Recurrent Neural Networks (RNNs)

1. Theoretical Overview

What is an RNN?

A **Recurrent Neural Network (RNN)** is a type of neural network designed to handle **sequential data**. Unlike traditional feedforward neural networks, RNNs have **loops** allowing information to persist, making them suitable for tasks like:

- Text generation
- Sentiment analysis
- Machine translation
- Speech recognition

Key Idea

RNNs process sequences **one element at a time**, maintaining a hidden state that contains information from previous time steps. This allows RNNs to learn **temporal dependencies** in the data.

OR

What are RNNs?

Recurrent Neural Networks (RNNs) are designed for sequential data (e.g., time series, text, speech). Unlike feedforward networks, RNNs have loops to retain information from previous steps, enabling them to model temporal dependencies.

Key Features

Hidden State: A memory component capturing information from past inputs.

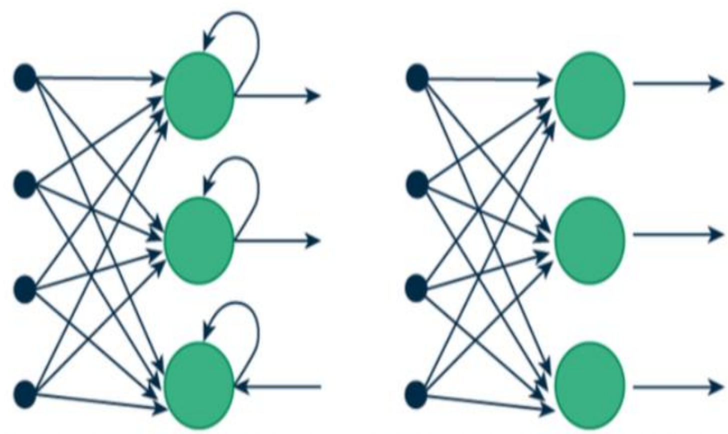
Shared Parameters: The same weights are reused across all time steps.

Applications

Text generation, machine translation, speech recognition.

Time series prediction (stock prices, weather).

Recurrent Neural Networks (RNNs) solve this **by incorporating loops that allow information from previous steps to be fed back into the network**. This feedback enables RNNs to remember prior inputs making them ideal for tasks where context is important.



(a) Recurrent Neural Network (b) Feed-Forward Neural Network

1. State Update:

$$h_t = f(h_{t-1}, x_t)$$

where:

- h_t is the current state
- h_{t-1} is the previous state
- x_t is the input at the current time step

Hidden State Update

$$h_t = \tanh(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$

Output Computation

$$y_t = \mathrm{softmax}(W_{hy}h_t + b_y)$$

> The function `tanh` introduces non-linearity and squashes values between -1 and 1.

2. 🧮 Mathematical Formulation

Let's define the following:

- x_t: Input vector at time step t
- h_t: Hidden state at time t
- y_t: Output at time t
- W_{xh}: Weight matrix from input to hidden
- W_{hh}: Weight matrix from hidden to hidden (recurrent)
- W_{hy}: Weight matrix from hidden to output
- b_h, b_y: Biases

Notation

- $\mathbf{x}_t$: Input at time t (vector of size d).
- $\mathbf{h}_t$: Hidden state at time t (vector of size h).
- $\mathbf{W}_{xh}, \mathbf{W}_{hh}, \mathbf{W}_{hy}$: Weight matrices.
- $\mathbf{b}_h, \mathbf{b}_y$: Bias terms.
- $\tanh$: Activation function (outputs values in $[-1, 1]$).

3. Numerical Example

Task: Compute hidden states and outputs for the input sequence $\mathbf{X} = [1, 2, 3]$.

Parameters (scalars for simplicity):

- $\mathbf{W}_{xh} = 0.5$, $\mathbf{W}_{hh} = 0.3$, $\mathbf{b}_h = 0.1$
- $\mathbf{W}_{hy} = 0.4$, $\mathbf{b}_y = 0.2$
- $\mathbf{h}_0 = 0$ (initial hidden state)

Step-by-Step Calculation:

Time Step	Input ($\mathbf{x}_t$)	Hidden State ($\mathbf{h}_t$)	Output ($\mathbf{y}_t$)
$t = 1$	1	$\tanh(0.5 \times 1 + 0.3 \times 0 + 0.1) = \tanh(0.6) \approx 0.537$	$0.4 \times 0.537 + 0.2 \approx 0.415$
$t = 2$	2	$\tanh(0.5 \times 2 + 0.3 \times 0.537 + 0.1) = \tanh(1.261) \approx 0.850$	$0.4 \times 0.85 + 0.2 \approx 0.540$
$t = 3$	3	$\tanh(0.5 \times 3 + 0.3 \times 0.85 + 0.1) = \tanh(1.855) \approx 0.952$	$0.4 \times 0.952 + 0.2 \approx 0.581$