Group 9

CLASSIFICATION WITH NEURAL NETWORK

October 2024

Lecturer: DO THANH THAI

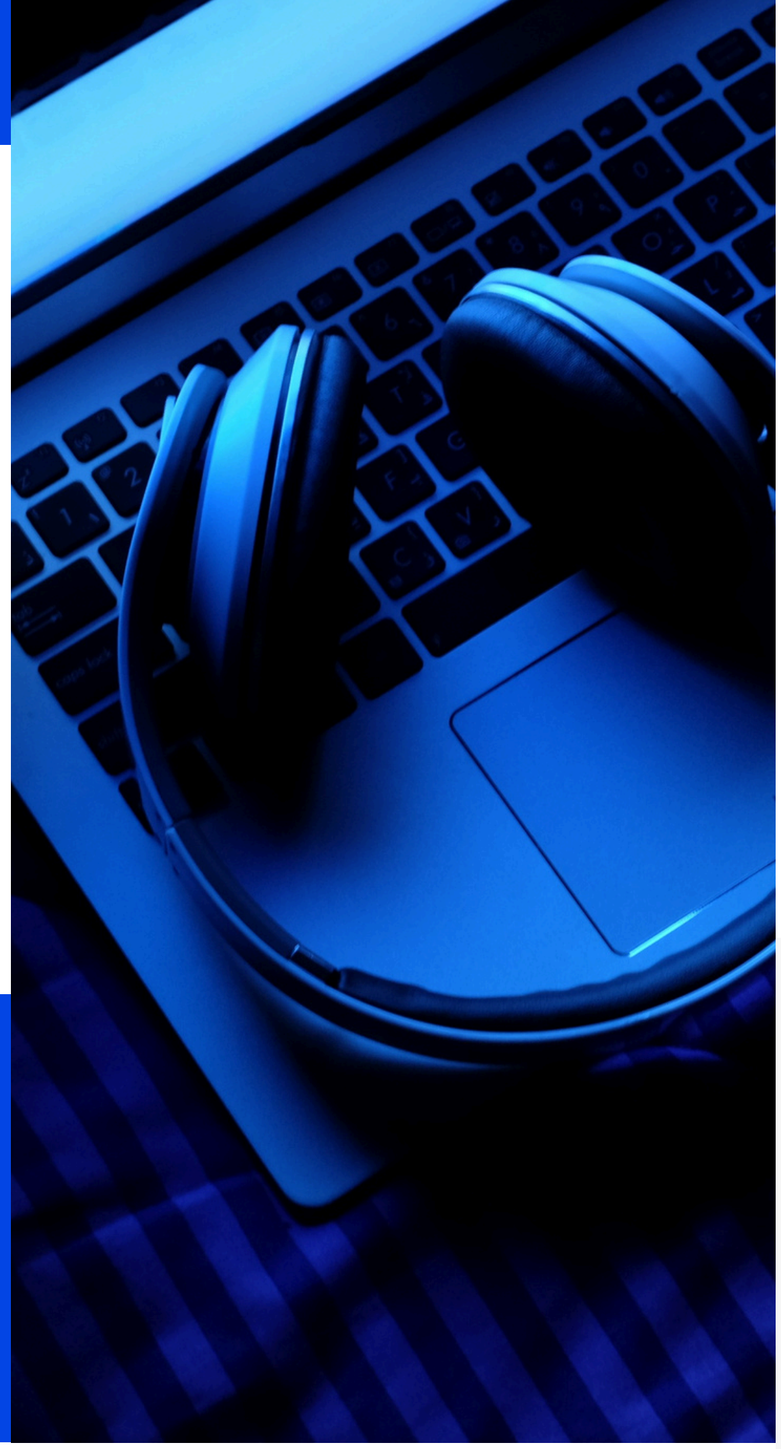
GROUP MEMBER

2110102 - Vo Van Dung

2112256 - Nguyen Thai Tan

2220035 - Tran Thi Lai

Group 9

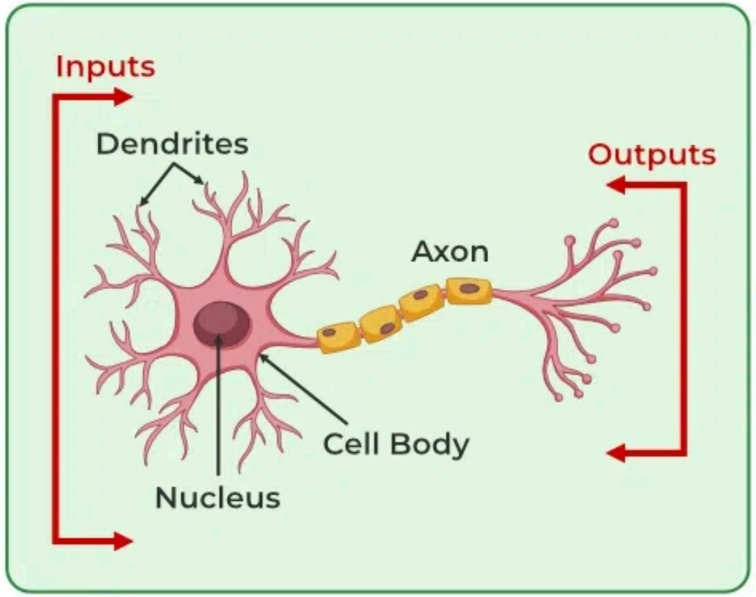


CONTENT

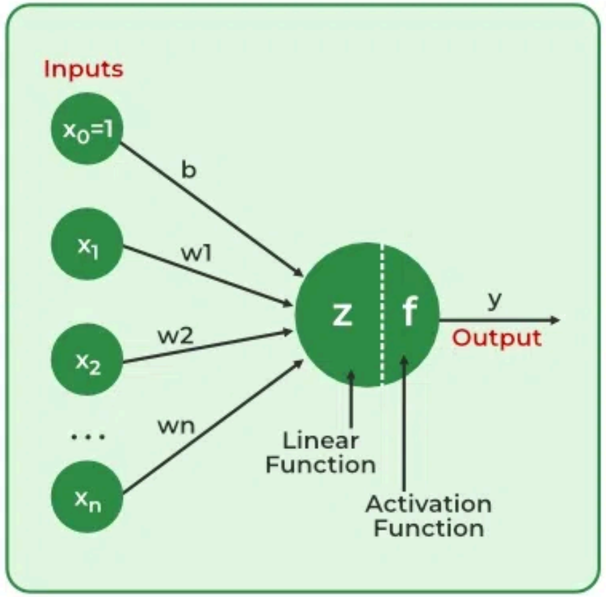
INTRODUCTION NEURAL NETWORK

Group 9

Biological neural network

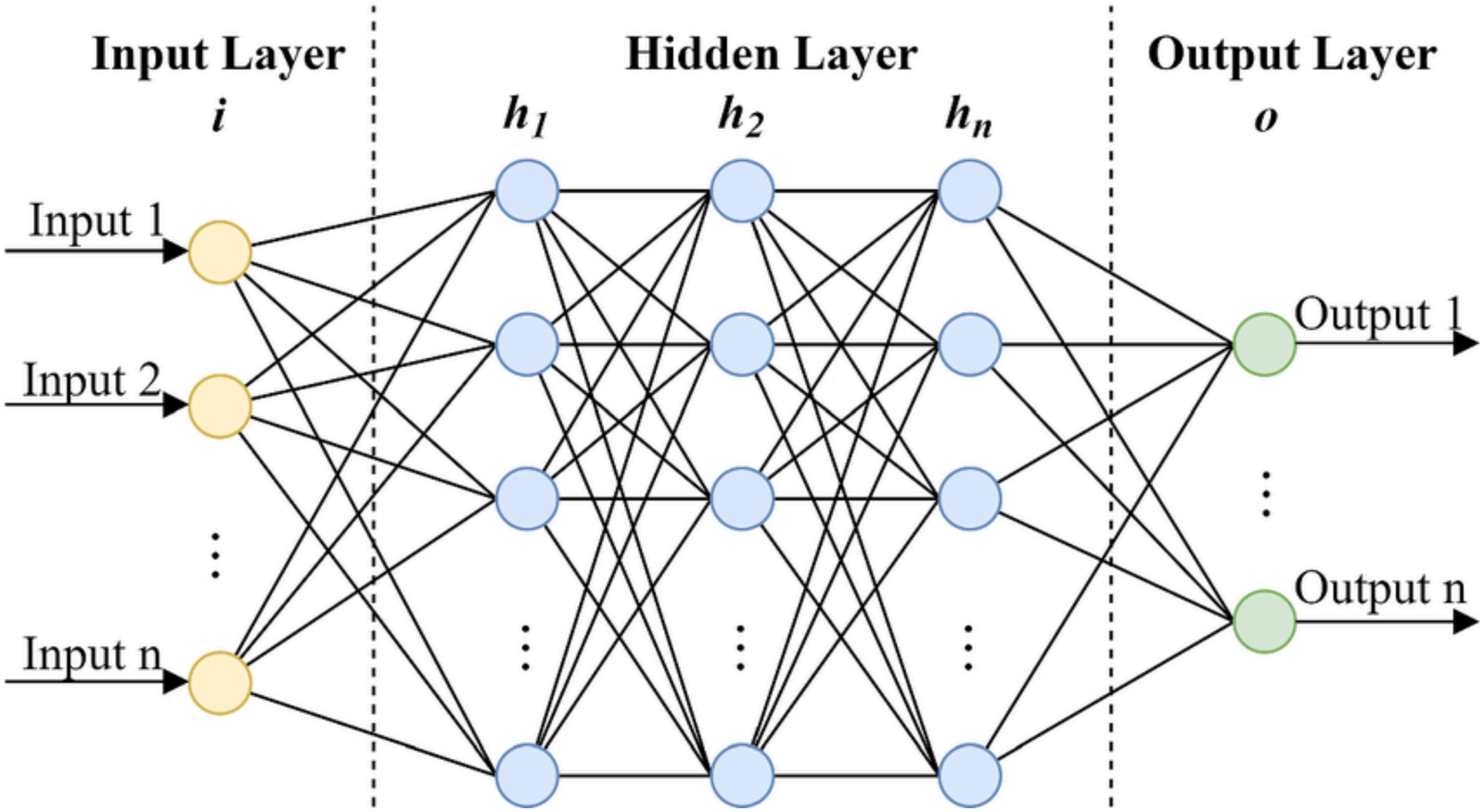


