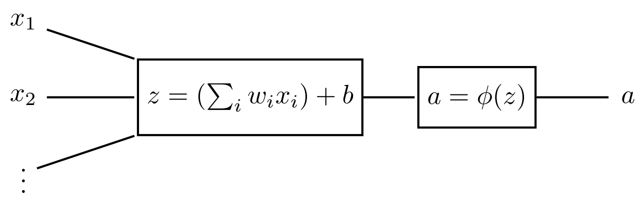
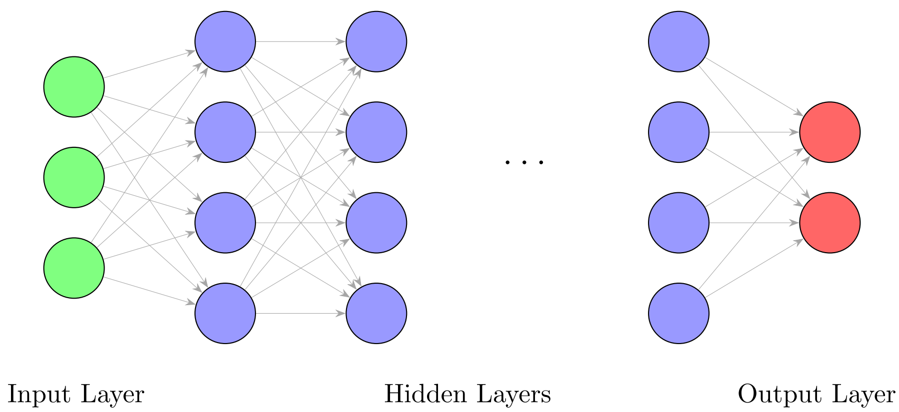
Neural Network Cheat Sheet

Minh Anh Nguyen

Neuron Architecture

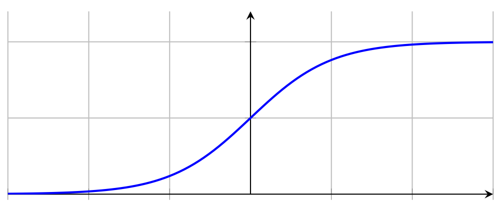


Neural Network Architecture



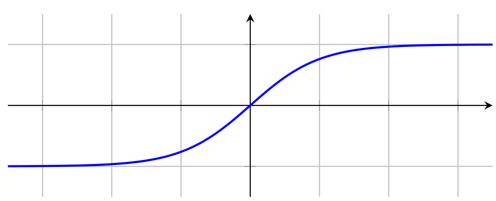
Activation Functions

Sigmoid



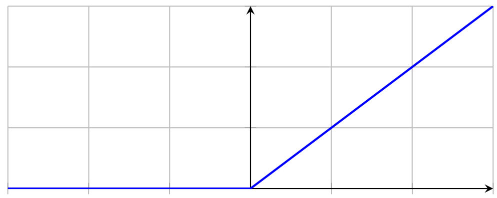
$$a = \sigma(z) = \frac{1}{1+e^{-z}}, \qquad \frac{da}{dz} = a\,(1-a).$$

Tanh



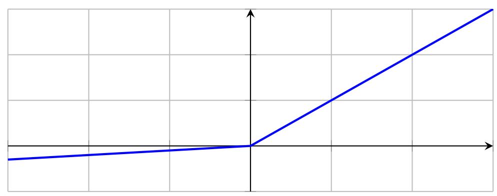
$$a = \tanh(z) = \frac{2}{1+e^{-2z}} - 1, \qquad \frac{da}{dz} = 1 - a^2.$$

