Artificial Neural Networks

Gain an understanding of the structure and background of ANN

Gain an understanding of components and mechanism that enable Learning in a ANN

Gain an understanding of Different Deep Learning Models

Lecture Content

What is an Artificial Neural Network

Structure and Components of a ANN

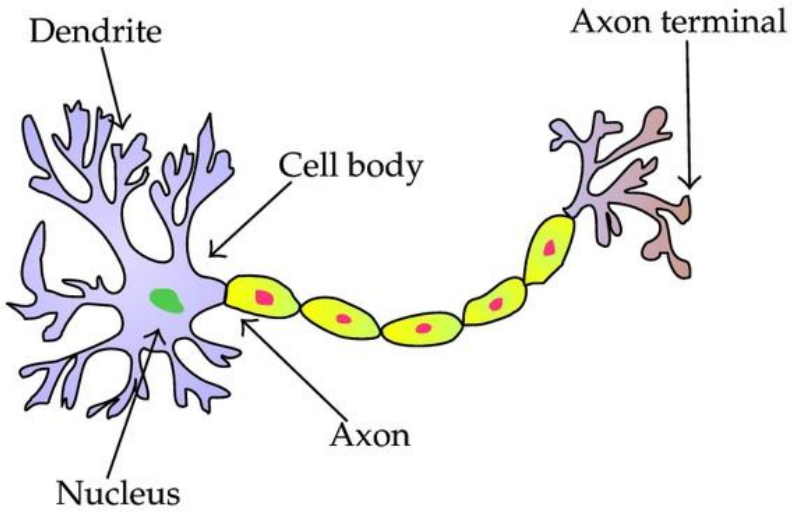
Forward Pass for Predicting Values

Why we need Activation Functions

Backward Pass

Biological Inspiration

- They were inspired by the Biological Neural Networks that makes up our Brains.
- Functionality is very similar
- But the inner mechanism has many differences



Mathematical Intuition

- Let us take a real-world example:
  - You go on shopping for a new Laptop. What are the factors you base your decision on?
    - The Price (x_P)
    - Is it better than the Current Phone. (x_b)
    - Is the merchant reliable. (x_r)
  - How do we make the Decision?
  - Which conditions should we emphasize?

It's Decision-Making Time

$$x_P \cdot w_1 + x_b \cdot w_2 + x_r \cdot w_3 = \mathrm{y}$$

- If y > 5 ➔ buy
- If y<= ➔ Do not buy

Contribution of the Weights on the Decision

$$x_P \cdot w_1 + x_b \cdot w_2 + x_r \cdot w_3 = \mathrm{y}$$

- If y > 5 ➔ buy
- If y<= ➔ Do not buy

- W1?
- W2?
- W3?

We Don’t want to buy from an unreliable Merchant

$$x_P \cdot w_1 + x_b \cdot w_2 + x_r \cdot w_3 = \mathrm{y}$$

- If y >= t ➔ buy(1)
- If y< t ➔ Do not buy (0)

- W1?
- W2?
- W3 ➔

Generalizing the Decision Statement

$$y = \begin{cases} 0, x_P \cdot w_1 + x_b \cdot w_2 + x_r \cdot w_3 < t \\ 1, x_P \cdot w_1 + x_b \cdot w_2 + x_r \cdot w_3 \geq t \end{cases}$$