Artificial neural network



GENERAL MODEL OF NEURAL NETWORK

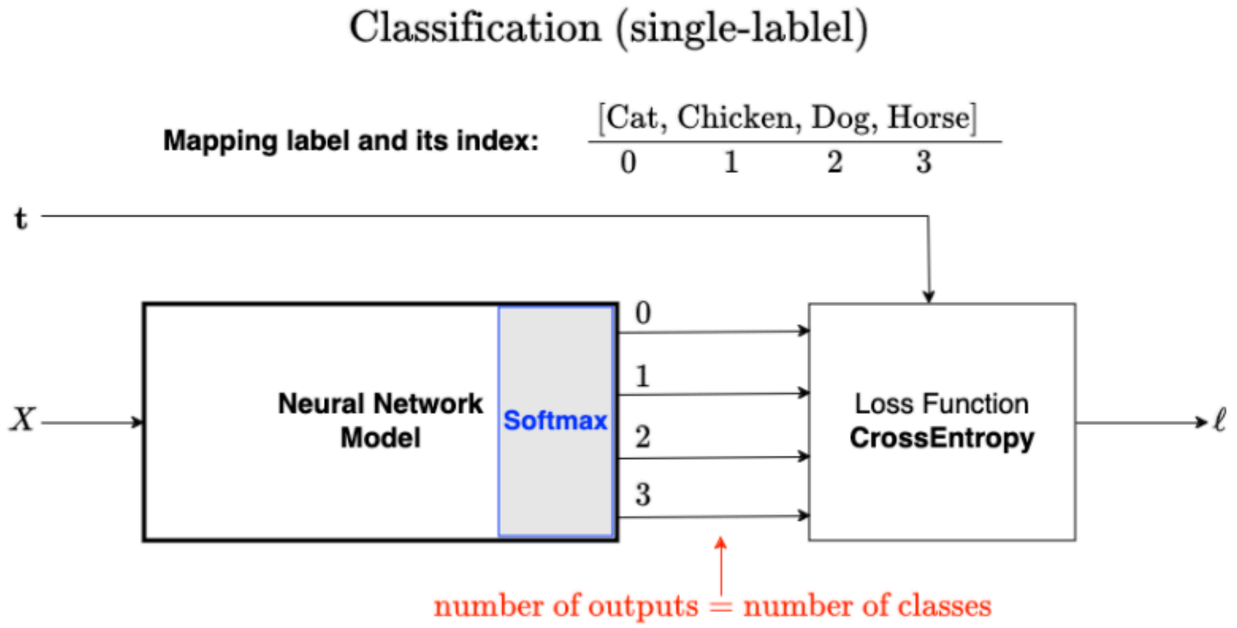
Group 9

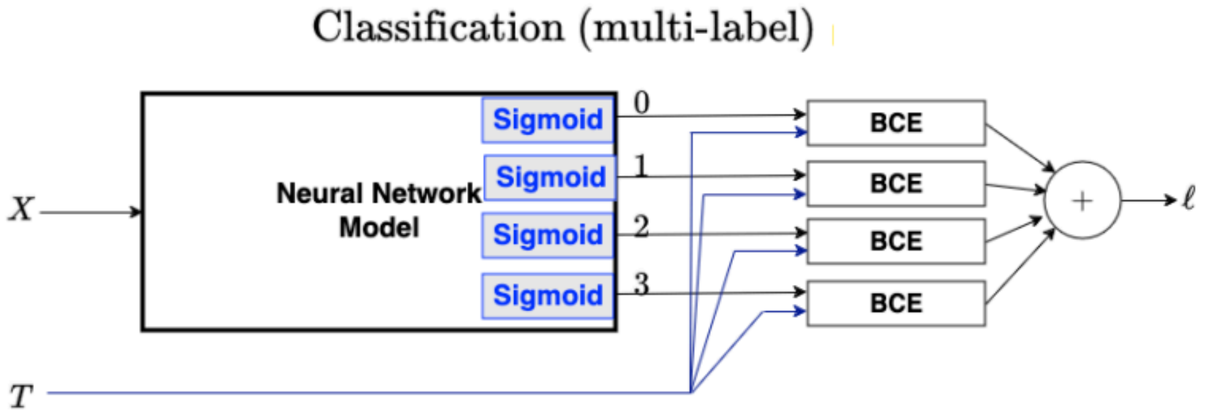


Total of layer = number layer - 1 (not couting input layer)

CLASSIFICATION WITH NEURAL NETWORK

Group 9





ADVANTAGES OF ANN COMPARED TO OTHER CLASSIFICATION METHODS

- **Logistic Regression**: Poor performance when classifying non-linear data.

