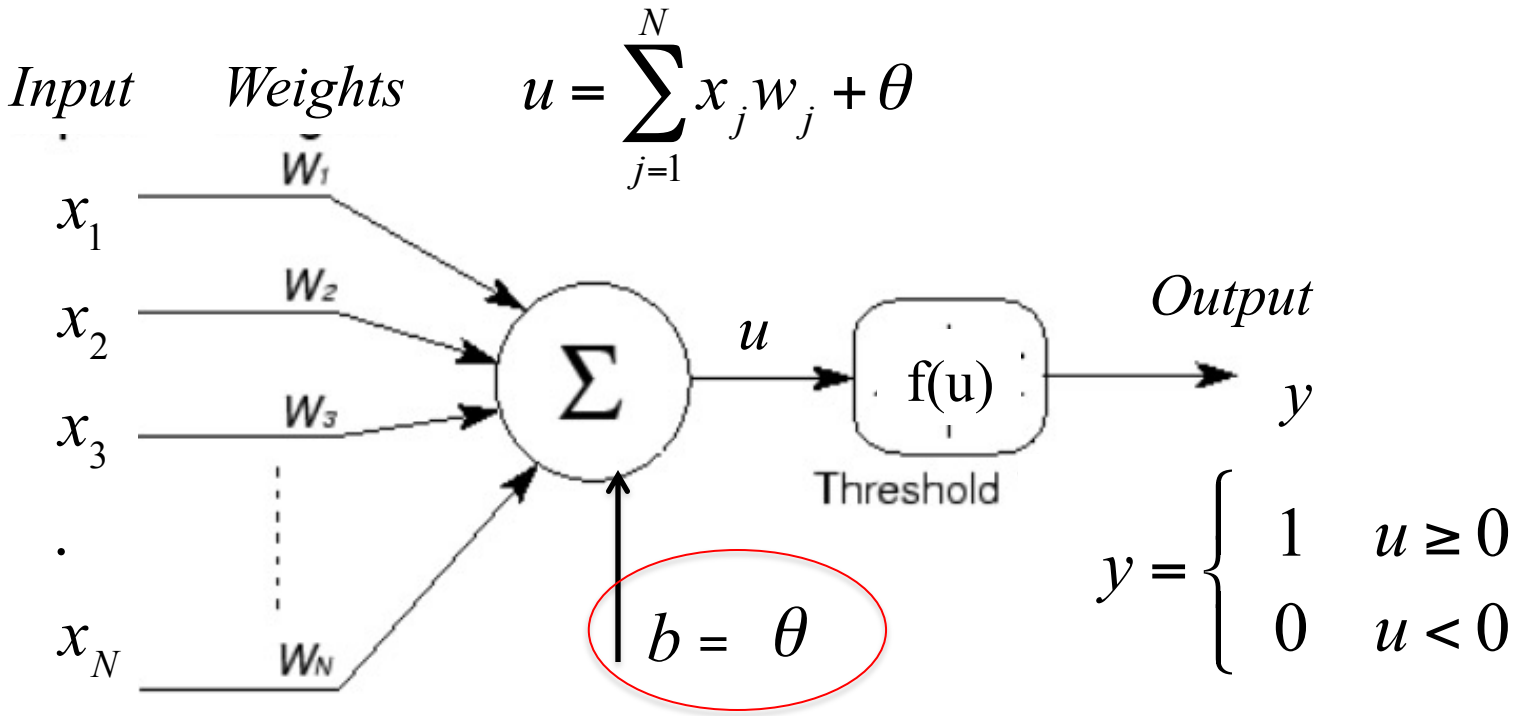
$$u = \sum_{j=1}^{N} x_j w_j$$

$$y = f(u)$$



Perceptron neuron model

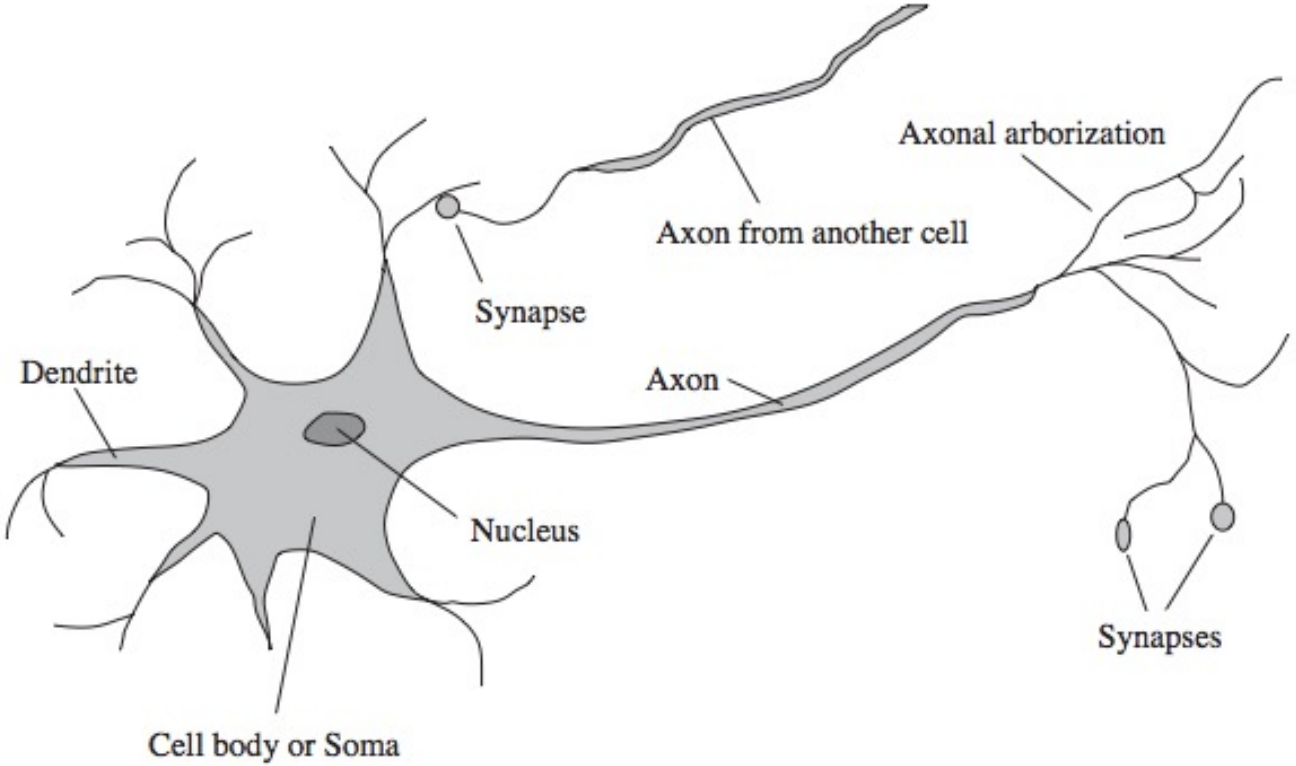
- An enhanced version of McCulloch-Pitts model:
- Merge Hebbian learning rule of adjusting weights
- Add bias

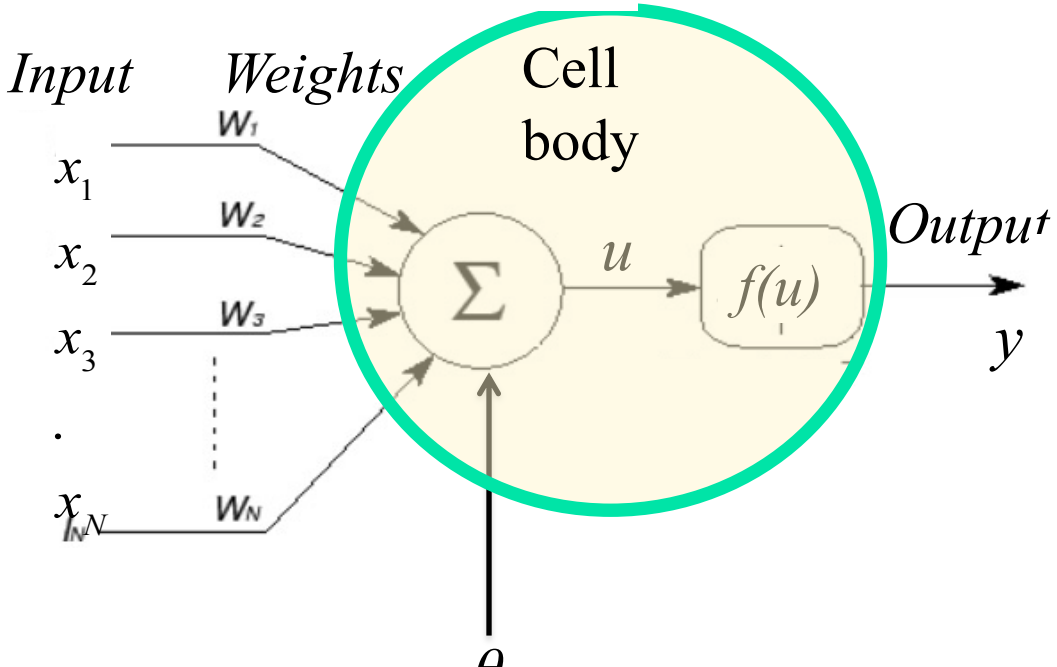


Artificial Neural Network (ANN)