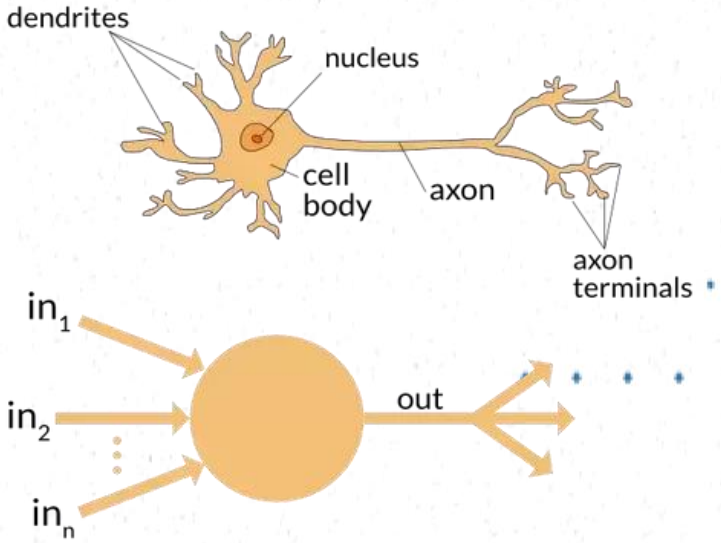
- Its easier to compute with vectors

$$y = \begin{cases} 0, x \cdot w < t \\ 1, x \cdot w \geq t \end{cases}$$

$$y = \begin{cases} 0, x \cdot w + b < 0 \\ 1, x \cdot w + b \geq 0 \end{cases}$$