➡ ANN can learn complex non-linear relationships, whereas logistic regression is limited to linear relationships between input and output.

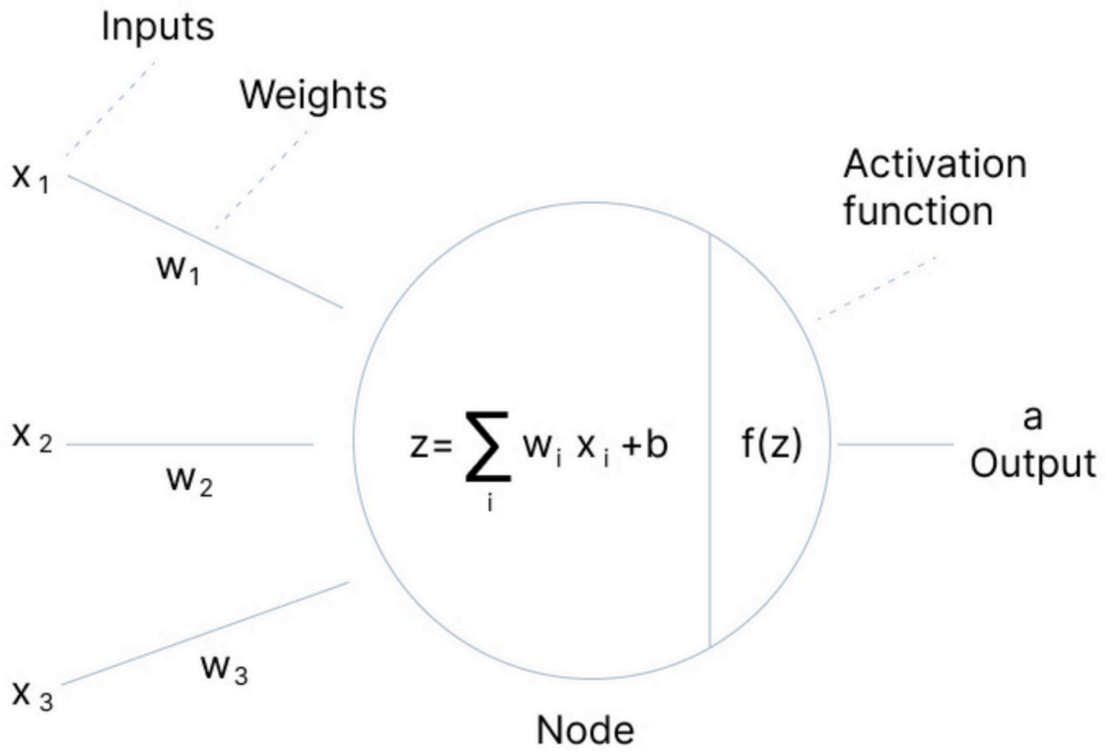
- **Decision Tree**: Prone to overfitting, especially with deep trees.

➡ ANN has better generalization for complex data and is less prone to overfitting if properly tuned, though harder to interpret than decision trees.

- **Bayesian Method**: Requires assumptions of independence between variables, which is often unrealistic.

➡ ANN can learn complex relationships between features without assuming independence, but it requires more computational resources and training time.

ACTIVATION FUNCTION



ACTIVATION FUNCTION

Activation Function introduces non-linearity to the neural network, allowing it to learn from the data and model complex patterns

important role in the hyperparameters of AI-based models

WHY NEURAL NETWORKS NEED ACTIVATION FUNCTIONS?

- Non-linearity: Without activation functions, a neural network would simply be a linear regression model
- Gradient: Activation functions and their derivatives (used in backpropagation) provide the necessary gradients that guide the weight updates during training.
- Thresholding: Activation functions introduce a threshold, ensuring that neurons are only activated when necessary.

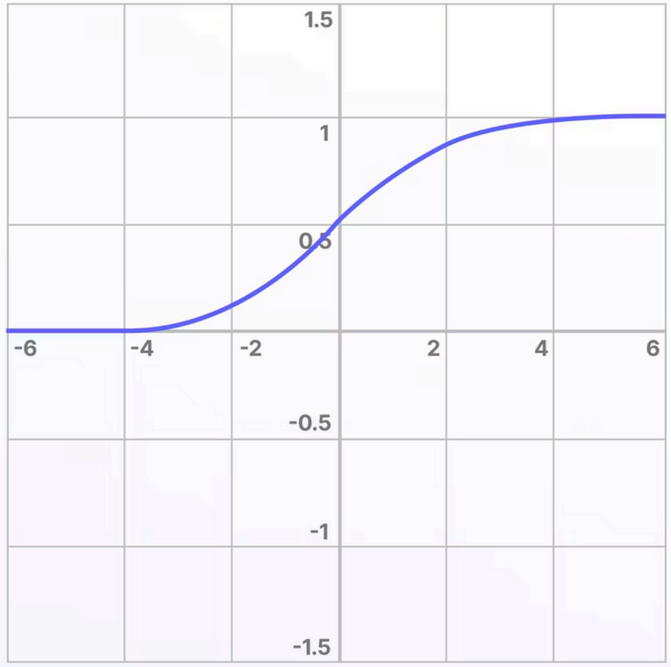
COMMON ACTIVATION FUNCTIONS

Group 9

Sigmoid, Logistic Activation Functions

The sigmoid function has an "S"-shaped curve.
It is a continuous, differentiable function, and it is bounded within the range (0, 1)

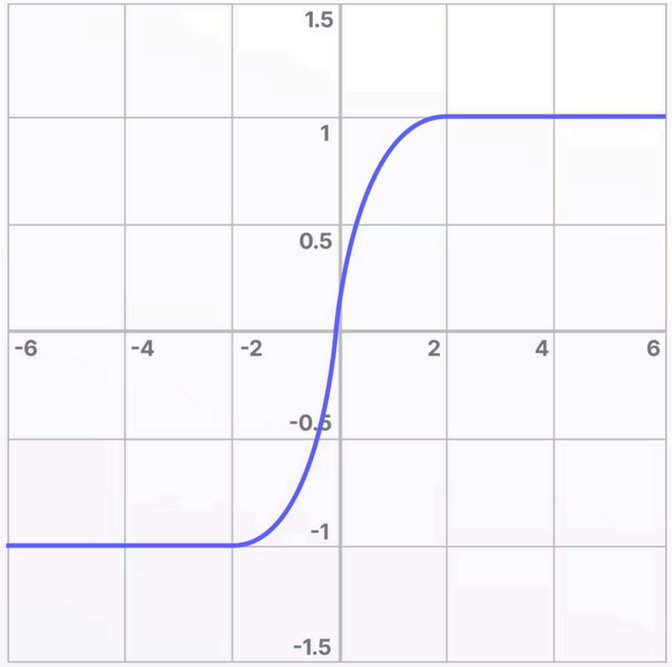
$$a_j^i = o(z_j^i) = \frac{1}{1 + \exp(-z_j^i)}$$