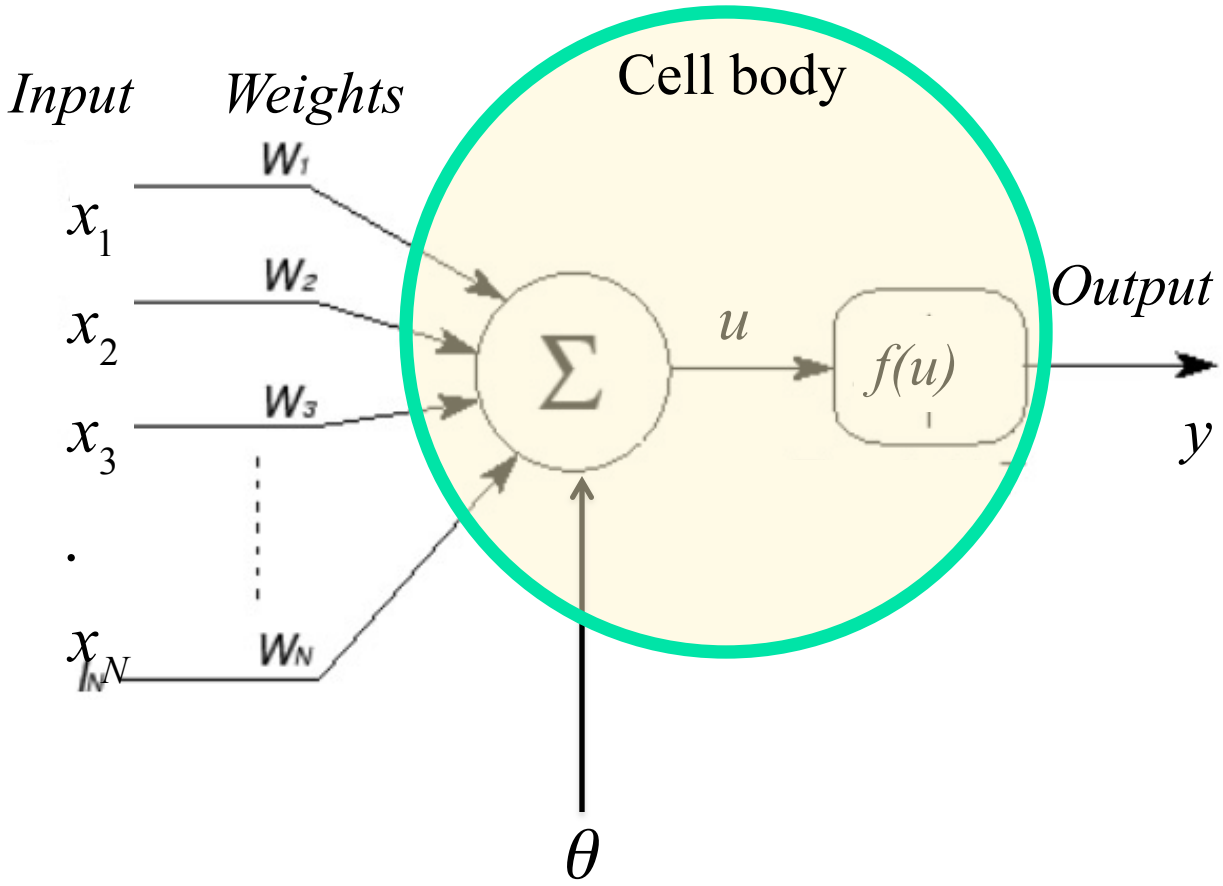
- **Outline:**

1. Overview of ANN
2. **Components of ANN**
3. ANN training (forward and backward propagation)
4. ANN characteristics
5. ANN design


Axonal arborization
Axon from another cell
Synapse
Dendrite
Axon
Nucleus
Synapses
Cell body or Soma


Input
Weights
Cell body
x_1
x_2
x_3
x_N
w_1
w_2
w_3
w_N
Σ
u
$f(u)$
Output
y
θ

General neuron model



$\{w_j;\ 1 \le j \le N\}$: *synaptic weights*

θ : *threshold*

Net function

$$u = \sum_{j=1}^{N} x_j w_j + \theta$$

Activation function

$$y = f(u)$$

Ex:

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

Popular net functions

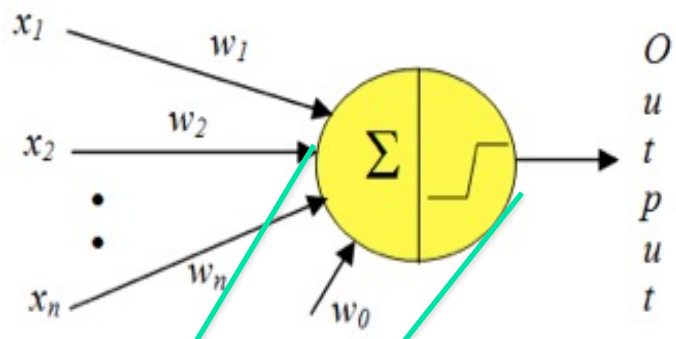
Net functions	*Formula*	*Comments*
Linear	$u = \sum_{j=1}^{N} w_j y_j + \theta$	Most commonly used.
Higher order (2nd order formula exhibited)	$u = \sum_{j=1}^{N} \sum_{k=1}^{N} w_{jk} y_j y_k + \theta$	u_i is a weighted linear combination of higher order polynomial terms of input variable. The number of input terms equals to N^d where d is the order of the polynomial.
Delta (Σ-Π)	$u = \prod_{j=1}^{N} w_j y_j$	Seldom used

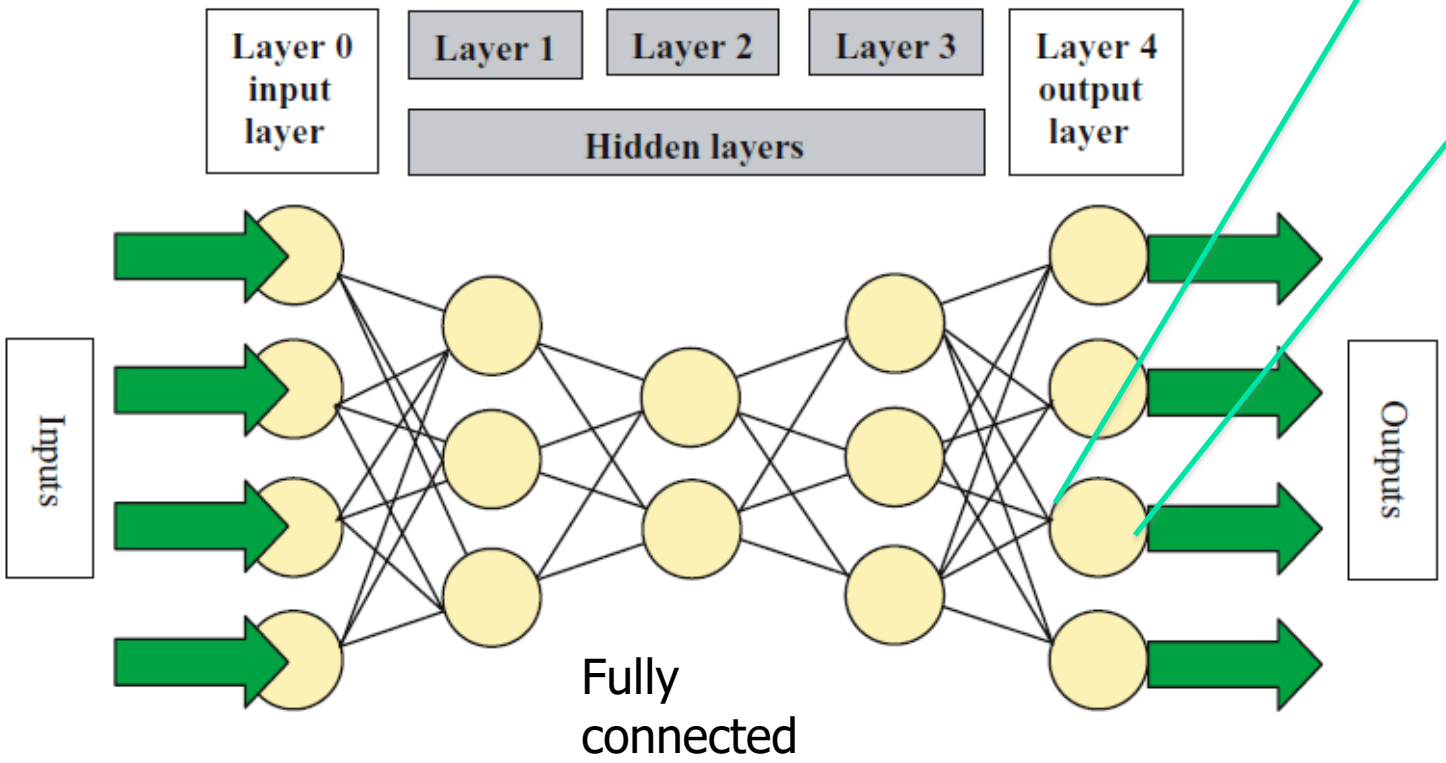
Popular activation functions

Activation function	*Formula* $a = f(u)$	*Derivatives* $\frac{df(u)}{du}$	*Comments*
Sigmoid	$f(u) = \frac{1}{1+e^{-u/T}}$	$f(u)[1-f(u)]/T$	Commonly used. Derivative can be computed from *f(u)* directly. T: temperature parameter.
Hyperbolic tangent	$f(u) = \tanh\left(\frac{u}{T}\right)$	$(1-[f(u)]^2)/T$	
Inverse tangent	$f(u) = \frac{2}{\pi}\tan^{-1}\left(\frac{u}{T}\right)$	$\frac{2}{\pi T}\cdot\frac{1}{1+(u/T)^2}$	Less frequently used.
Threshold	$f(u) = \begin{cases} 1 & u > 0; \\ -1 & u < 0. \end{cases}$	Derivatives does not exist at $u = 0$.	
Gaussian radial basis	$f(u) = \exp[-\|u-m\|^2/\sigma^2]$	$-2(u-m)\cdot f(u)/\sigma^2$	Used for radial basis neural network. m and σ^2 are parameters to be specified.
Linear	$f(u) = au + b$	a	

Multilayer perceptron model (MLP)

- Layered network of perceptron neurons





Artificial Neural Network (ANN)

- **Outline:**

1. Overview of ANN
2. Components of ANN
3. **ANN training (forward and backward propagation)**
4. ANN characteristics
5. ANN design

ANN training process?

- **Calibrating all of the weights** by repeating forward-backward propagation steps until the output is predicted accurately
- **Forward propagation:**
  - Applying a set of weights to the input data
  - Calculating the output
- **Backward propagation:**
  - Measuring the error of the output (difference between desired output and actual output)
  - Adjusting the weights to decrease the error in the next step

ANN training example – epoch 1

INPUT DATA	DESIRED VALUE	ACTUAL OUTPUT
	0	0.21
	0	0.156
	1	0.78
	1	0.83

ANN training example – epoch 2

INPUT DATA	DESIRED VALUE	ACTUAL OUTPUT
	0	0.194
	0	0.143
	1	0.802
	1	0.895

ANN training example – epoch n

INPUT DATA	DESIRED VALUE	ACTUAL OUTPUT
	0	0.119
	0	0.056
	1	0.884
	1	0.926

Error back propagation learning

- Step 1: initialization
- Step 2: output calculating
- Step 3: error calculating

$$E = \sum_{k=1}^{K} [e(k)]^2 = \sum_{k=1}^{K} [d(k) - z(k)]^2$$

d – desired output values (target)

z – actual outputs

- Step 4: weight updating then go back to step (2) until the stop condition is satisfied

Weight updating

- To achieve the minimum error $\mathbf{W}(t+1) = \mathbf{W}(t) + \Delta\mathbf{W}(t)$

Algorithm	$\Delta W(t)$	*Comments*
Steepest descend gradient method	$= -\eta\, g(t) = -\eta\ dE/dW$	*g is known as the gradient vector. η is the step size or learning rate. This is also known as error back-propagation learning*
Newton's method	$= -H^{-1}g(t)$ $= -[d^2E/dW^2]^{-1}(dE/dW)$	*H is known as the Hessian matrix. There are several different ways to estimate it.*
Conjugate-Gradient method	$= \eta\, p(t)$ *where* $p(t+1) = -g(t+1) + \beta\, p(t)$	