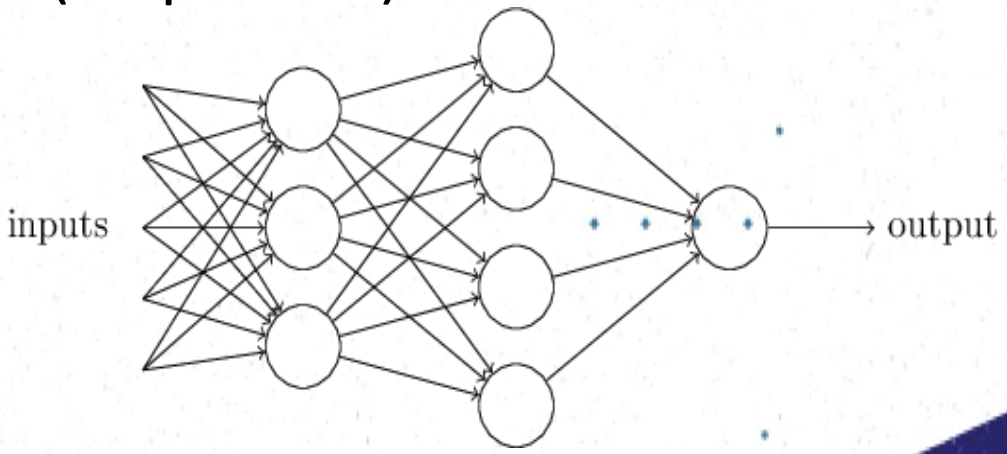
A Perceptron

- Multiple inputs
- Corresponding weights
- Bias
- Single output

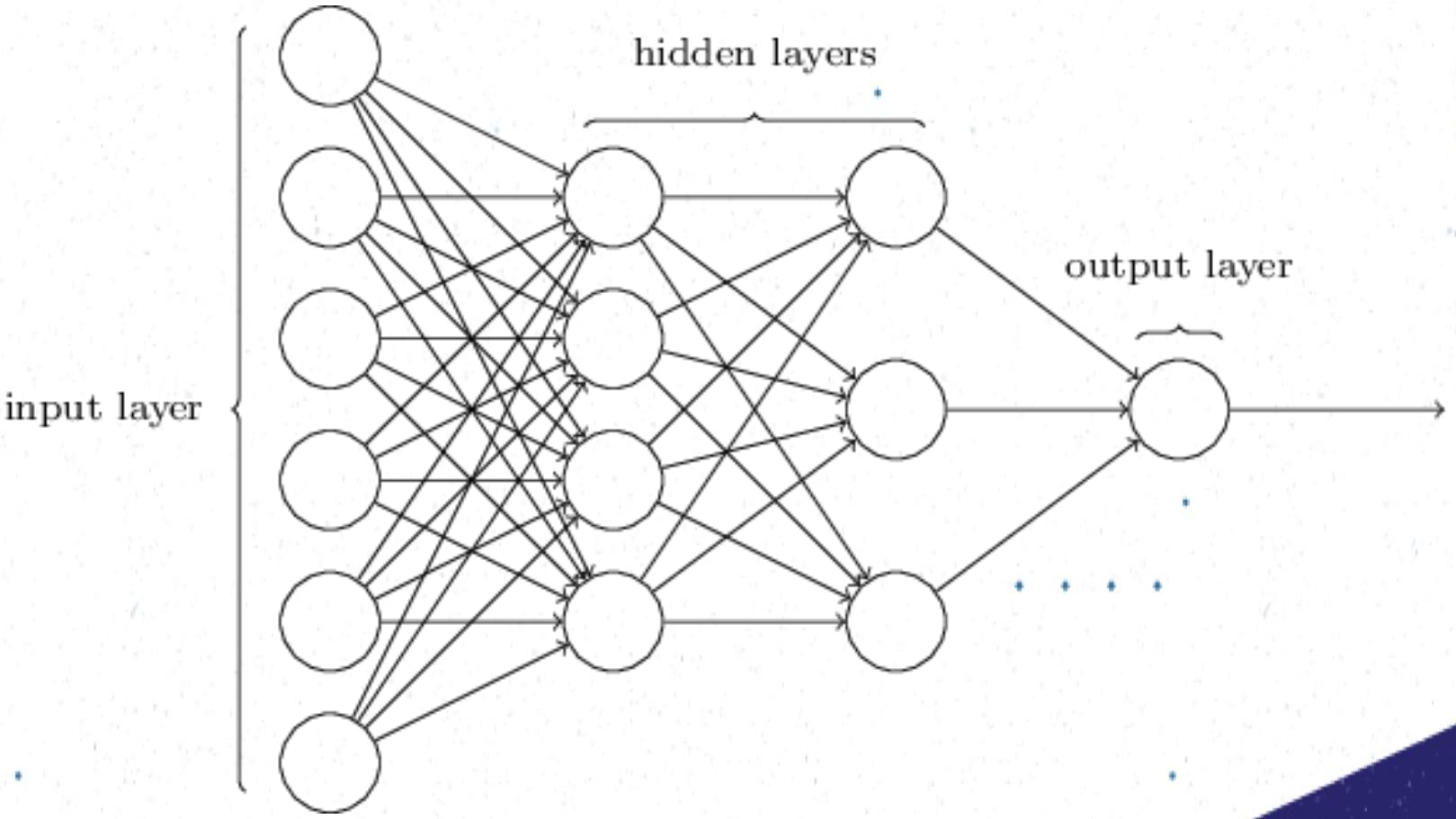


What is a Neural Network

- A Collection of Perceptron
- A Layer can be made by stacking a set of Perceptron to get the outputs from the previous layer or Inputs
- A network is a set of layers that are arranged in a organized manner (Sequential)



Structure of an ANN



ANN output vs Biological NN output

- Firing a Neuron
- 0 or 1

More on Bias

- Bias = -Threshold value
- Use to shift the output value.

Activation Function

- Output of a Perceptron is binary
- This may not work all the time
- Linear output can be useful but will make it difficult to learn complex data
- We need a Non-Linear Continuous value ➔Activation Function.
- Normalize the values

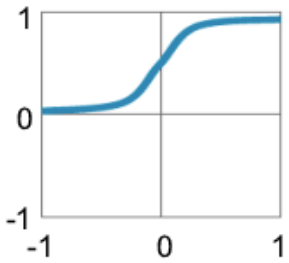
How to Choose an Activation Function

- Any Function can be used as an Activation function if,
  - The function is continuous and differentiable everywhere (or almost everywhere).
  - The derivative of the function does not saturate (i.e., become very small, tending towards zero) over its expected input range. Very small derivatives tend to stall out the learning process.
  - The derivative does not explode (i.e., become very large, tending towards infinity), since this would lead to issues of numerical instability.)