Tanh Function (Hyperbolic Tangent)

similar to sigmoid/logistic, with the S shape curve
the output range is -1 to 1
outputs of tanh are zero-centric, the values can be more easily mapped on a scale between strongly negative, neutral, or positive

$$a_j^i = o(z_j^i) = \tanh(z_j^i)$$

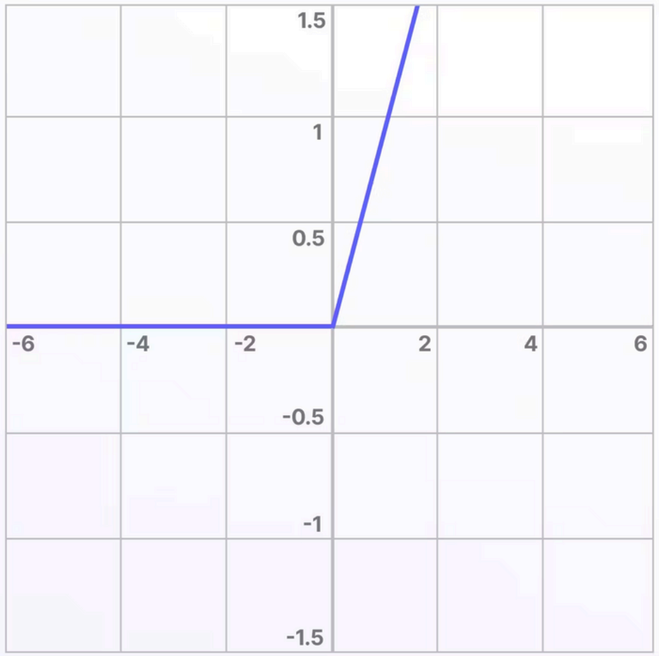


$$f(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$$

COMMON ACTIVATION FUNCTIONS

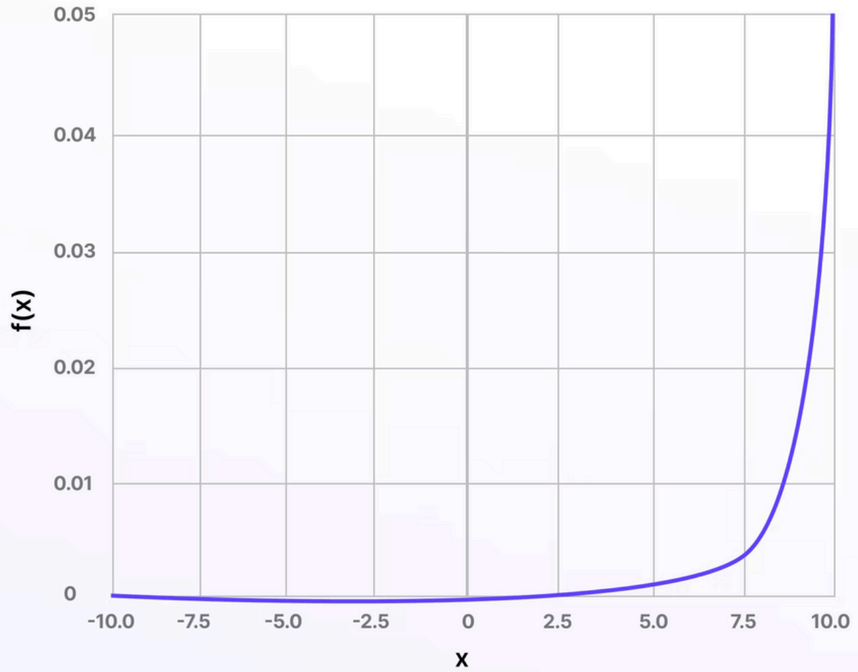
Rectified Linear Unit (ReLU)

