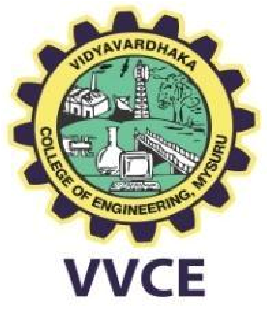
**VIDYAVARDHAKA COLLEGE OF ENGINEERING**

**GOKULAM III STAGE, MYSURU-570 002**

**Accredited by NAAC with A ‘Grade’, *Autonomous institution affiliated to***

**Visvesvaraya Technological University, Belagavi**



**Subject: Research Methodology & Intellectual Property**

**Subject Code: BRIPK608**

**“Latex Document”**

**BACHELOR OF ENGINEERING**

**In**

**INFORMATION SCIENCE & ENGINEERING**

**By**

**Anusha Ponnamma K S [4VV22IS009]**

**Faculty:**

**Prof. Chandini S B**

**Associate professor, ISE Department**



**DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING**

**Accredited by NBA, New Delhi** 2024-25

Recurrent Neural Networks (RNN)

Anusha Ponnamma K S

# Overview

Recurrent Neural Networks (RNNs) are designed for sequential data processing, enabling learning from previous states. Unlike feedforward neural networks, RNNs maintain an internal memory.

# Architecture

For each timestep *t*, activation *a<t>* and output *y<t>* are computed as:

*a<t>* = *g*1(*Waaa<t*−1*>* + *Waxx<t>* + *ba*)

*y<t>* = *g*2(*Wyaa<t>* + *by*)

where:

* *Wax,Waa,Wya,ba,by* are shared weights and biases.
* *g*1*,g*2 are activation functions.

# Advantages and Drawbacks

* **Advantages:**
  + Process sequences of any length.
  + Share weights across timesteps.

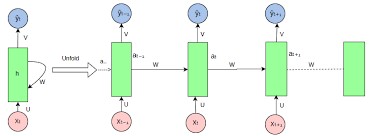


Figure 1: Basic RNN Architecture

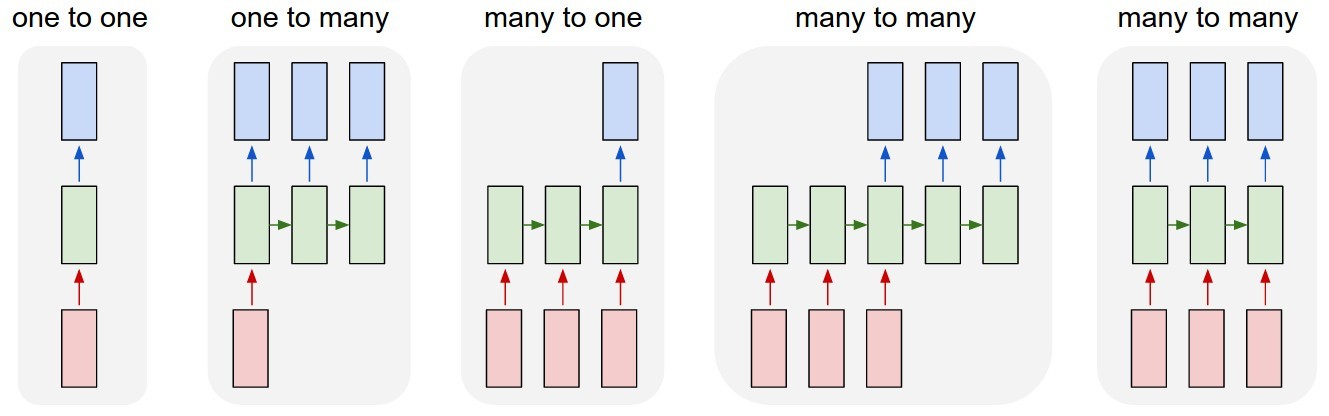


Figure 2: Different Variants of RNNs

* + Maintain context through time dependencies.
* **Drawbacks:**
  + High computational cost.
  + Struggles with long-term dependencies.
  + Cannot directly consider future inputs.

# Types of RNNs

* **One-to-One:** Standard feedforward network.
* **One-to-Many:** Music generation.
* **Many-to-One:** Sentiment classification.
* **Many-to-Many (Equal Length):** Named entity recognition.
* **Many-to-Many (Different Length):** Machine translation.

