

# Initialization

The weights and biases of an ANN are typically initialized randomly before training

The choice of method can affect the performance of the model

These methods aim to set the initial weights and biases in a way that will help the ANN converge to a good solution

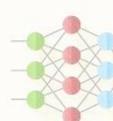
Random initialization, Xavier initialization, and He initialization

## Random

Weights and biases are initialized randomly

Usually from a uniform or normal distribution

Simple and can work well for smaller networks





#### **Xavier**

Takes into account the number of neurons in the input and output layers of the network

Initialized with a Gaussian distribution with zero mean and a standard deviation of sqrt(2/(n\_in + n\_out))

Helps prevent the gradients from exploding or vanishing during training





He

Similar to **Xavier** but takes into account the number of neurons in the current layer

Weights are initialized with a Gaussian distribution with zero mean and a standard deviation of sqrt(2/n)

Works well for deep neural networks





# **Training**

Learns by **updating** its weights and biases based on the error between its **predicted** output and the **actual** output

Aims to minimize the error between the predicted output and the actual output by adjusting the weights and biases of the ANN

Stochastic gradient descent (SGD),
Adam, and Adagrad





#### Backpropagation

Widely used method for training feedforward neural networks

Propagates errors backwards through the network to adjust the weights of the connections between neurons

Repeated many times using stochastic gradient descent (**SGD**)



# Convolutional Neural Network (CNN)

Used for image processing and computer vision tasks

Involves **forward** propagation of the image through a series of **convolutional**, pooling, and fully connected layers

Output is compared to the ground truth label and the weights of the connections between neurons are adjusted using

Backpropagation and SGD



## Recurrent Neural Network (RNN)

Useful for tasks where the input and output are sequences, such as natural language processing and speech recognition

Involves forward propagation of the input sequence through a series of recurrent layers, where the output of one layer is fed back as the input to the next layer

The weights of the connections between neurons are adjusted using backpropagation through time (BPTT) and SGD



# Validation

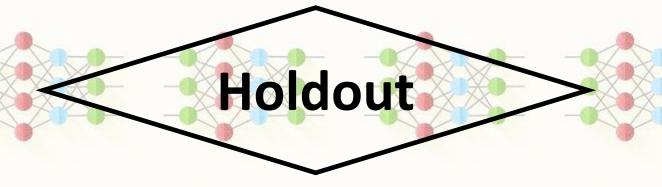
Performance of the ANN is evaluated using a validation set, which consists of data that was not used during training

Measured using a metric such as accuracy, precision, recall, or F1 score

Holdout, Cross-validation, Leaveone-out, Stratified







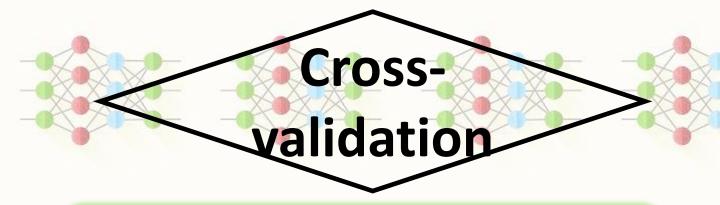
Dataset is split into two parts: a training set and a validation set

Trained on the training set and its performance is evaluated on the validation set

Simple and fast, but it can be prone to overfitting if the validation set is too small







Dataset is split into k parts, or folds

Trained on **k-1 folds** and validated on the remaining fold. Process is repeated **k times**, with each fold used for validation exactly once

Helps reduce **overfitting** and can be useful for **smaller** datasets





#### Stratified

Dataset is split into training and validation sets while maintaining the same class proportions in both sets

Useful for **imbalanced** datasets where one class may have fewer samples than another

Helps ensure that the performance of the ANN is evaluated on a representative sample of the data