$$a_j^i = o(z_j^i) = \max(0, z_j^i)$$



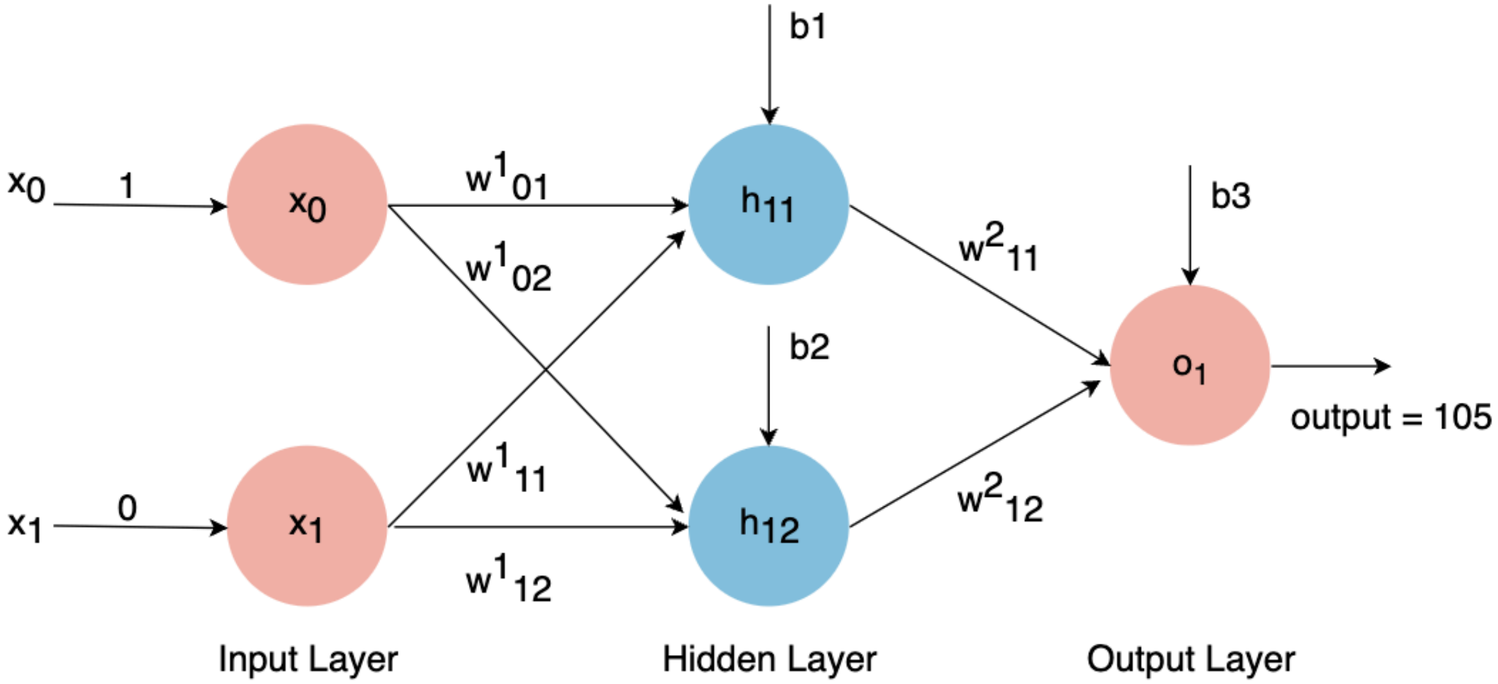
Softmax

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

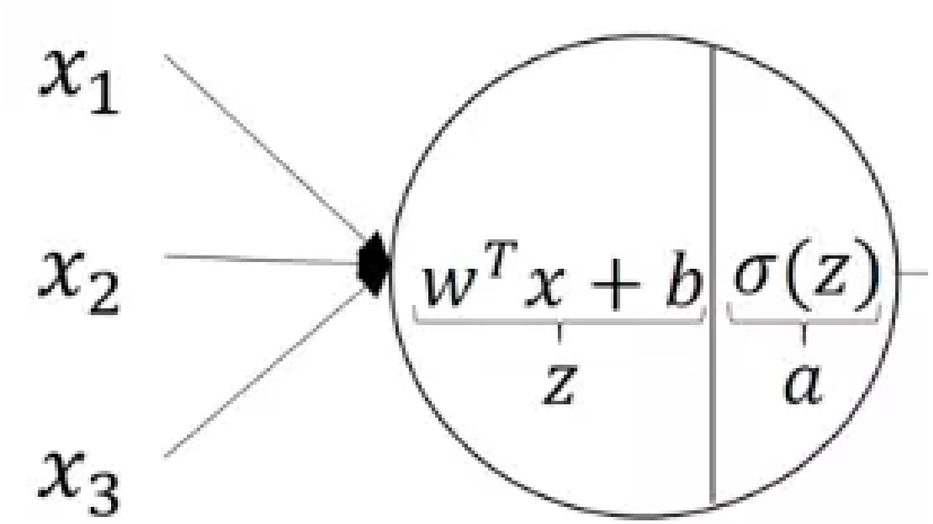


FORWARD PROPAGATION

Group 9



FEED-FORWARD NETWORK

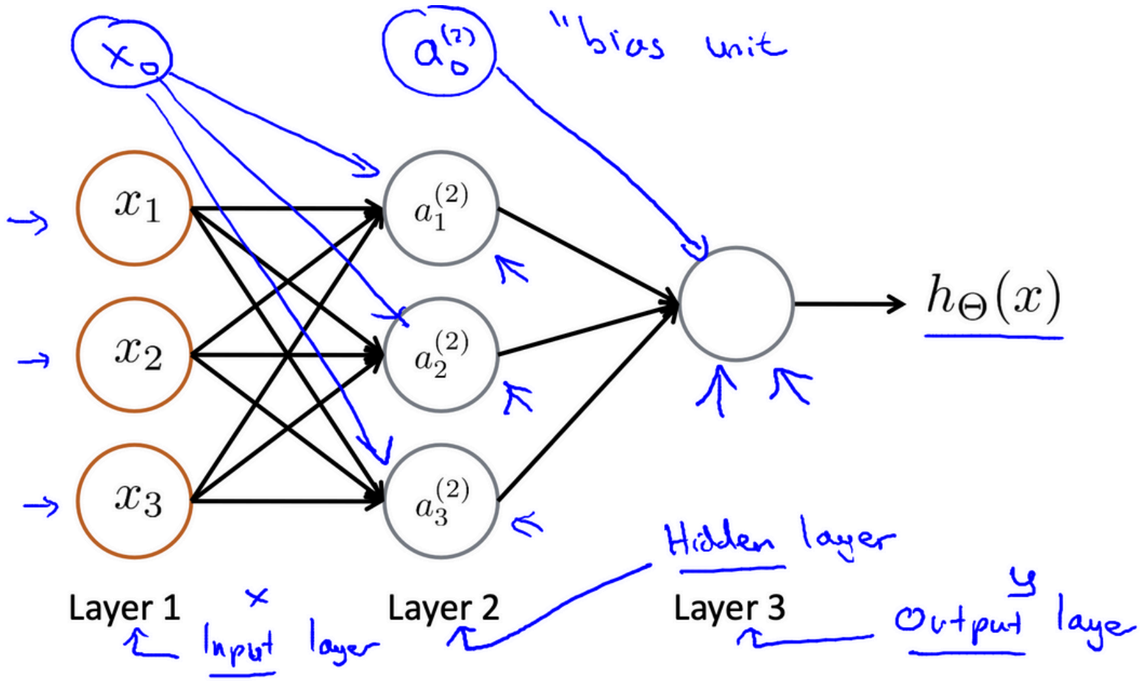


z: is pre-activation, a is activation

$$z = \sum_i w_i x_i + b = \mathbf{w}^\mathsf{T}\mathbf{x} + \mathrm{b}$$

$$a = \sigma(z)$$

CALCULATIONS IN HIDDEN LAYER



$a_i^{(j)}$: The activation of unit i in layer j.

$\Theta^{(j)}$: The matrix of weights controlling the mapping from layer j to $j+1$.

$\Theta^{(1)} \in \mathbb{R}^{3\times 4}$ corresponds to the weights from the input layer (with 3 units) to the hidden layer (with 4 units, including a bias).

$\Theta^{(2)}$ corresponds to the weights from the hidden layer to the output layer.

CALCULATIONS IN HIDDEN LAYER

Hidden layer activations

$$a_1^{(2)} = g(\Theta_{10}^{(1)} x_0 + \Theta_{11}^{(1)} x_1 + \Theta_{12}^{(1)} x_2 + \Theta_{13}^{(1)} x_3)$$

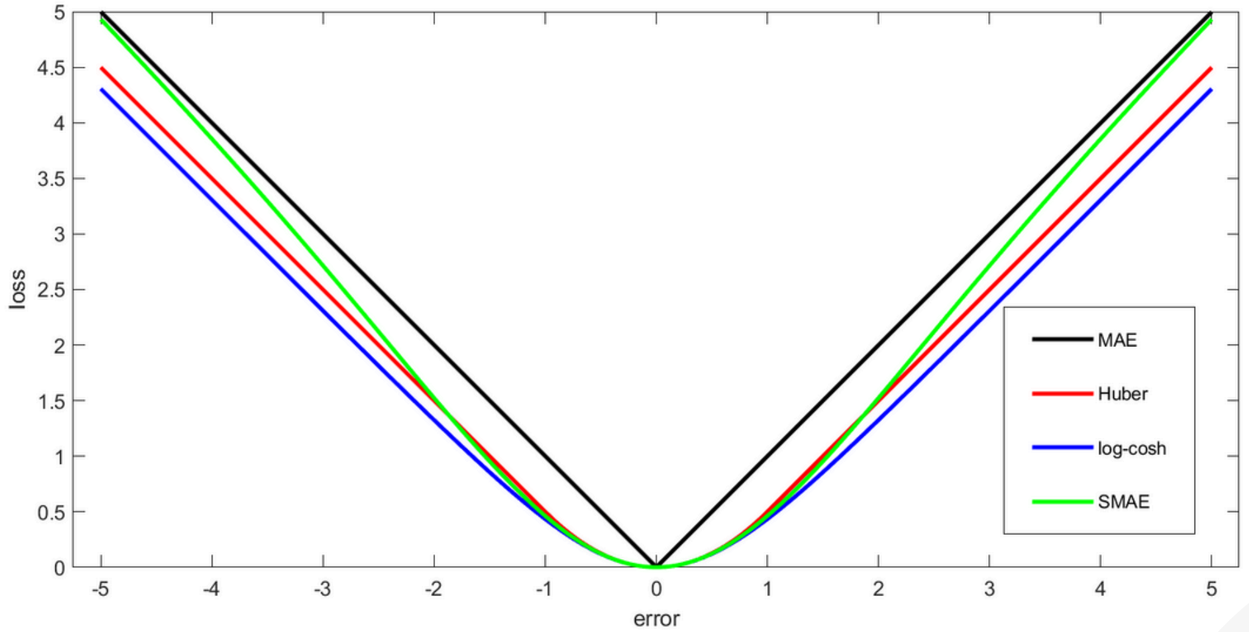
$$a_2^{(2)} = g(\Theta_{20}^{(1)} x_0 + \Theta_{21}^{(1)} x_1 + \Theta_{22}^{(1)} x_2 + \Theta_{23}^{(1)} x_3)$$

$$a_3^{(2)} = g(\Theta_{30}^{(1)} x_0 + \Theta_{31}^{(1)} x_1 + \Theta_{32}^{(1)} x_2 + \Theta_{33}^{(1)} x_3)$$