Common Activation Functions

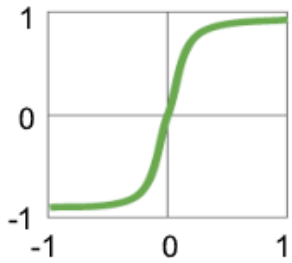
Traditional Non-Linear Activation Functions

Sigmoid



$y=1/(1+e^{-x})$

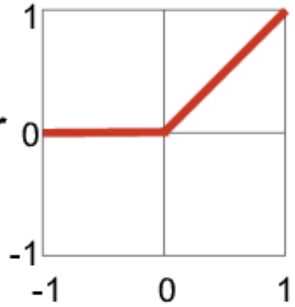
Hyperbolic Tangent



$y=(e^{x}-e^{-x})/(e^{x}+e^{-x})$

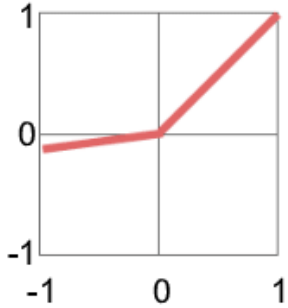
Modern Non-Linear Activation Functions

Rectified Linear Unit (ReLU)



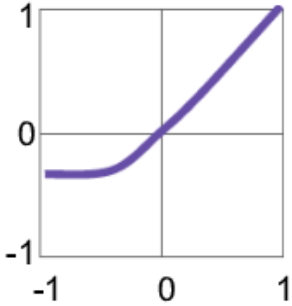
$y=\max(0,x)$

Leaky ReLU



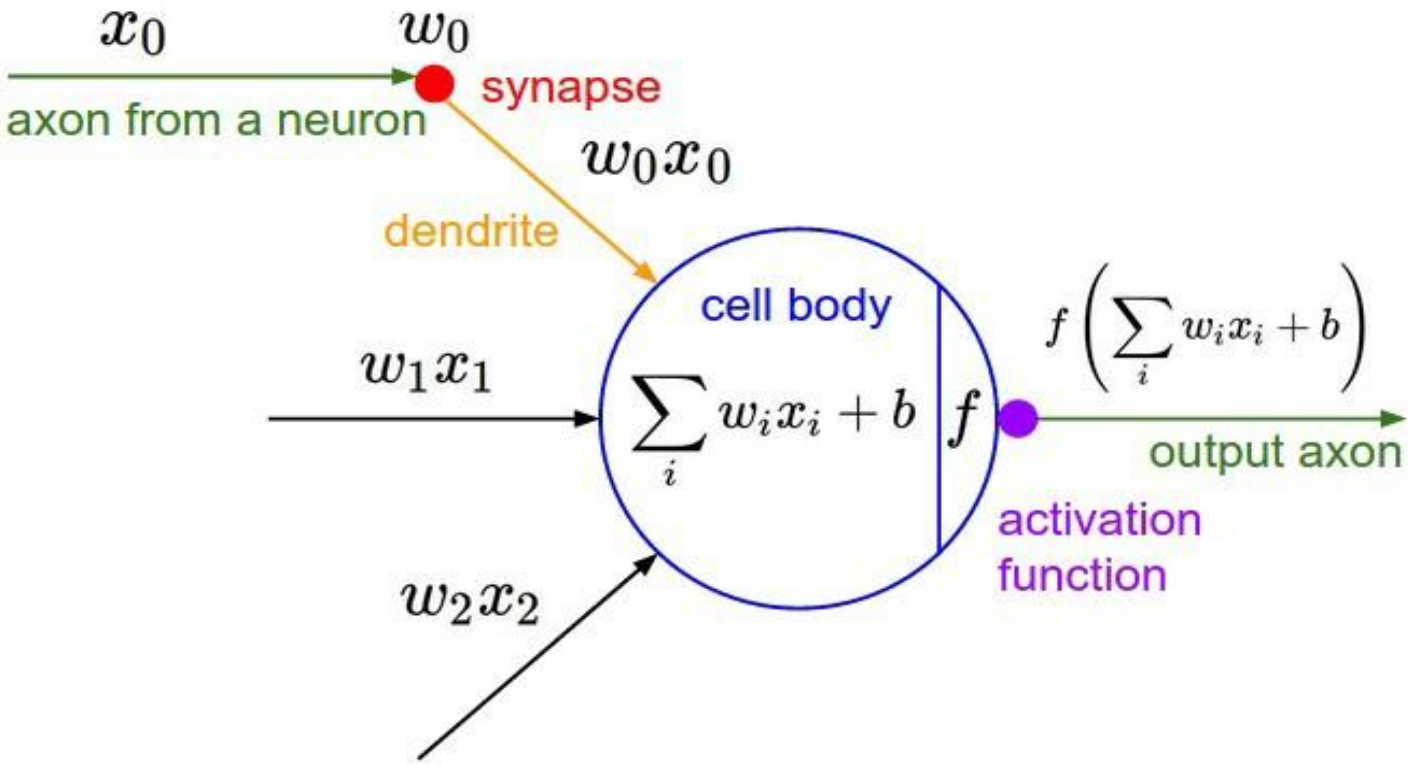
$y=\max(\alpha x,x)$

Exponential LU



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

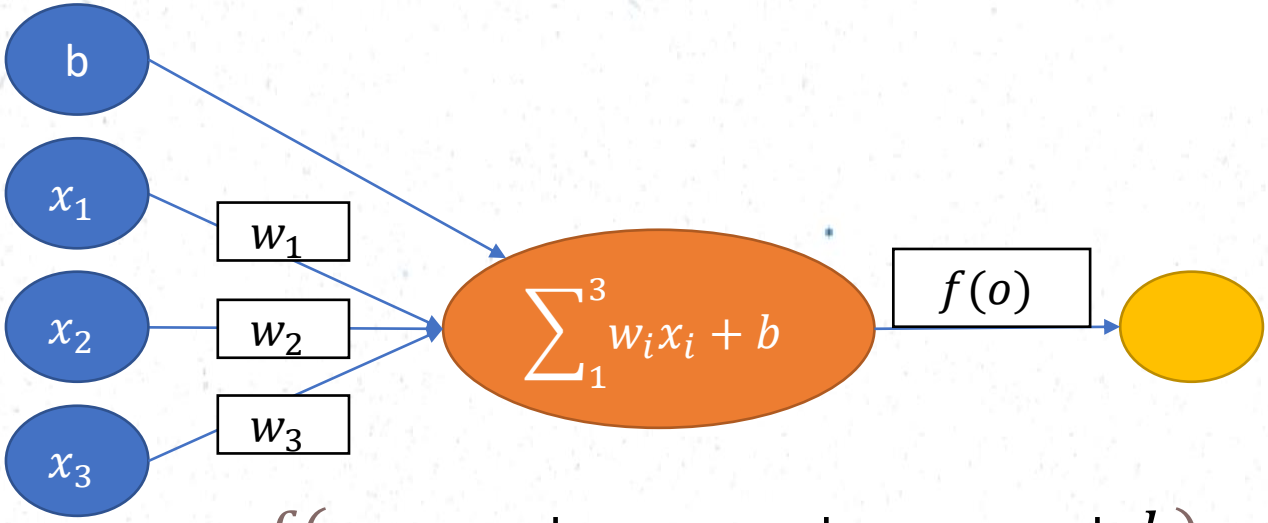
α = small const. (e.g. 0.1)


x_0
w_0
synapse
axon from a neuron
$w_0 x_0$
dendrite
cell body
$w_1 x_1$
$\sum_i w_i x_i + b$
f
$f\left(\sum_i w_i x_i + b\right)$
output axon
activation function
$w_2 x_2$

Feed Forward Networks

- Most used type of ANN
- Signal Flow is uni-directional
- No signal loops.
  - MLP
  - CNN

Forward Pass



$$y = f(x_1 \cdot w_1 + x_2 \cdot w_2 + x_3 \cdot w_3 + b)$$

$$y = f(x \cdot w + b)$$

$$w = [w_1 \;\; w_2 \;\; w_3 \;\; b\,]$$

$$x = \begin{matrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{matrix}$$

Loss Function (Cost)

- A measure of how well the network at Predicting the values.
- Notion of Learning is based on the Loss Function
- Goal of the Network is to find the w and b values such that the cost/loss is minimized.
- Number of Different Loss Functions are available