# Loss Function

The total loss L across all time steps is:

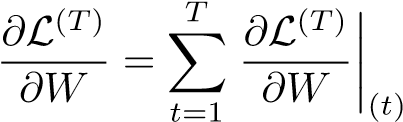
*Ty*

L(*y,y*ˆ ) = XL(*y*ˆ*<t>,y<t>*)

*t*=1

# Backpropagation Through Time (BPTT)

Gradient computation at timestep *T*:



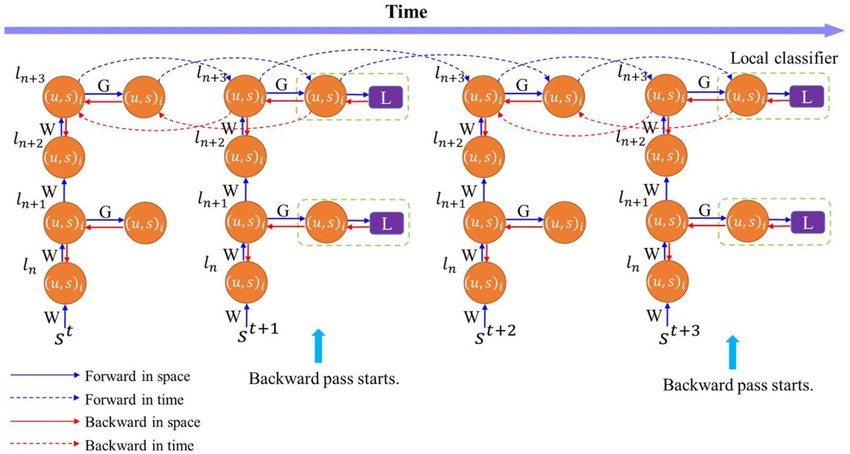


Figure 3: Illustration of Backpropagation Through Time (BPTT)