Output of the network

$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)})$$

LOSS FUNCTION



DEFINITION

- A loss function (or cost function) is a mathematical function used to evaluate how well a machine learning model performs.
- It quantifies the **difference** between the predicted output and the actual output (ground truth).
- In ANN: The loss function is essential during the training phase. It provides feedback to the model about its predictions, guiding the optimization process.

POPULAR LOSS FUNCTIONS

- *Difference* between the predicted output of the model and the actual output => **Mean Absolute Error (MAE)**

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- *Robust to Outliers:* MAE treats all errors equally, making it less sensitive to outliers.

- Useful in *regression* tasks

POPULAR LOSS FUNCTIONS

- MAE *not continous* => **Mean Squared Error (MSE)**

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- *Sensitive to Outliers:* Because errors are squared, larger errors have a disproportionately large impact on the MSE.

- Primarily used in *regression* problems

POPULAR LOSS FUNCTIONS

- **Cross-Entropy**

$$\mathrm{CE} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

- *Sensitivity:* Highly sensitive to incorrect class predictions; it heavily penalizes predictions that are confident but wrong.
- Used when dealing with *single-label classification*

POPULAR LOSS FUNCTIONS

- **Binary Cross-Entropy**

$$\mathrm{BCE} = -\left(y \log(\hat{y}) + (1-y) \log(1-\hat{y})\right)$$

- *Sensitivity:* It is sensitive to the distribution of classes; a heavily imbalanced dataset may require adjustments.
- Used when dealing with *two-class classification*

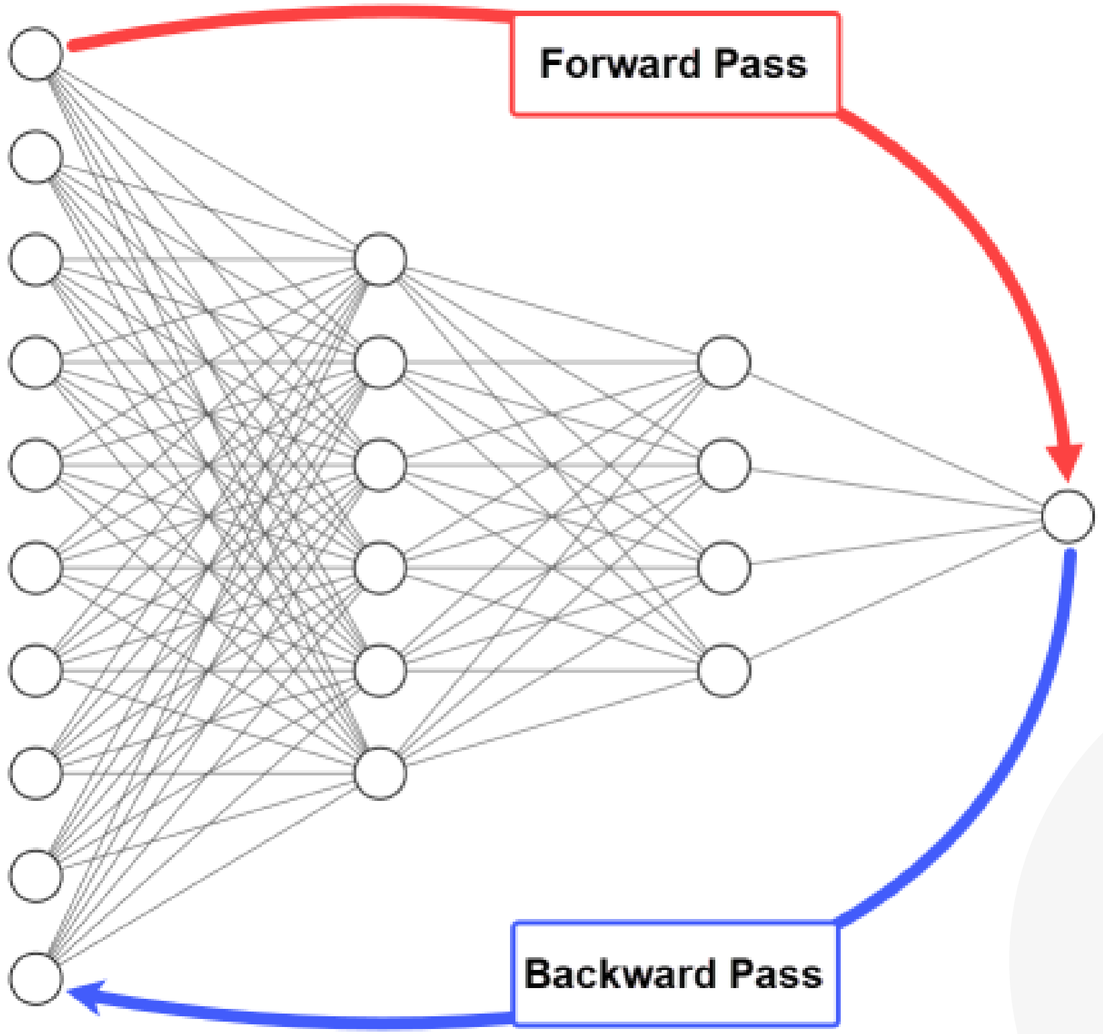
POPULAR LOSS FUNCTIONS

- **Categorical Cross-Entropy**

$$\text{CCE} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

- *Sensitivity:* Highly sensitive to incorrect class predictions; it heavily penalizes predictions that are confident but wrong.
- Used when dealing with *multi-label classification*

BACKWARD PROPAGATION



DEFINITION

- Backpropagation is a *gradient estimation method* commonly used for training neural networks to compute the *network parameter updates*.
- Minimizes the error of loss function by adjusting weights and biases.