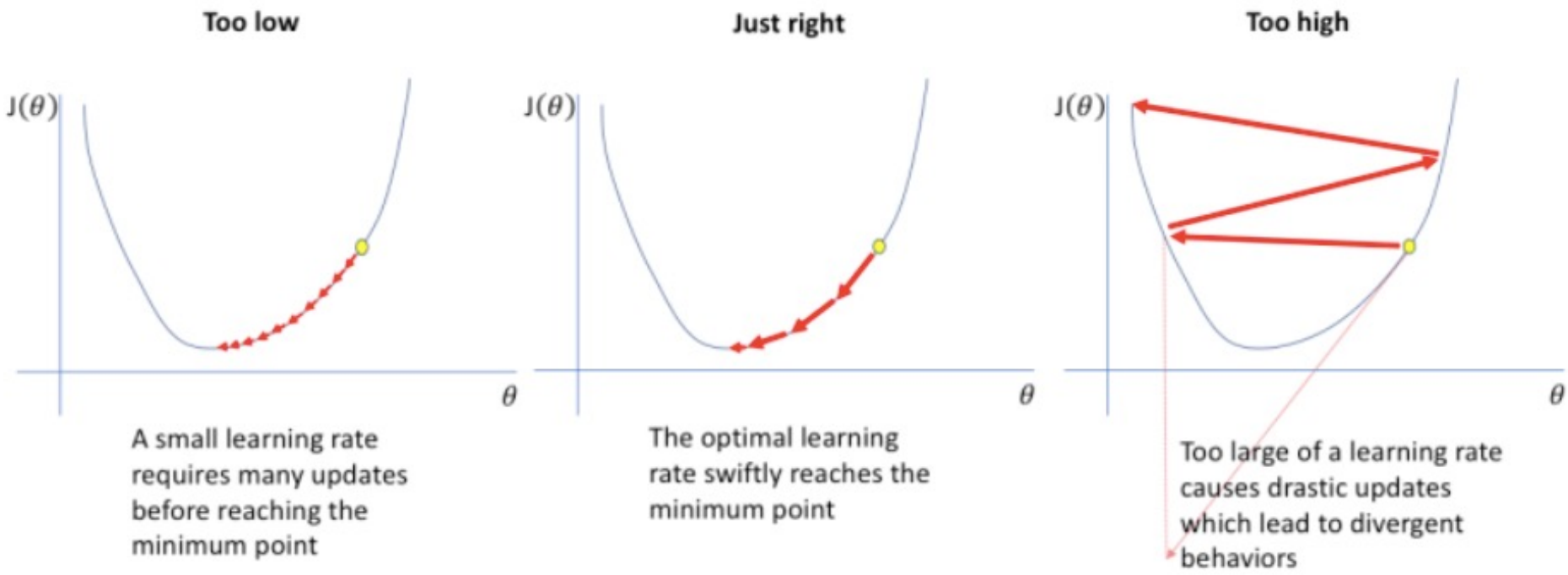
η – *learning rate*

W – weight

g - gradient vector

E – error

Learning rate choosing



Stopping conditions

- **Average squared error change:** the absolute rate of change in the average squared error per epoch is sufficiently small (in the range [0.1, 0.01]).
- **Generalization based criterion:** after each epoch the ANN is tested for generalization. If the generalization performance is adequate then stop.
- **Good generalization:** the I/O mapping is nearly correct for new data

Artificial Neural Network (ANN)

- **Outline:**

1. Overview of ANN
2. Components of ANN
3. ANN training (forward and backward propagation)
4. **ANN characteristics**
5. ANN design

ANN characteristics

- Parameters, hyperparameters
- Shallow NN, deep NN
- Underfitting, overfitting
- Generalization

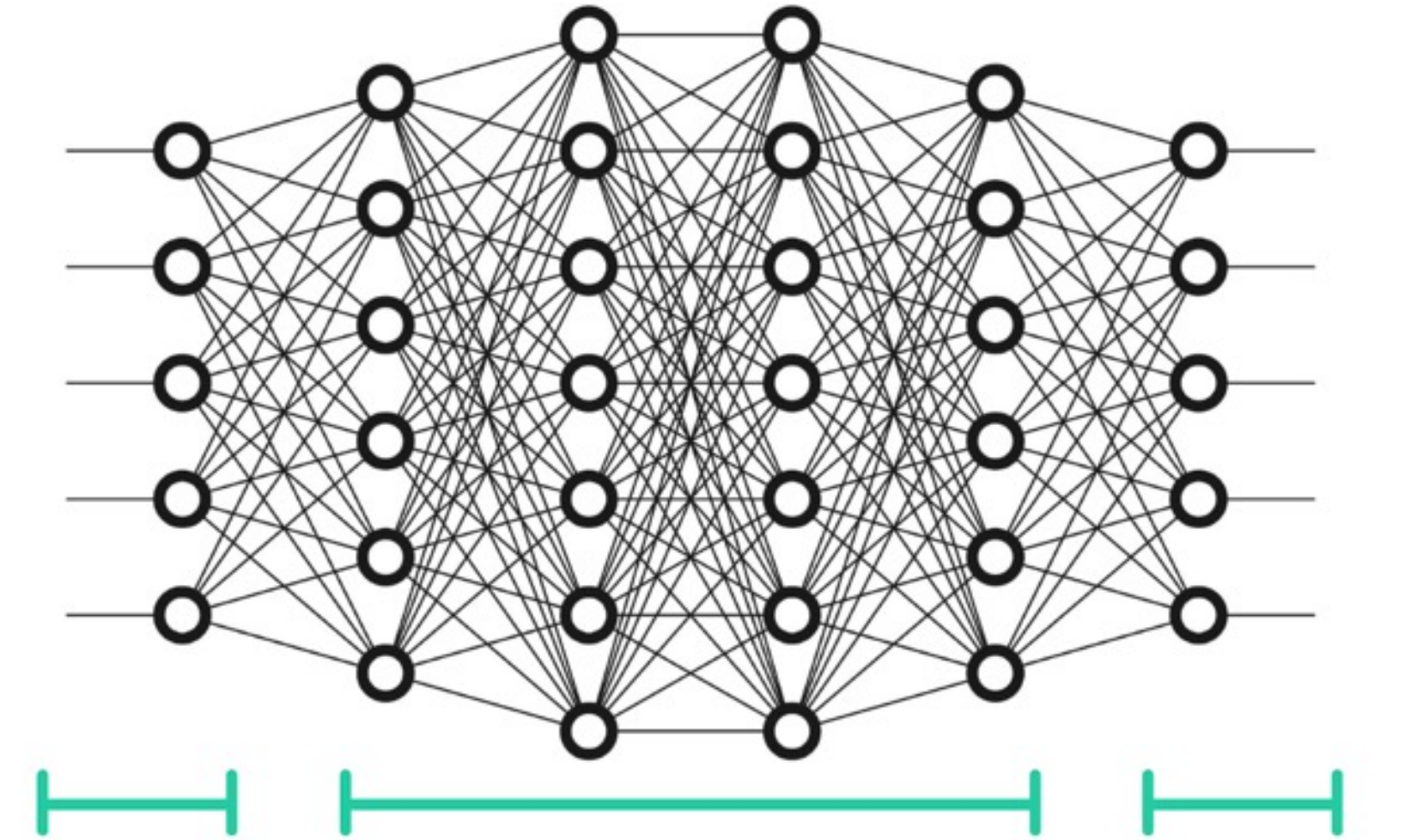
ANN parameters

- **Parameters:** changing while training ANN
  - Weights
  - Biases
- **Hyperparameters:** constant parameters related to ANN configuration defined before training ANN
  - Learning rate
  - Number of hidden layers
  - Net function
  - Activation function,
  - Number of examples in the training dataset…

Shallow NN and deep NN

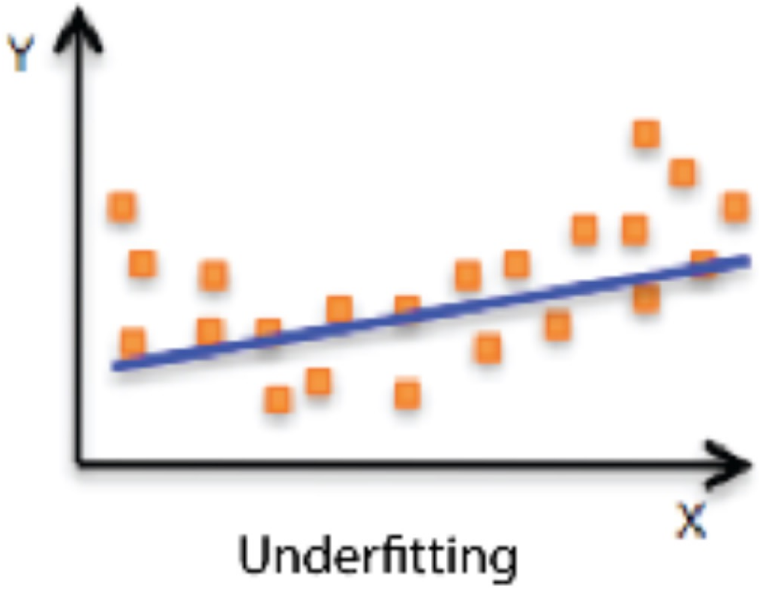
- **Shallow NN:**
  - One hidden layer
  - Used for simple problems
- **Deep NN:**
  - Many hidden layers
  - Used for complex problems
  - Each layer is used for a specific role in the entire problem

Deep neural network



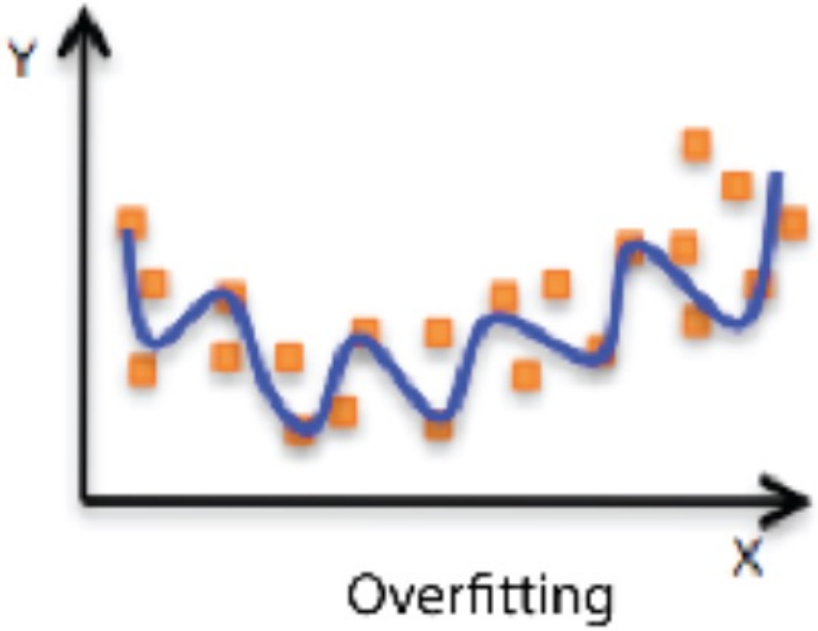
Model fitting

- **Underfitting (high bias):**
  - Model is too simple for data
  - Train error is large, vali/test error is large too.
  - Model can do accurate predictions, but the initial assumption about the data is incorrect