Different Loss Functions

- Mean Squared Error
- Hinge Loss

Computing Derivatives

$$
\begin{aligned}
\nabla w &= \frac{\partial}{\partial w}[\frac{1}{2} * (f(x) - y)^2] \\
&= \frac{1}{2} * [2 * (f(x) - y) * \frac{\partial}{\partial w}(f(x) - y)] \\
&= (f(x) - y) * \frac{\partial}{\partial w}(f(x)) \\
&= (f(x) - y) * \frac{\partial}{\partial w}\left(\frac{1}{1 + e^{-(wx+b)}}\right) \\
&= (f(x) - y) * f(x) * (1 - f(x)) * x
\end{aligned}
$$

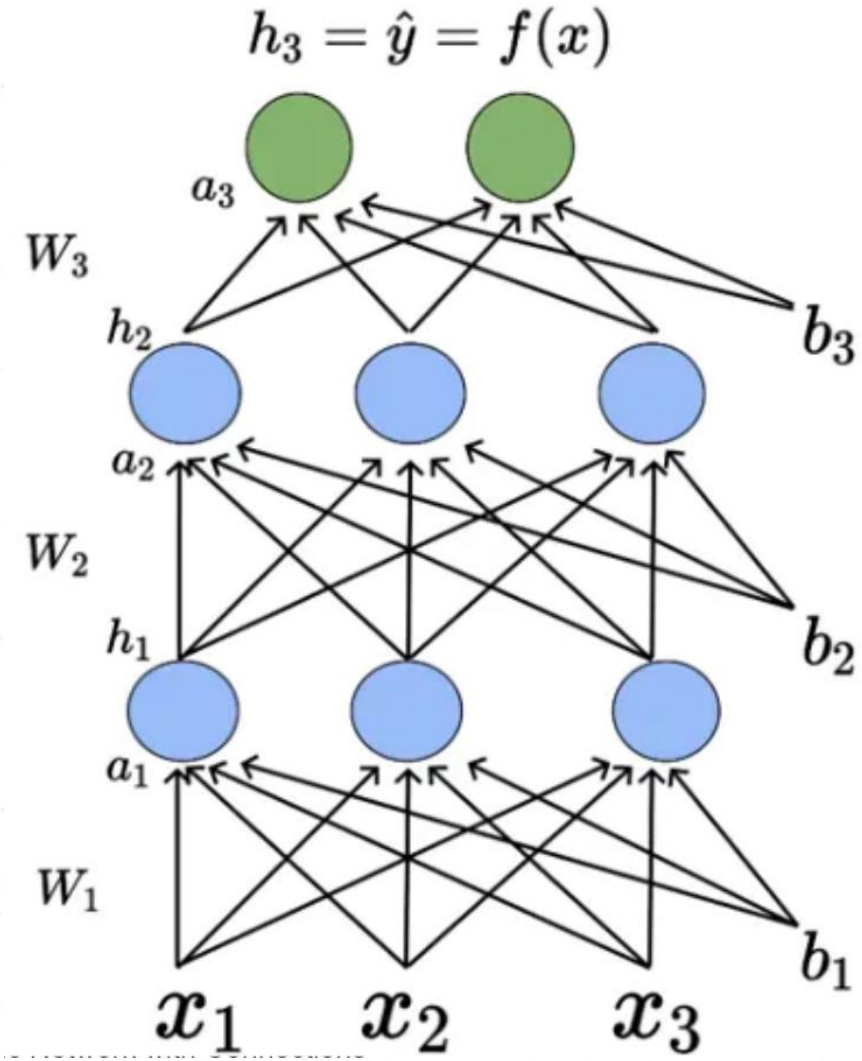
$$
\begin{aligned}
&\frac{\partial}{\partial w}\left(\frac{1}{1 + e^{-(wx+b)}}\right) \\
&= \frac{-1}{(1 + e^{-(wx+b)})^2}\frac{\partial}{\partial w}(e^{-(wx+b)})) \\
&= \frac{-1}{(1 + e^{-(wx+b)})^2} * (e^{-(wx+b)})\frac{\partial}{\partial w}(-(wx + b))) \\
&= \frac{-1}{(1 + e^{-(wx+b)})} * \frac{e^{-(wx+b)}}{(1 + e^{-(wx+b)})} * (-x) \\
&= \frac{1}{(1 + e^{-(wx+b)})} * \frac{e^{-(wx+b)}}{(1 + e^{-(wx+b)})} * (x) \\
&= f(x) * (1 - f(x)) * x
\end{aligned}
$$