$$\frac{\partial L}{\partial w_1}$$

CHAIN RULE

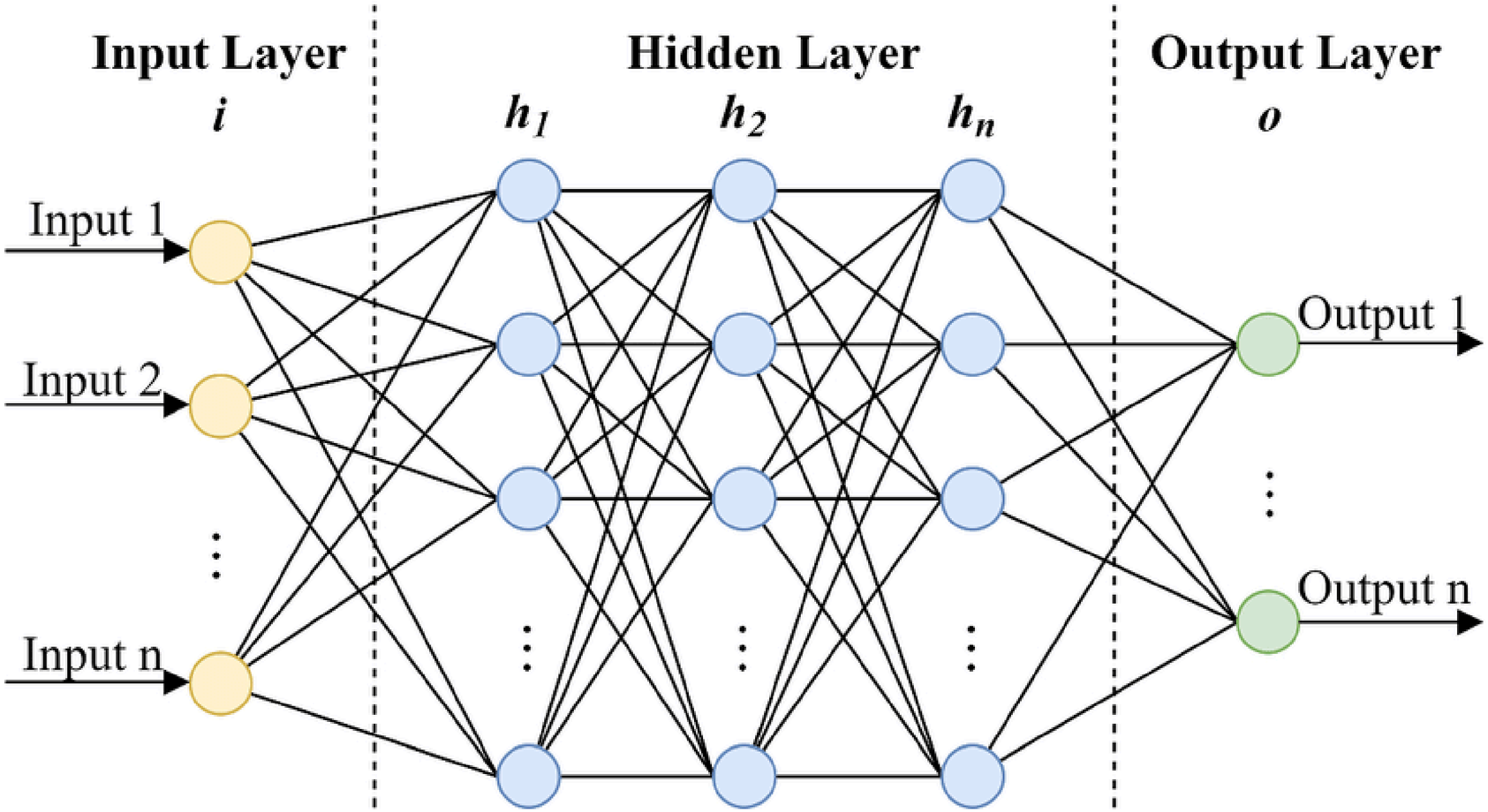
- The chain rule is a fundamental principle in calculus used to compute the derivative of a composite function.

  If we have a function ***z = f(y)*** and ***y=g(x)***, the chain rule can be expressed as:

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y} \cdot \frac{\partial y}{\partial x}$$

BACKPROPAGATION

Group 9



$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial h_n} \cdot \frac{\partial h_n}{\partial h_{n-1}} \cdot \ldots \cdot \frac{\partial h_2}{\partial h_1} \cdot \frac{\partial h_1}{\partial w_1}$$

UPDATE PARAMETERS

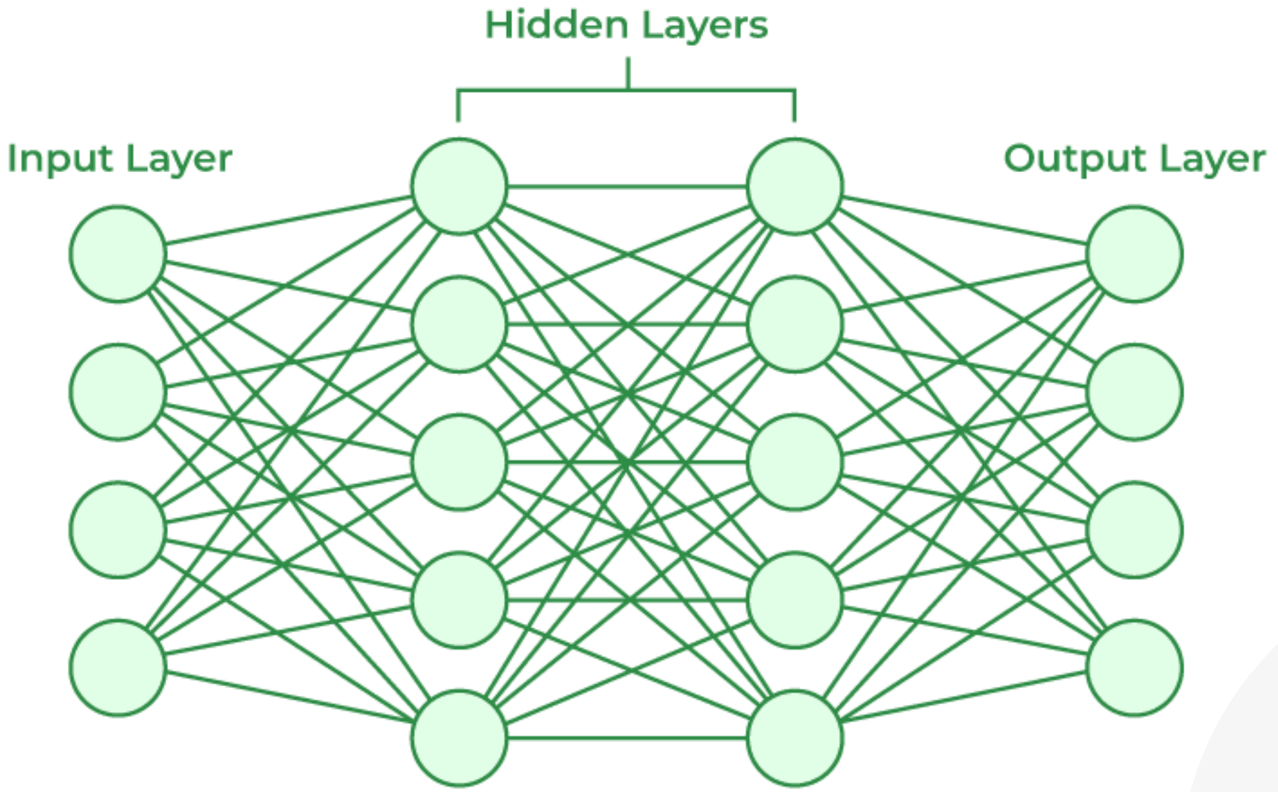
- Gradient Descent $$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w}$$

- **Learning Rate**: A hyperparameter that controls how much to change the weights during training.
  Small:
  - Stable convergence, less chance of overshooting.
  - Slow training process, may get stuck in local minima.