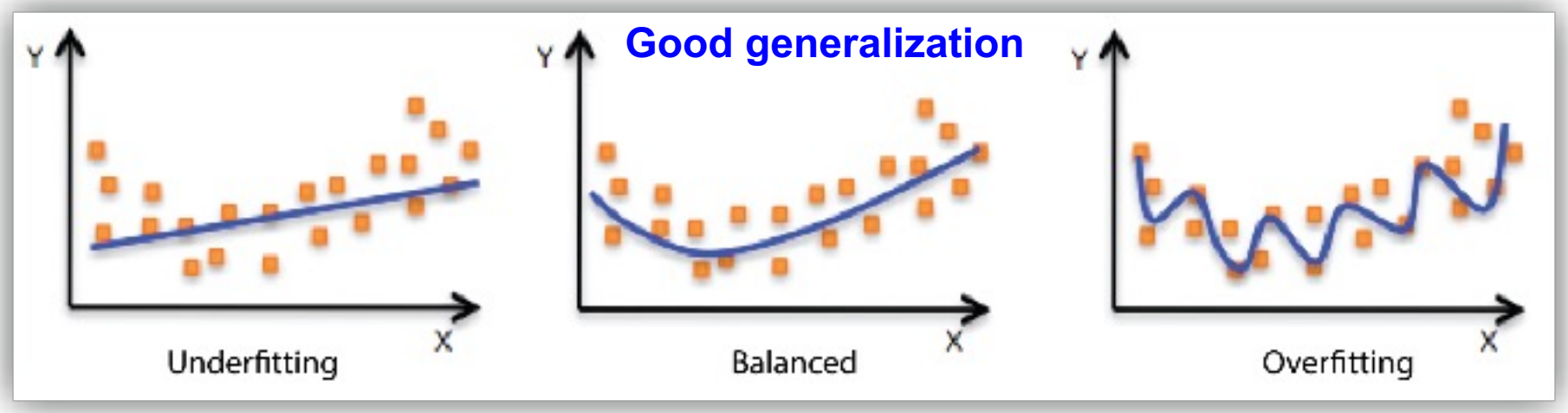


Model fitting (cont)

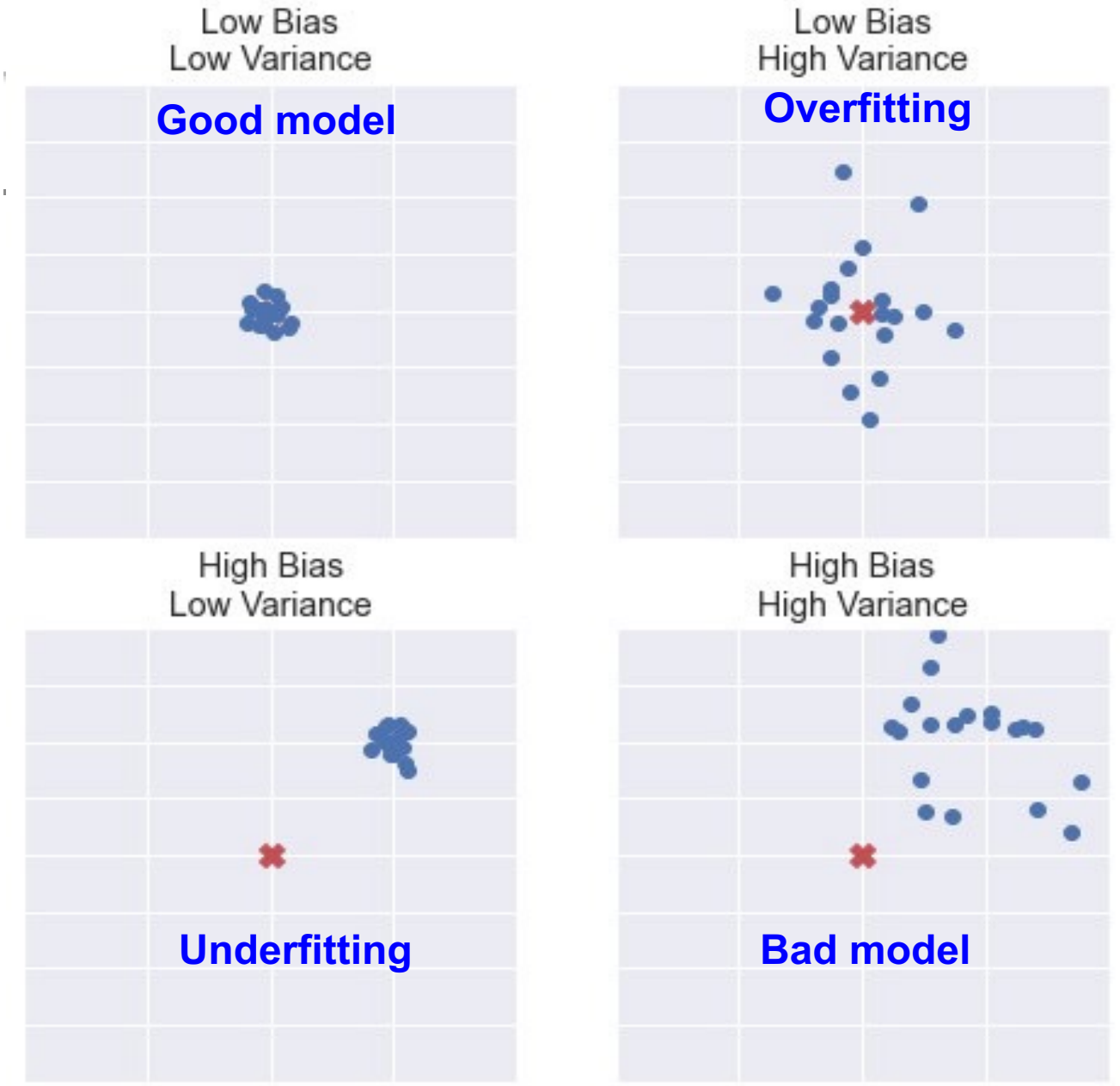
- **Overfitting (high variance):**
  - Model is too complex for data
  - Model memorizes the training data rather than generalize the data → error on training set is small, error on testing set is large

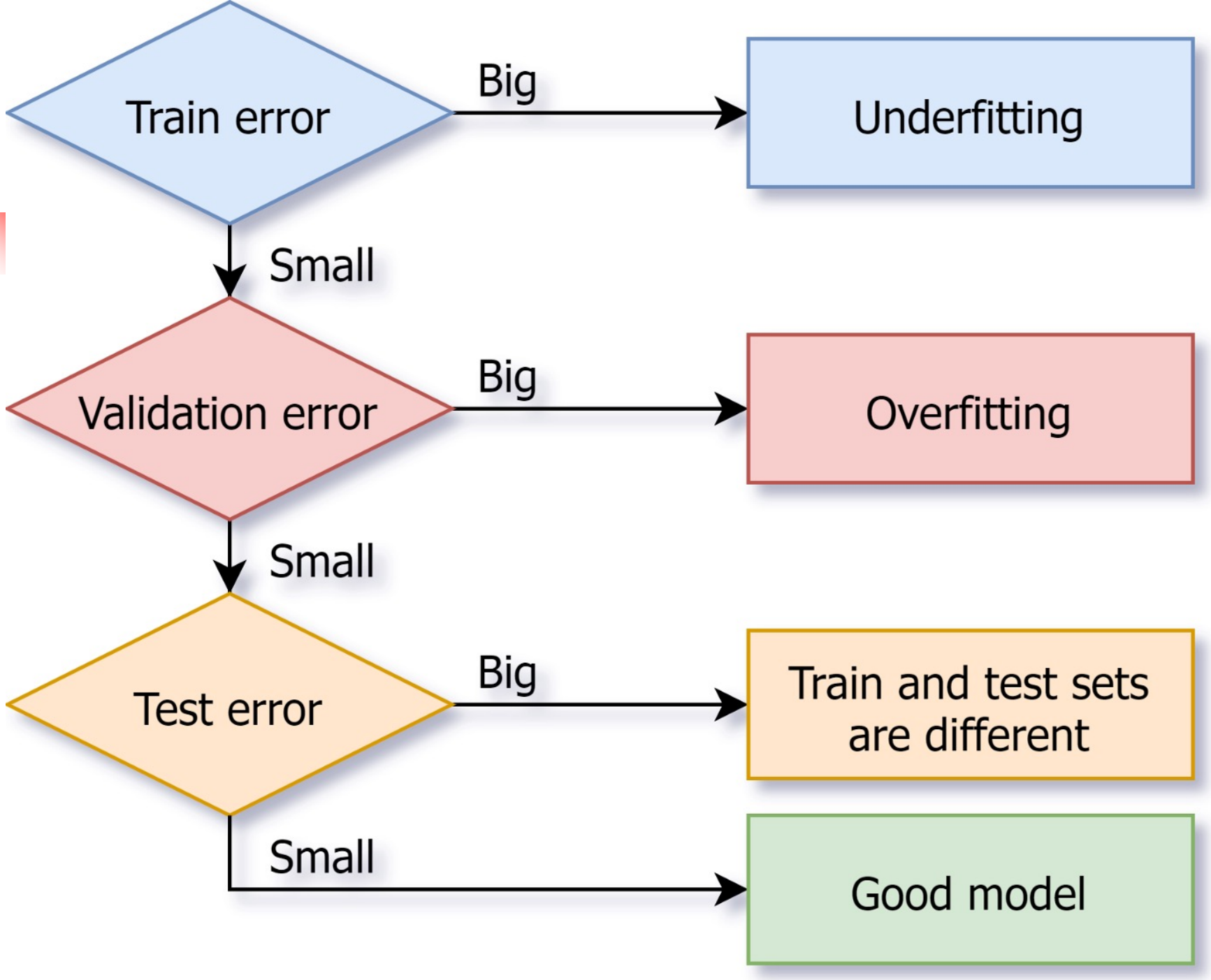


Model fitting (cont)



Model fitting (cont)




Train error
Big
Underfitting
Small
Validation error
Big
Overfitting
Small
Test error
Big
Train and test sets are different
Small
Good model

How to avoid underfitting?