Building Complete Neural Networks

- Stacking multiple Nerons to for a layer
- Organizing multiple Layers to form the network

Training a neural net

1. Randomly initialize weights
2. Implement forward propagation to get the output at each neuron
3. Compute the error at the output layer E_{total}
4. Implement backpropagation to compute partial derivatives $\frac{\partial E_{total}}{\partial w^{l}_{jk}}$
5. Use Gradient descent or any other optimization technique to update the weights to minimize E_{total}
6. Repeat this process over multiple iterations (epochs) until the error converges


$h_3 = \hat{y} = f(x)$
a_3
W_3
h_2
b_3
a_2
W_2
h_1
b_2
a_1
W_1
b_1
x_1
x_2
x_3

Additional Reading

- book13.dvi (stanford.edu)

Multi-Layer Perceptron

- MLP is a type of neural network that consists of multiple layers of neurons, including an input layer, one or more hidden layers, and an output layer.
- Each neuron in the hidden layers uses an activation function to transform the input signal into an output signal, which is then passed on to the next layer.
- MLP is a feedforward network, which means that information flows in only one direction, from the input layer to the output layer.

Applications of MLP

- MLP can be used for a variety of tasks, including classification, regression, and prediction.
- MLP has been used for image classification, speech recognition, and natural language processing.
- MLP is particularly useful for tasks where there are complex relationships between the input and output variables.

Convolutional Neural Networks

- CNN is a type of neural network that is particularly well-suited for image and video processing tasks.
- CNN uses convolutional layers to extract features from images and reduce the dimensionality of the input data.
- The output from the convolutional layers is then passed through one or more fully connected layers to produce the final output.

Applications of CNN

- CNN has been used for a variety of image and video processing tasks, including image classification, object detection, and segmentation.
- CNN has also been used in natural language processing tasks, such as text classification and sentiment analysis.
- CNN is particularly useful for tasks where the input data has a grid-like structure, such as images or videos.

Recurrent Neural Networks

- RNN is a type of neural network that is particularly well-suited for sequence data, such as time-series data or natural language processing.
- RNN uses feedback connections to allow information to be passed from one step in the sequence to the next.
- RNN can be trained using backpropagation through time, which is a variant of the standard backpropagation algorithm.

Applications of RNN

- RNN has been used for a variety of sequence data tasks, including speech recognition, machine translation, and sentiment analysis.
- RNN can also be used for time-series forecasting
- RNN is particularly useful for tasks where the output at each step depends on the previous steps in the sequence.

Generative Adversarial Neural Networks

- GAN is a type of neural network that is used for generative tasks, such as image or text generation.
- GAN consists of two neural networks: a generator network and a discriminator network.
- The generator network generates new data samples, while the discriminator network evaluates the generated samples and tries to distinguish them from real data samples.

Applications of GAN

- GAN has been used for a variety of generative tasks, including image generation, text generation, and music generation.
- GAN can also be used for data augmentation, where it is used to generate new data samples to augment an existing dataset.
- GAN is particularly useful for tasks where the output needs to be highly realistic and similar to the input data.

Transformers

- Transformers are a type of neural network architecture that were introduced in 2017 for natural language processing tasks.
- Transformers are based on a self-attention mechanism that allows the model to focus on different parts of the input sequence when processing each token.
- Unlike RNNs, Transformers can process the entire input sequence in parallel, making them much faster and more efficient for long sequences.

Applications of Tranformers

- Transformers have been used for a variety of natural language processing tasks, such as language translation, question answering, and text summarization.
- Transformers are particularly useful for tasks where the input and output sequences can have variable lengths, and where long-range dependencies are important.
- Transformers have also been applied to non-language tasks, such as image processing and music generation.

Generative Pre-Trained Transformers

- GPT models are a family of transformer-based neural networks that are designed for generative tasks, such as text generation.
- GPT models are pre-trained on large amounts of text data, and then fine-tuned on specific downstream tasks.
- GPT models use a left-to-right autoregressive language modeling objective, where the model predicts the next word in a sequence based on the previous words.