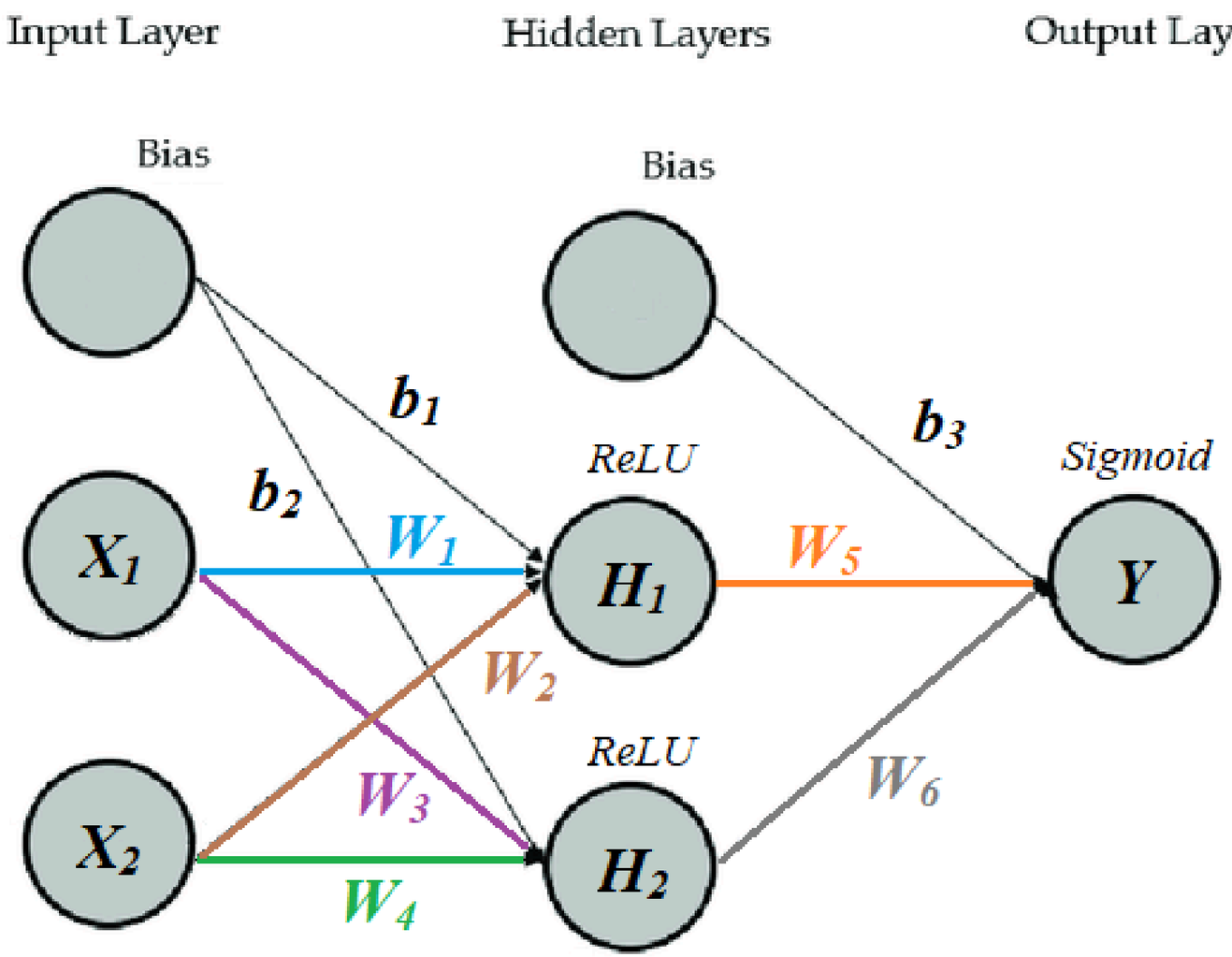
  Large:
  - Faster convergence.
  - Risk of overshooting the minimum

  Adaptive Learning Rates: adjust the learning rate during training (like Adam, RMSprop)

EXAMPLE



THE NETWORK



$X_1 = 0.5$
$X_2 = 0.8$
$W_1 = 0.4$
$W_2 = 0.3$
$W_3 = 0.2$
$W_4 = 0.9$
$b_1 = 0.1$
$b_2 = -0.3$
$W_5 = 0.7$
$W_6 = 0.6$
$b_3 = 0.2$

FORWARD

$$H_1 = \text{ReLU}(W_1 \cdot X_1 + W_2 \cdot X_2 + b_1) = \text{ReLU}(0.4 \cdot 0.5 + 0.3 \cdot 0.8 + 0.1) = 0.54$$

$$H_2 = \text{ReLU}(W_3 \cdot X_1 + W_4 \cdot X_2 + b_2) = \text{ReLU}(0.2 \cdot 0.5 + 0.9 \cdot 0.8 - 0.3) = 0.52$$

$$z = W_5 \cdot H_1 + W_6 \cdot H_2 + b_3 = 0.7 \cdot 0.54 + 0.6 \cdot 0.52 + 0.2 = 0.89$$

$$\hat{y} = \text{sigmoid}(z) = \text{sigmoid}(0.89) = \frac{1}{1 + e^{-0.89}} = 0.71$$

LOSS FUNCTION

$$L = -\left[y \log \hat{y} + (1 - y) \log(1 - \hat{y})\right]$$

$$= -\left[1 \cdot \log 0.71 + (1 - 1) \cdot \log(1 - 0.71)\right] = 0.342$$

BACKWARD

$$\frac{\partial L}{\partial \hat{y}} = -\left(\frac{y}{\hat{y}} - \frac{1-y}{1-\hat{y}}\right) = -1.408$$

$$\frac{\partial \hat{y}}{\partial z} = \hat{y}(1-\hat{y}) = 0.2059$$

$$\frac{\partial z}{\partial H_1} = W_5 = 0.7$$

$$\frac{\partial H_1}{\partial W_1} = X_1 = 0.5$$

$$\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial H_1} \cdot \frac{\partial H_1}{\partial W_1}$$

$$\frac{\partial L}{\partial W_1} = -1.408 \cdot 0.2059 \cdot 0.7 \cdot 0.5$$

$$= -0.10146752$$

BACKWARD

$$\frac{\partial L}{\partial \hat{y}} = -\left(\frac{y}{\hat{y}} - \frac{1-y}{1-\hat{y}}\right) = -1.408$$

$$\frac{\partial \hat{y}}{\partial z} = \hat{y}(1-\hat{y}) = 0.2059$$

$$\frac{\partial z}{\partial H_1} = W_5 = 0.7$$

$$\frac{\partial H_1}{\partial b_1} = 1$$

$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial H_1} \cdot \frac{\partial H_1}{\partial b_1}$$

$$\frac{\partial L}{\partial b_1} = -1.408 \cdot 0.2059 \cdot 0.7 \cdot 1$$

$$= -0.20293504$$

UDPATE

$$W_1 = W_1 - \eta \frac{\partial L}{\partial W_1} = 0.4 - 0.1 \cdot -0.10146752 = 0.410146752$$

$$b_1 = b_1 - \eta \frac{\partial L}{\partial b_1} = 0.1 - 0.1 \cdot -0.20293504 = 0.120293504$$

Group 9

DEMO

www.reallygreatsite.com

Group 9

THANK YOU

FOR YOUR ATTENTION

Oct 2024

Lecturer: DO THANH THAI

Data mining - Semester 241