- Try more complex model
  - More powerful model with a larger number of parameters
  - More layers
  - More neurons per layer
- Try larger quantity of features
  - Get additional features
  - Feature engineering
- Data cleaning, cross validation (hold-out, K-fold, LOOCV)

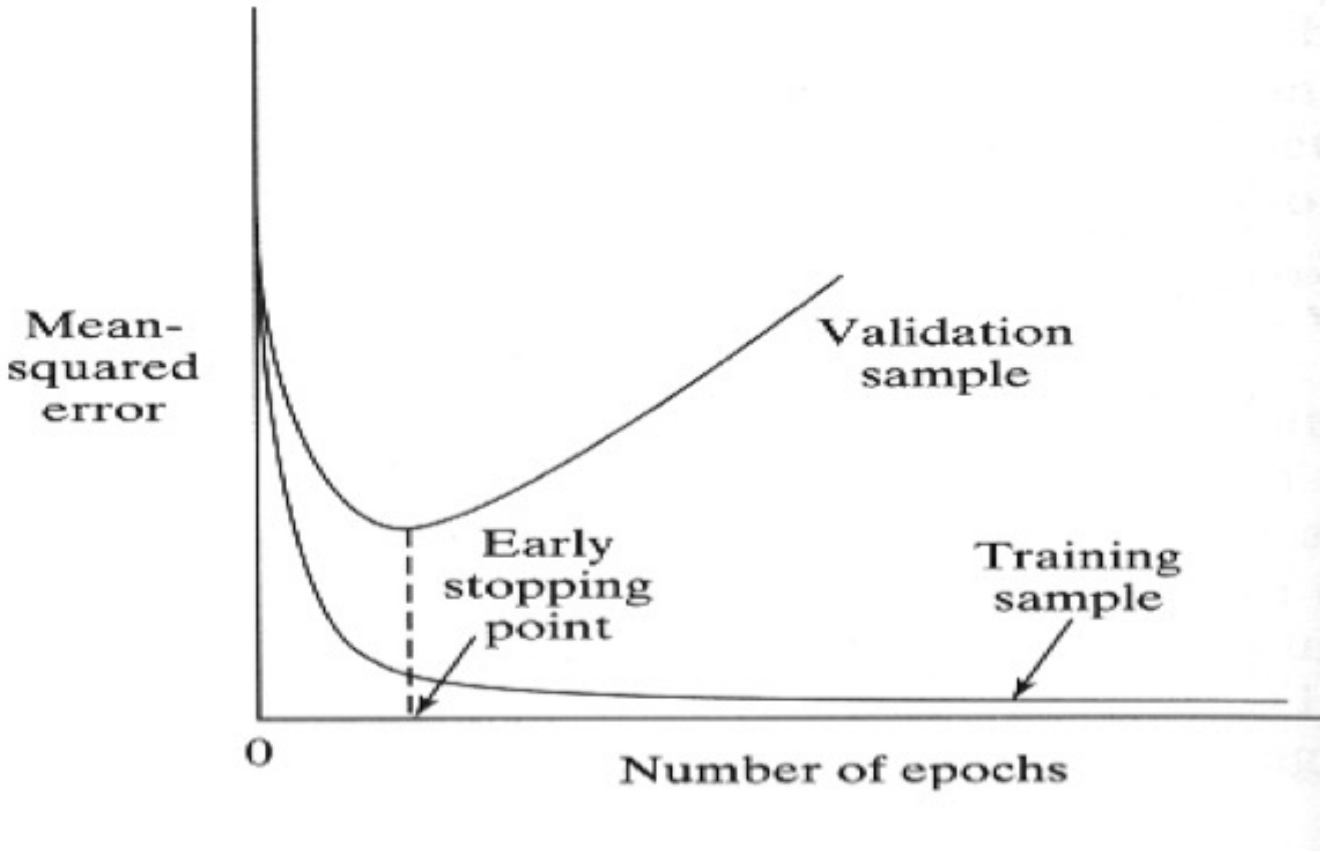
How to avoid overfitting?

- Try more simple model
  - Less powerful model with a fewer number of parameters
  - Less layers, less neurons per layer
- Try a smaller quantity of features
  - Remove additional features
  - Feature selection

How to avoid overfitting? (cont)

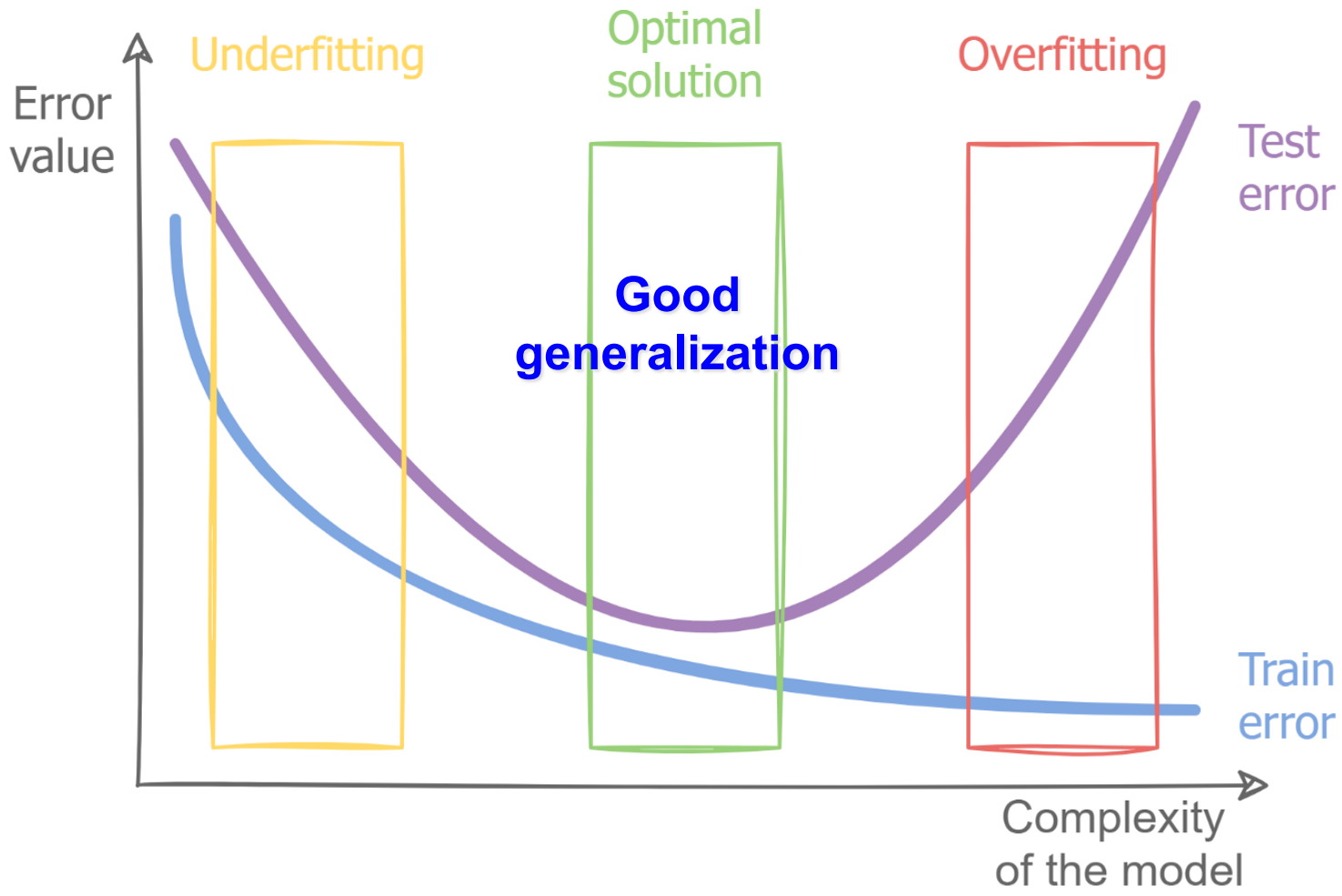
- Enlarge data
  - Data cleaning
  - Cross validation (hold-out, K-fold, LOOCV)
  - Data augmentation (rotate, flip, scale,…)
- More regularization
  - Early stopping
  - Drop out
  - L1, L2 regularization

Early stopping



Generalization

- **Good generalization:** the I/O mapping is nearly correct for new data



Generalization

- **Factors that influence generalization:**
  - Training set size
  - ANN architecture
  - Problem complexity
- **How to improve the generalization?**
  - Collect more data for training
  - Train several networks then select the best one
  - Avoid overfitting, avoid underfitting

Artificial Neural Network (ANN)

- **Outline:**

1. Overview of ANN
2. Components of ANN
3. ANN training (forward and backward propagation)
4. ANN characteristics
5. **ANN design**

ANN design process

- Data collection and representation
- Setup network topology
- Create network parameters
- Initialize weight and bias values
- Training
- Validation → re-design or using

ANN design process

- Data collection and representation
- Setup network topology
- Create network parameters
- Initialize weight and bias values
- Training
- Validation → re-design or using

Data representation

$$d_{k,j} = \begin{cases} 1, x_j \in C_k \\ 0, x_j \notin C_k \end{cases}$$

One-hot encoding

$$\begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \leftarrow k^{\text{th}} \text{ element}$$

$C_k - class\ k$

$x_j - input\ \ j$

$d_{k,j} - desired\ output$

ANN design process

- Data collection and representation
- Setup network topology
- Create network parameters
- Initialize weight and bias values
- Training
- Validation → re-design or using

Network topology

- **The way to connect neurons to form a network**
- **Topology consists of:**
  - **Neural framework:** described by the number of neuron layers, the number of neurons per layer
  - **Interconnection structure:** different kinds of connections such as interlayer connection, intralayer connection, self connection, sublayer connection

Types of ANN structure

- Feed forward neural network: may or may not have the hidden layers (one or multiple hidden layers)
- Radial basis function neural network
- Self organizing neural network
- Recurrent neural network
- Convolutional neural network
- Modular neural network

ANN design process

- Data collection and representation
- Setup network topology
- Create network parameters
- Initialize weight and bias values
- Training
- Validation → re-design or using

Network parameters

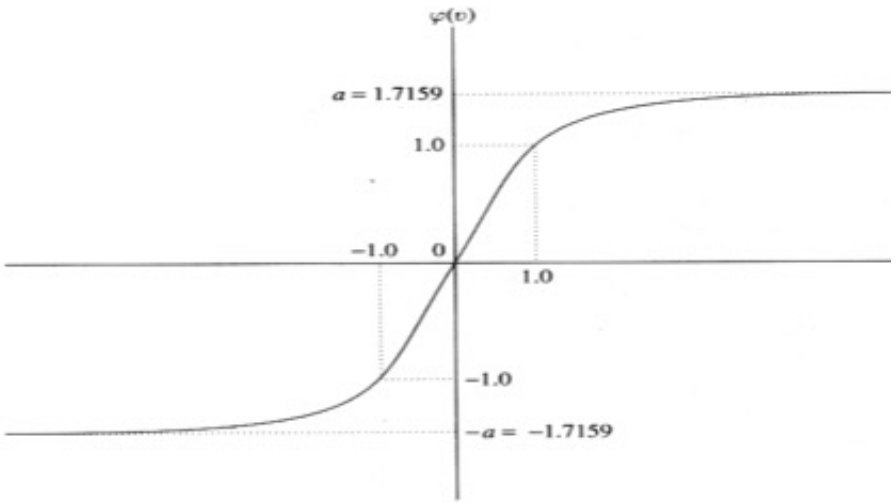
- Learning rate
- Activation function
- Net function
- Data preprocessing
- Number of examples in the training data set

