Introduction to AI

By Himanshu Sharma

ANN

Artificial Neural Networks (ANNs) are computational models inspired by the structure and function of the human brain. They are a core component of deep learning and are used for a wide variety of tasks like image recognition, natural language processing, and forecasting.

Here are the essential components and principles of Artificial Neural Networks:

1. Neurons (Nodes)

- **Basic Units:** Neurons, also called nodes or units, are the fundamental building blocks of an ANN.
- **Processing:** Each neuron receives inputs, performs a simple computation, and then transmits an output to other neurons.

2. Layers

ANNs are typically organized into layers:

- **Input Layer:** This is the first layer and receives the raw input data (e.g., pixel values of an image, words in a sentence). It has neurons equal to the number of input features.
- **Hidden Layers:** These are intermediate layers between the input and output layers where most of the complex processing and learning occur. An ANN can have one or more hidden layers. Networks with multiple hidden layers are called Deep Neural Networks.
- **Output Layer:** This is the final layer that produces the network's predictions or results. The number of neurons in the output layer depends on the type of problem (e.g., one neuron for binary classification, multiple neurons for multi-class classification or regression).

3. Connections and Weights

- **Connections (Links):** Neurons in one layer are connected to neurons in subsequent layers.
- **Weights (w):** Each connection has an associated weight, which represents the strength or importance of that connection. Weights are parameters that the network learns during training. A higher weight means the input signal through that connection has a stronger influence on the receiving neuron.

4. Bias (b)

- **Offset:** A bias is an additional parameter associated with each neuron (or layer). It allows the neuron to activate even if all inputs are zero, effectively shifting the activation function. It helps the model to fit the data better by providing more flexibility.

5. Summation Function (Weighted Sum)

- Before applying an activation function, each neuron calculates a weighted sum of its inputs and adds the bias. If xi are the inputs and wi are their corresponding weights, the weighted sum z for a neuron is typically: $z=\sum i(xi*wi)+b$

6. Activation Function

- **Non-linearity:** After the weighted sum, an activation function is applied to introduce non-linearity into the network. Without activation functions, an ANN would simply be a linear model, limiting its ability to learn complex patterns.
- **Common Activation Functions:**
  - **Sigmoid:** Squashes values between 0 and 1, often used in output layers for binary classification.
  - **ReLU (Rectified Linear Unit):** Returns the input if positive, otherwise zero ($f(x)=\max(0,x)$). Widely used in hidden layers due to its computational efficiency and ability to mitigate vanishing gradient problems.
  - **Tanh (Hyperbolic Tangent):** Squashes values between -1 and 1.
  - **Softmax:** Used in the output layer for multi-class classification, converting outputs into probabilities that sum to 1.

7. Forward Propagation (Forward Pass)

- **Information Flow:** This is the process where input data is fed into the network and flows through the layers, from the input layer, through the hidden layers, to the output layer.
- **Prediction:** Each neuron performs its weighted sum and applies its activation function, passing the result to the next layer. This continues until the output layer produces a prediction.

8. Loss Function (Cost Function/Objective Function)

- **Error Measurement:** After forward propagation, the network's output (prediction) is compared to the actual target value using a loss function. This function quantifies the discrepancy or "error" between the predicted and true values.
- **Goal:** The objective of training an ANN is to minimize this loss.
- **Common Loss Functions:**
  - **Mean Squared Error (MSE):** For regression tasks.
  - **Binary Cross-Entropy:** For binary classification tasks.
  - **Categorical Cross-Entropy:** For multi-class classification tasks.

9. Back-propagation

- **Learning Mechanism:** This is the core algorithm for training ANNs. After calculating the loss, back-propagation calculates the gradient of the loss function with respect to each weight and bias in the network.
- **Error Propagation:** The error is "propagated backward" through the network, from the output layer to the input layer, indicating how much each weight and bias contributed to the error.

10. Optimization Algorithm

- **Weight Adjustment:** Optimization algorithms use the gradients calculated during back-propagation to adjust the weights and biases of the network in a way that minimizes the loss function.
- **Common Optimizers:**
  - **Gradient Descent:** Iteratively adjusts parameters in the direction of the steepest descent of the loss function.
  - **Stochastic Gradient Descent (SGD):** Updates weights after processing each individual data point or a small batch.
  - **Adam, RMSProp:** More advanced adaptive learning rate optimizers that often converge faster and more robustly.

11. Training

- **Iterative Process:** The entire process of forward propagation, loss calculation, back-propagation, and weight adjustment is repeated iteratively over many epochs (passes through the entire dataset).
- **Learning:** Through this iterative process, the network "learns" to identify patterns in the input data and make more accurate predictions by adjusting its internal parameters (weights and biases).
- These essentials form the foundation of how Artificial Neural Networks are constructed, function, and learn to solve complex problems by mimicking, in a simplified way, the information processing capabilities of the human brain.