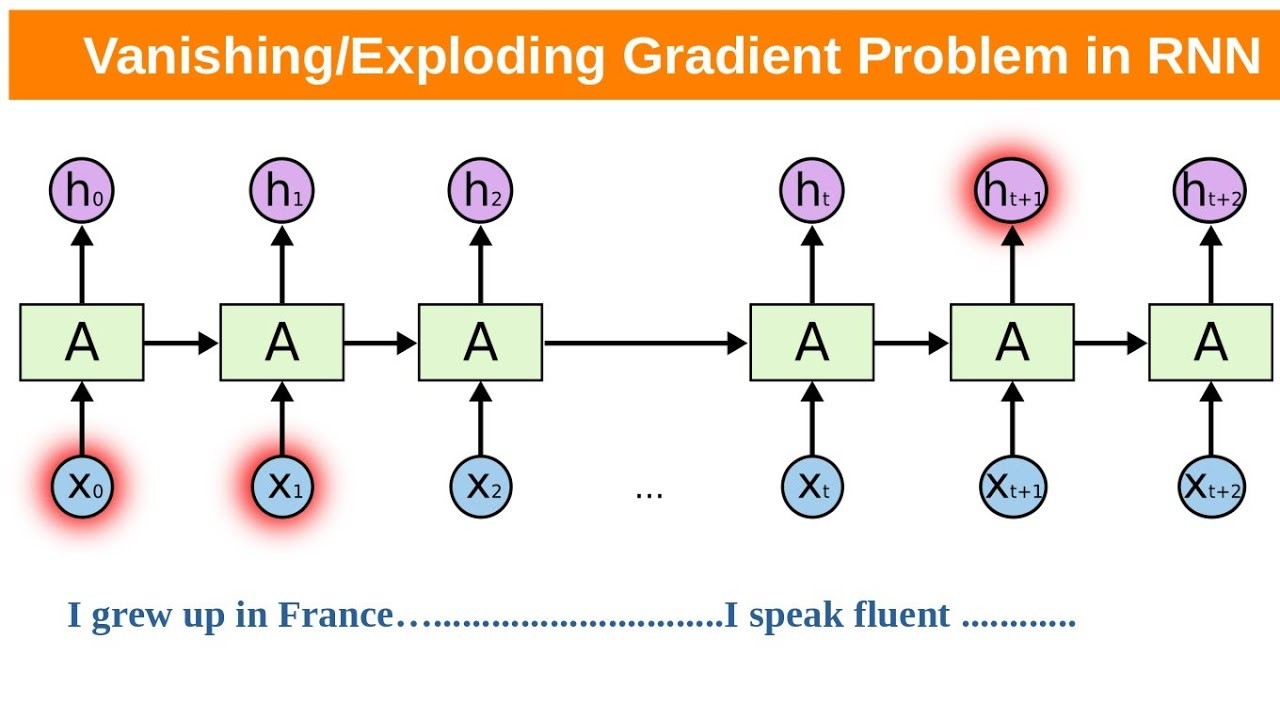


Figure 4: Vanishing and Exploding Gradients in RNNs

# Handling Long-Term Dependencies

* **Vanishing Gradient:** Gradients diminish exponentially, making learning difficult.
* **Exploding Gradient:** Gradients grow exponentially, causing instability.

# Common Activation Functions

* **Sigmoid:** 

• 

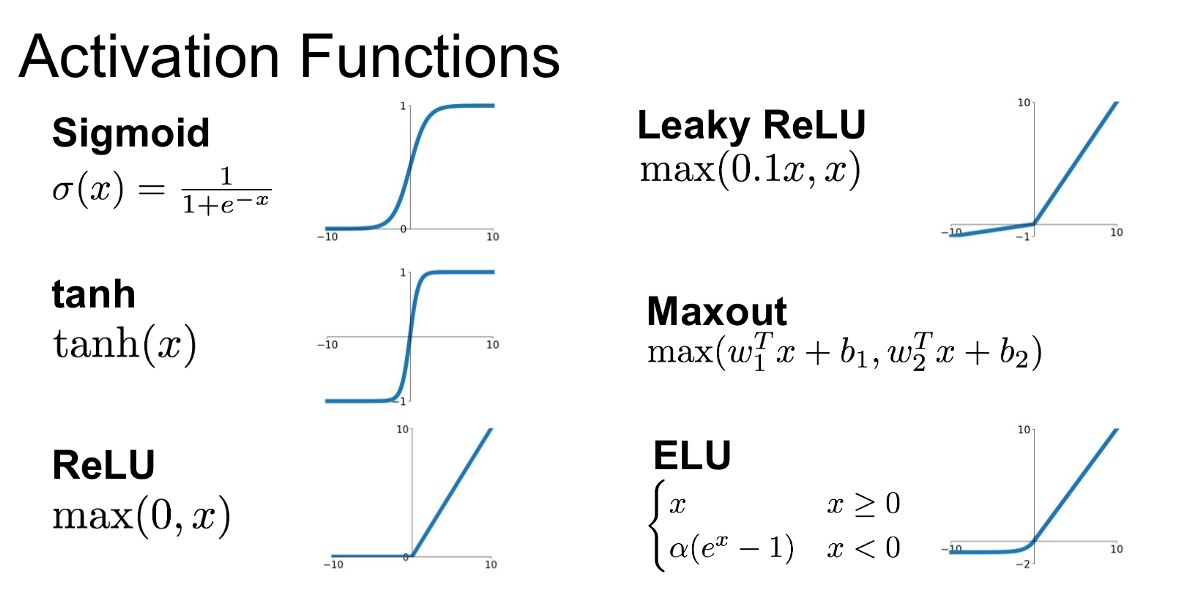


Figure 5: Comparison of Common Activation Functions

* **ReLU:** *g*(*z*) = max(0*,z*)

# CNN vs RNN

|  |  |  |
| --- | --- | --- |
| **Feature** | **CNN (Convolutional Neural**  **Network)** | **RNN (Recurrent Neural Network)** |
| Input Type | Grid-like data (e.g., images) | Sequential data (e.g., text, time series) |
| Architecture | Convolutional layers with filters | Loops with memory through hidden states |
| Data Processing | Processes entire input spatially in parallel | Processes input step-by-step over time |
| Memory of Past Inputs | No memory of previous inputs | Maintains memory of previous inputs via hidden state |
| Parallelism | High parallelism possible during training | Limited parallelism due to sequential dependencies |
| Parameter Sharing | Shared across space (filters) | Shared across time (weights) |
| Common Use Cases | Image classification, object detection | Language modeling, speech recognition, sequence prediction |

Table 1: Comparison of CNN and RNN Architectures

# References

• Stanford CS 230 Recurrent Neural Networks Cheatsheet: https://stanford.edu/ shervine/teaching/cs230/cheatsheet-recurrent-neural-networks