ReLU



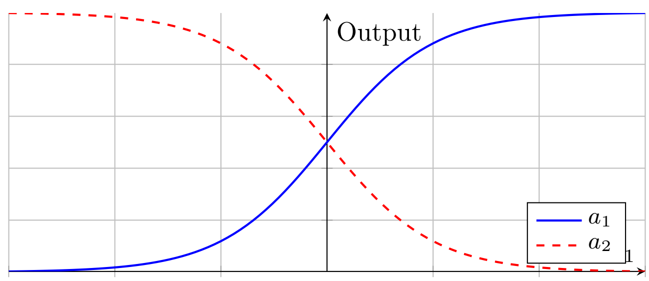
$$a = \max(0, z), \qquad \frac{da}{dz} = \begin{cases} 0, & z \le 0, \\ 1, & z > 0. \end{cases}$$

Leaky ReLU



$$a = \begin{cases} \epsilon\, z, & z \le 0, \\ z, & z > 0, \end{cases} \qquad \frac{da}{dz} = \begin{cases} \epsilon, & z \le 0, \\ 1, & z > 0, \end{cases} \qquad (\epsilon \ll 1).$$

Softmax



$$a_i = \frac{e^{z_i}}{\sum_k e^{z_k}}, \qquad \frac{\partial a_i}{\partial z_j} = a_i\big(\delta_{ij} - a_j\big) = \begin{cases} a_i(1 - a_i), & i = j, \\ -\, a_i\, a_j, & i \neq j, \end{cases}$$

Loss Functions

Sum of Squared Errors (SSE)

$$\mathcal{L}_{\text{SSE}} = \sum_i (y_i - \hat{y}_i)^2, \quad \frac{\partial \mathcal{L}_{\text{SSE}}}{\partial \hat{y}_i} = -2\,(y_i - \hat{y}_i).$$

Mean Squared Error (MSE)

$$\mathcal{L}_{\text{MSE}} = \frac{1}{n} \sum_i (y_i - \hat{y}_i)^2, \quad \frac{\partial \mathcal{L}_{\text{MSE}}}{\partial \hat{y}_i} = -\frac{2}{n}(y_i - \hat{y}_i).$$

Binary Cross-Entropy (Log Loss)

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{n} \sum_i \big[y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)\big], \quad \frac{\partial \mathcal{L}_{\text{BCE}}}{\partial \hat{y}_i} = -\frac{1}{n}\Big(\frac{y_i}{\hat{y}_i} - \frac{1 - y_i}{1 - \hat{y}_i}\Big).$$

Categorical Cross-Entropy (with Softmax)

$$\mathcal{L}_{\text{CCE}} = -\sum_i y_i \log(a_i), \quad \frac{\partial \mathcal{L}_{\text{CCE}}}{\partial z_j} = a_j - y_j,$$

Backpropagation

Error Term

$$\delta = \frac{\partial \mathcal{L}}{\partial a} \cdot \frac{da}{dz}.$$

Gradients w.r.t. Weights and Bias

$$\frac{\partial \mathcal{L}}{\partial w_i} = \delta \cdot x_i, \quad \frac{\partial \mathcal{L}}{\partial b} = \delta.$$

Backprop through Layers

$$\delta_i^{(l)} = \left(\sum_j w_{ij}^{(l+1)} \, \delta_j^{(l+1)} \right) \cdot \frac{da_i^{(l)}}{dz_i^{(l)}}.$$

Gradient Descent Updates

$$w_i \leftarrow w_i - \eta \, \frac{\partial \mathcal{L}}{\partial w_i}, \quad b \leftarrow b - \eta \, \frac{\partial \mathcal{L}}{\partial b}.$$

Notations

x_i^l Input feature i to a neuron in layer l.

w_{ij}^l Weight from neuron i in layer $l-1$ to neuron j in layer l.

b_j^l Bias of neuron j in layer l.

z_j^l Pre-activation of neuron j in layer l: $z_j^l = \sum_i w_{ij}^l \, x_i^l + b_j^l$.

a_j^l Activation of neuron j in layer l: $a_j^l = \phi(z_j^l)$.

ϕ Activation function.

δ_j^l Error term for neuron j in layer l: $\partial \mathcal{L} / \partial z_j^l$.

η Learning rate.

$\hat{y}_i$ Model prediction for sample i.

y_i True target for sample i.

n Number of samples or batch size.