Heuristic 1

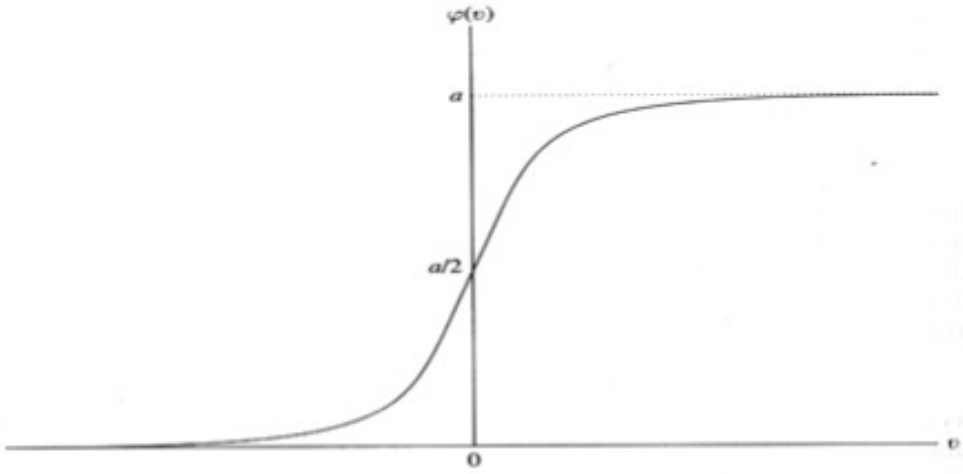
- **Maximization of information content:** every training example presented to the backpropagation algorithm must maximize the information content.
  - Use of an example that results in the largest training error.
  - Use of an example that is radically different from all those previously used.

Heuristic 2

- **Activation function:** network learns faster with antisymmetric functions when compared to nonsymmetric functions.



Antisymmetric function



Nonsymmetric function

Heuristic 3

- **How many training data?**

  Rule of thumb: the number of training examples should be at least five to ten times the number of weights of the network

ANN design process

- Data collection and representation
- Setup network topology
- Create network parameters
- Initialize weight and bias values
- Training
- Validation → re-design or using

Initialization

- Initializing weights and biases before training process
- Heuristics:
  - Weights should be initialized randomly (except zero)
  - Biases should be initialized as zero

ANN design process

- Data collection and representation
- Setup network topology
- Create network parameters
- Initialize weight and bias values
- Training
- Validation → re-design or using

Learning modes

- **Online learning:** learning as the data comes in (one example at a time)
  - Sequential mode or stochastic mode
- **Offline learning:** learning over the entire dataset
  - Batch mode: updating parameters after consuming the whole batch

ANN sequential training mode

- Presenting I/O-1 as x(1)-y(1)
- Performing a sequence of forward and backward computations
- Updating the weights
- Same for x(2)-y(2),… , x(N)-y(N)
- The learning process continues on an epoch-by-epoch basis until the stopping condition is satisfied

ANN design process

- Data collection and representation
- Setup network topology
- Create network parameters
- Initialize weight and bias values
- Training
- Validation → re-design or using

Method

- Hold out
- K-fold cross validation
- LOOCV

Performance

- Confusion matrix → Precision, recall, accuracy, F1-score, ROC, AUC, IoU,…
- Algorithm complexity, cost,…

What is meant by deep learning?

Deep learning is a subset of machine learning. It uses artificial neural networks with many hidden layers to extract features from raw data. The more input data, the more the model learns.

MACHINE LEARNING

INPUT

FEATURE EXTRACTION

CLASSIFICATION

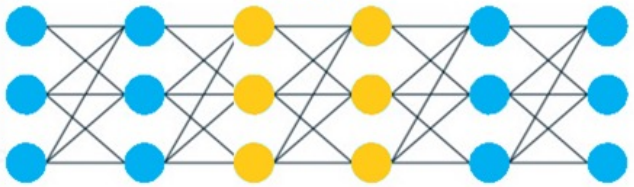
CAR
NOT CAR

OUTPUT

DEEP LEARNING

INPUT

HIDDEN
NEURONS



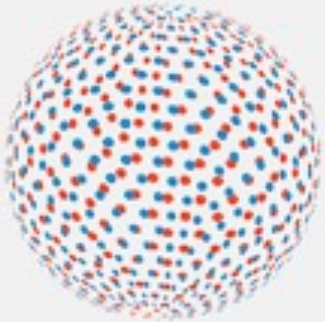
FEATURE EXTRACTION + CLASSIFICATION

CAR
Color: Red
Make: Ford
Model: Mustang
NOT CAR

OUTPUT

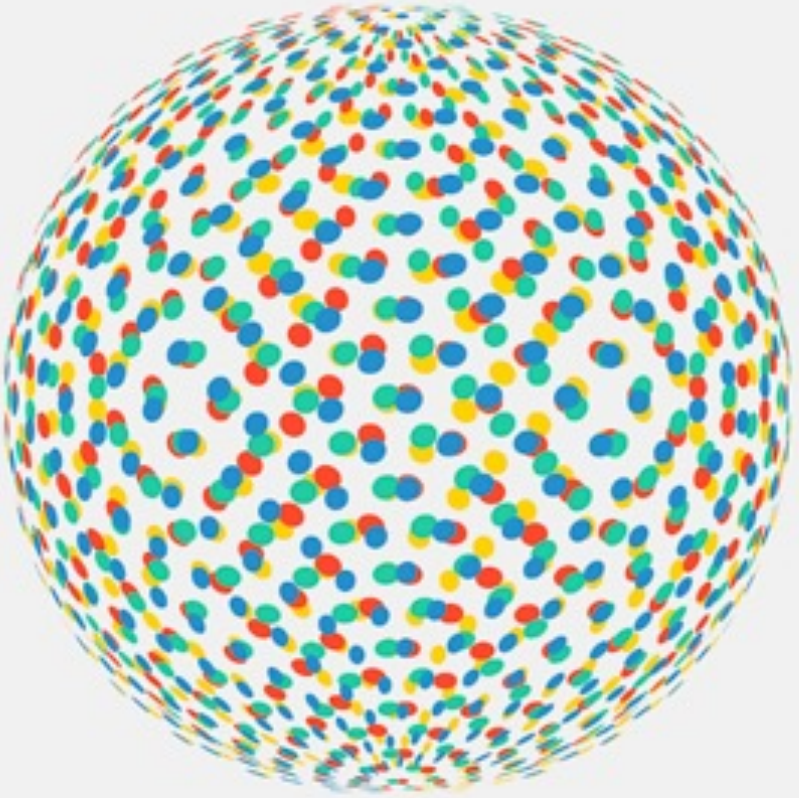
DLT LABS

Machine learning



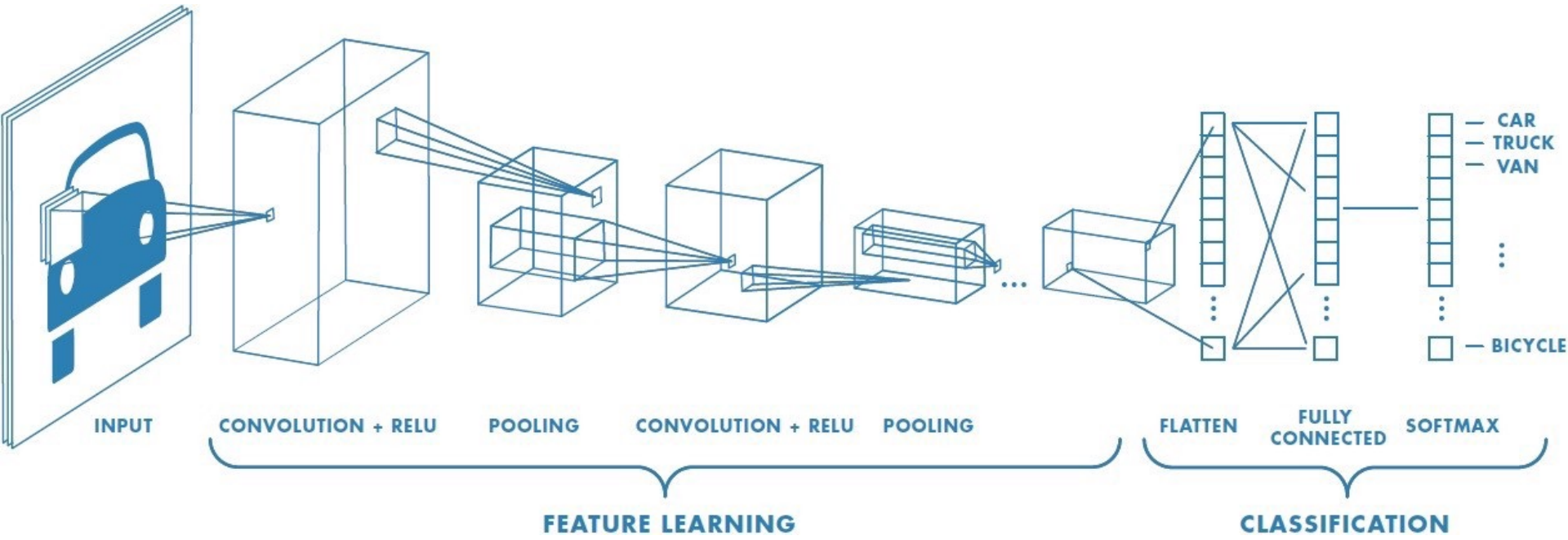
Thousands of data

Deep learning

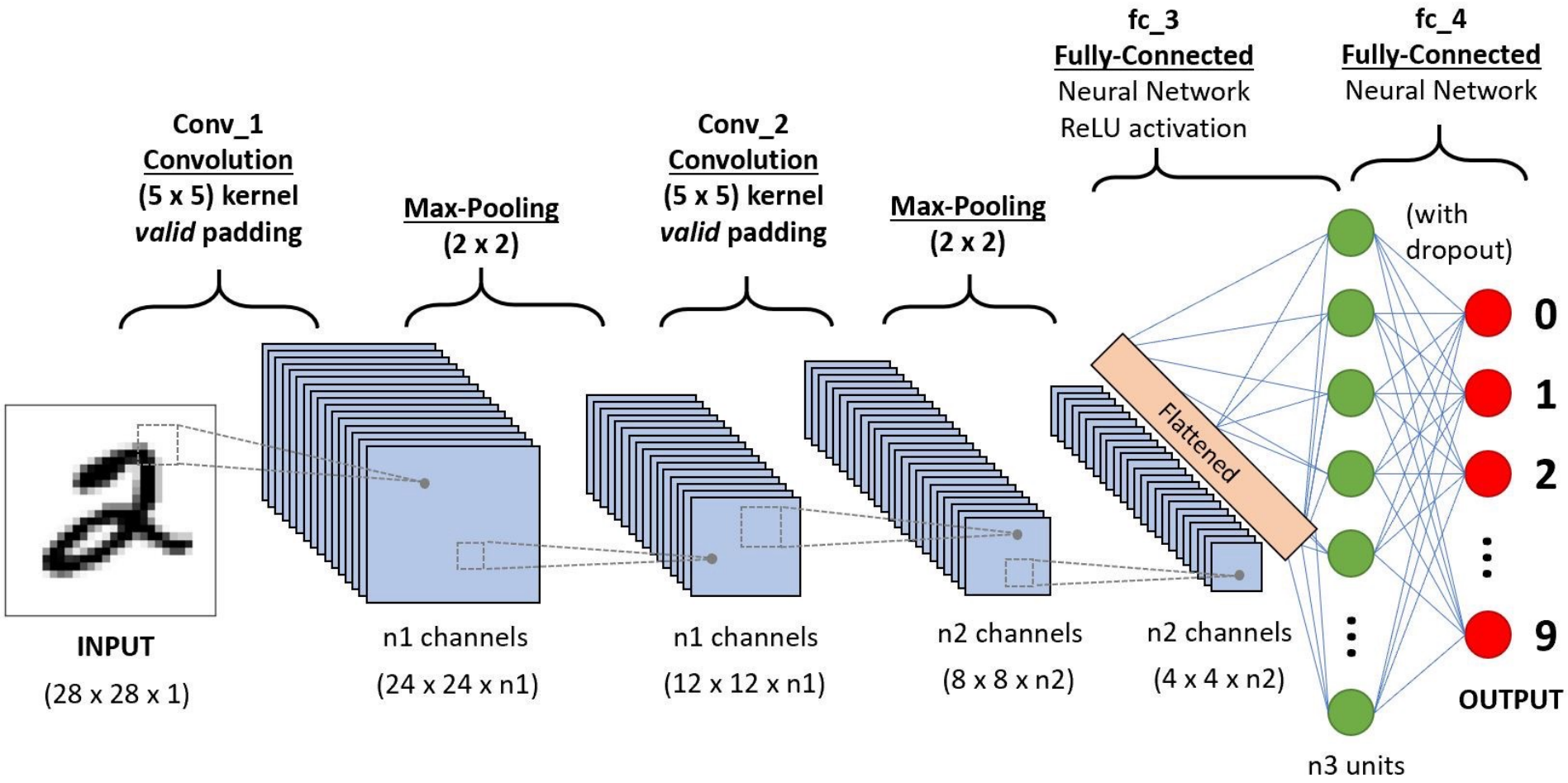


Millions of data

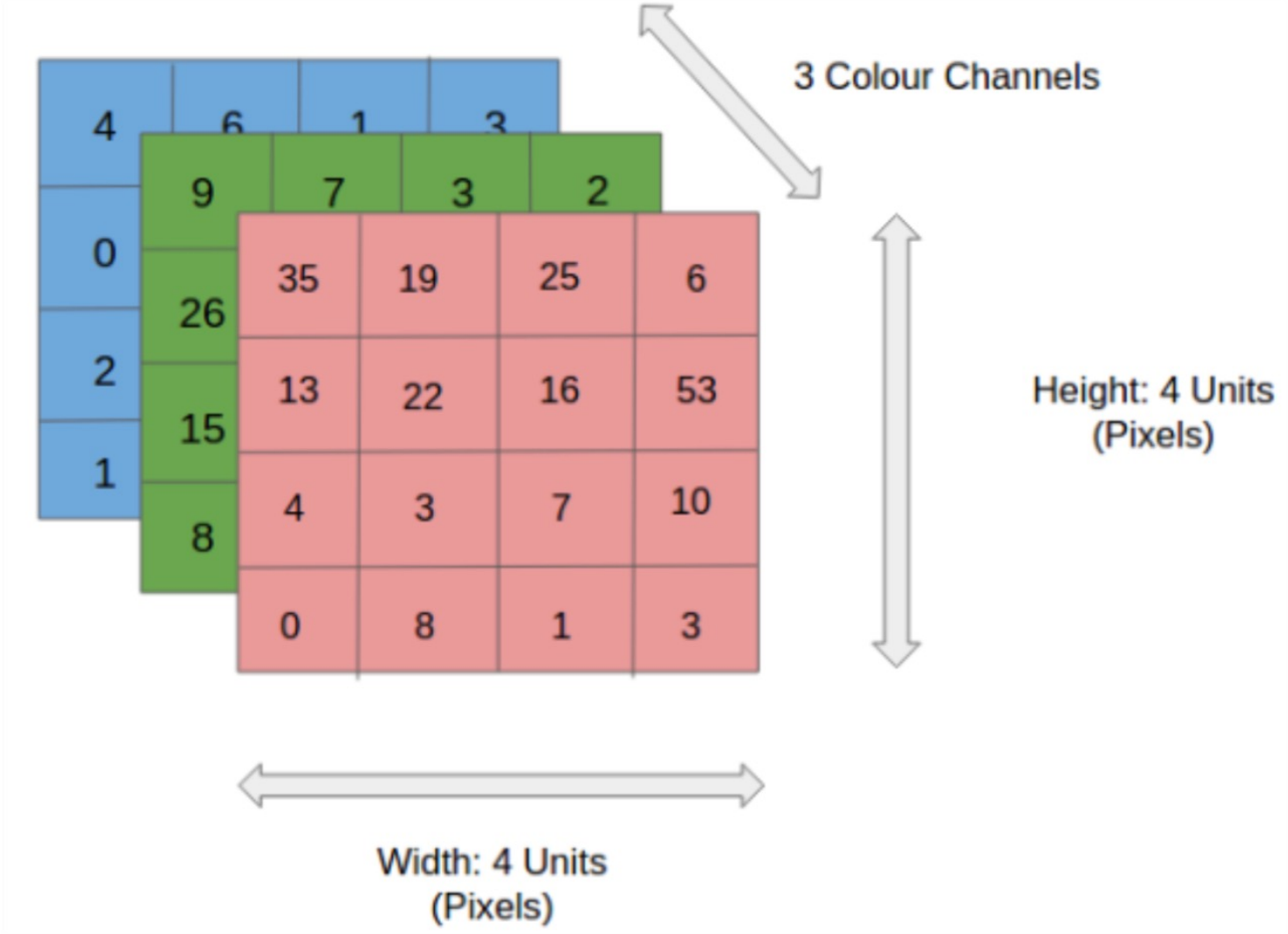
Convolutional neural network (CNN)



CNN to classify handwritten digits

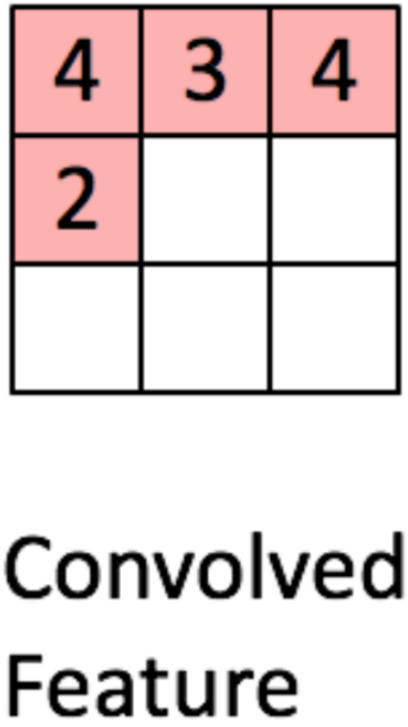


Input Image



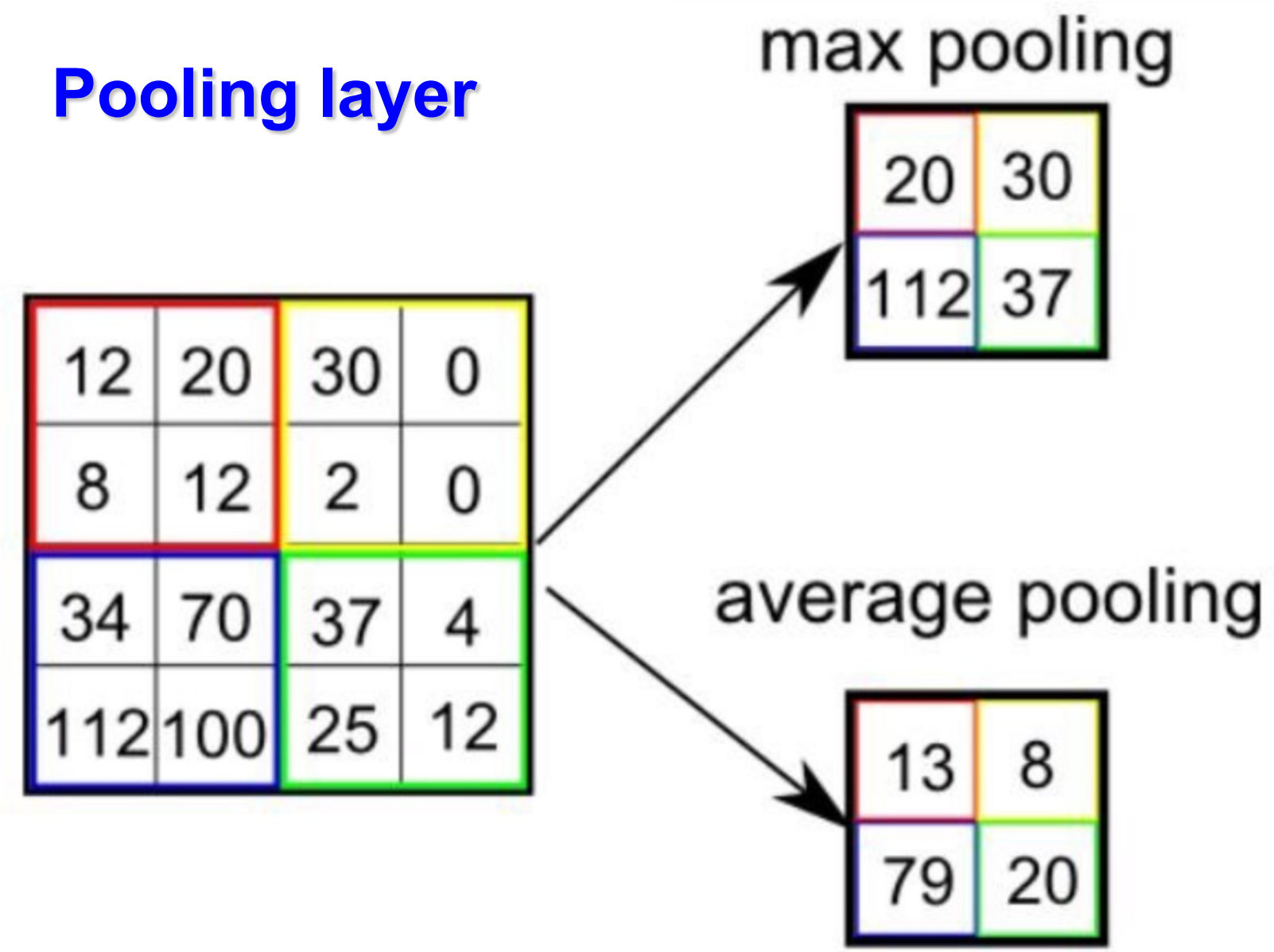
4×4×3 RGB Image

Convolution layer



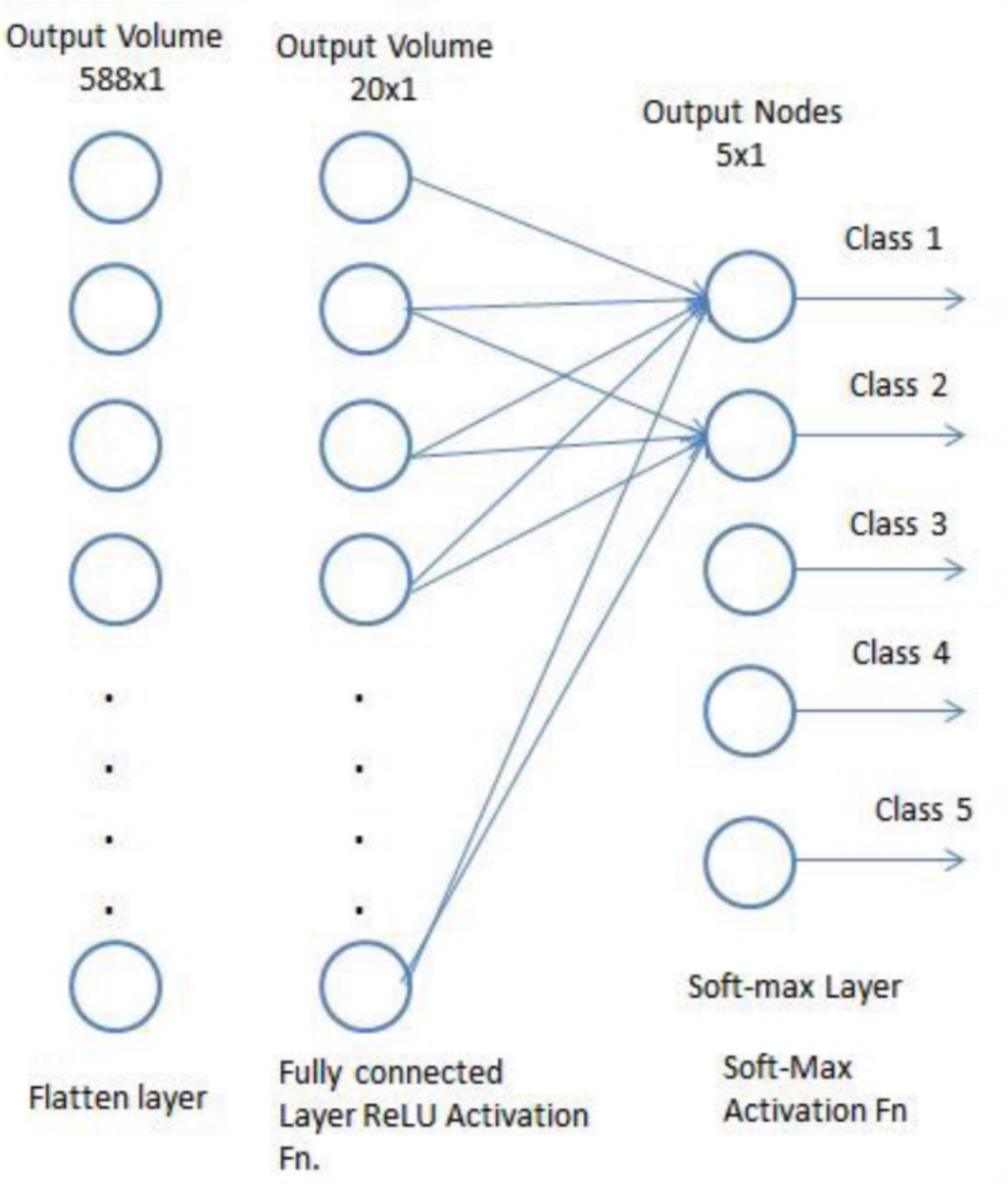
Convoluting a 5×5×1 image with a 3×3×1 kernel to get a 3×3×1 convolved feature

Pooling layer

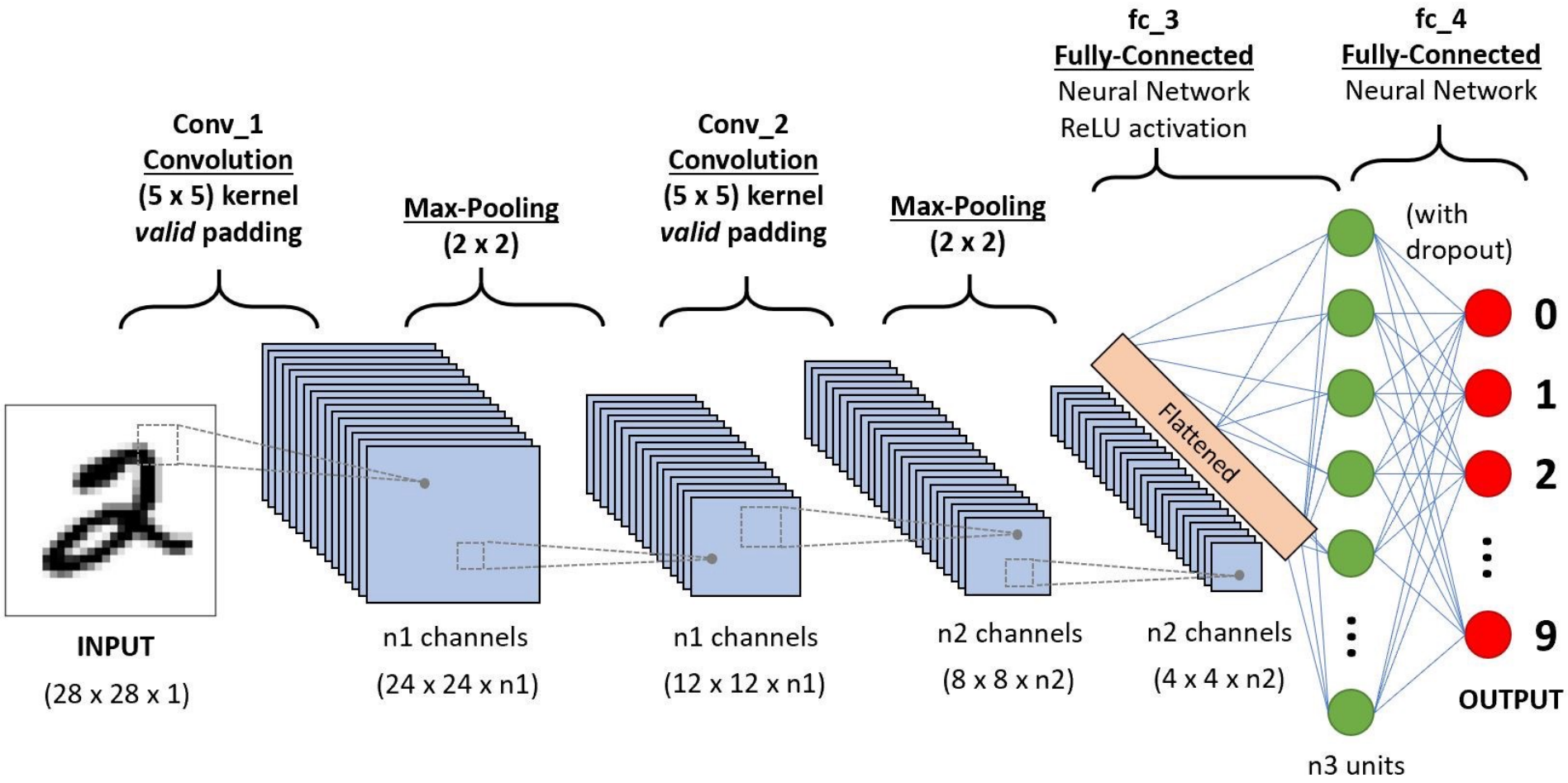


Types of Pooling

Classification



CNN to classify handwritten digits



Layer (type)	Output Shape	Param #
cv0 (Conv2D)	(None, 128, 128, 16)	448
max_pooling2d_11 (MaxPooling)	(None, 64, 64, 16)	0
cv1 (Conv2D)	(None, 64, 64, 32)	4640
max_pooling2d_12 (MaxPooling)	(None, 32, 32, 32)	0
cv2 (Conv2D)	(None, 32, 32, 64)	18496
max_pooling2d_13 (MaxPooling	(None, 16, 16, 64)	0
cv3 (Conv2D)	(None, 16, 16, 128)	73856
max_pooling2d_14 (MaxPooling	(None, 8, 8, 128)	0
cv4 (Conv2D)	(None, 8, 8, 256)	295168
max_pooling2d_15 (MaxPooling	(None, 4, 4, 256)	0
cv5 (Conv2D)	(None, 4, 4, 128)	32896
cv6 (Conv2D)	(None, 4, 4, 64)	8256
cv7 (Conv2D)	(None, 4, 4, 32)	2080
flatten_3 (Flatten)	(None, 512)	0
hiddenlayer1 (Dense)	(None, 512)	262656
hiddenlayer2 (Dense)	(None, 128)	65664
dense_3 (Dense)	(None, 51)	6579
activation_3 (Activation)	(None, 51)	0

Total params: 770,739

Deep learning crash course for beginners

https://www.youtube.com/watch?v=VyWAvY2CF9c

Course developed by Jason Dsouza

Duration: 1hr 30 minutes

Bài tập áp dụng 1

- Sinh viên làm bài tập theo nhóm đã phân công cho học phần, bao gồm các bước sau:
- Sử dụng dữ liệu lá cây tự sưu tầm (3 + 2) hoặc (2+2)
- Huấn luyện và kiểm tra mô hình ANN (số neuron lớp ẩn lần lượt là 10, 15), dùng đặc trưng Hu's moments, tốc độ học là $\eta = 0.1$. Đánh giá mô hình ANN bằng phương pháp 5-fold cross validation. Nhận xét kết quả.
- Viết báo cáo.

Bài tập áp dụng 2

- Sinh viên làm bài tập theo nhóm đã phân công cho học phần, bao gồm các bước sau:
- Sử dụng dữ liệu lá cây tự sưu tầm (3 + 2) hoặc (2+2)
- Huấn luyện và kiểm tra mô hình ANN (số neuron lớp ẩn lần lượt là 10, 15), dùng đặc trưng HOG với các tham số tự chọn. Đánh giá mô hình ANN bằng phương pháp 5-fold cross validation. Nhận xét kết quả.
- Viết báo cáo.

Ví dụ: Green vegetables

Diếp cá (fish herb)

Rau má (cica)

Lá lốt (piper lolot)

Lá bạc hà (mint)

Lá ngò (cilantro)