PART # 1

1.Understanding RNN:

Question: What are Recurrent Neural Networks, and how do they differ from traditional feedforward neural networks?

Task: Explain the working of RNN, and how information is passed through the network over time.

Answer:

- Recurrent Neural Networks (RNNs) are a type of neural network designed to handle sequential data, where the order of data points matters (e.g., time series, language). Unlike traditional feedforward neural networks, which assume inputs are independent of each other, RNNs have connections that loop back on themselves, allowing them to maintain a "memory" of previous inputs.

How RNNs Work:

Sequential Processing: RNNs process inputs sequentially. At each time step, they take an input, combine it with the hidden state from the previous step, and produce an output along with a new hidden state.

Hidden State: The hidden state acts as a memory, carrying information from one time step to the next. This allows the RNN to capture dependencies over time.

Weight Sharing: The same set of weights is applied at each time step, which enables RNNs to generalise across sequences of varying lengths.

Passing Information Over Time:

Information is passed through the network by updating the hidden state at each time step. The hidden state, combined with the current input, determines the output and is passed to the next time step. This allows the RNN to accumulate information and model dependencies over time.

2.Stacking RNN Layers and Bi-directional Architecture:

Question: Discuss the advantages and potential drawbacks of stacking RNN layers. What are Bi-directional RNNs, and how do they enhance the performance of sequence models?

Task: Explains when and why you would use stacked RNN layers and bi-directional RNNs in a sequence modelling task.

Answer:

Stacking RNN Layers:

Advantages:

Increased Capacity: Stacking multiple RNN layers allows the model to learn more complex patterns by processing sequences at different levels of abstraction.

Hierarchical Feature Learning: Lower layers can capture short-term dependencies, while higher layers can model more abstract, long-term dependencies.

Drawbacks:

Computational Complexity: More layers mean more parameters to train, leading to increased computational costs and potential overfitting.

Vanishing/Exploding Gradients: Deeper networks are more susceptible to vanishing or exploding gradients, making training difficult.

Bi-directional RNNs:

What They Are: Bi-directional RNNs are a variant of RNNs where two RNNs are run in parallel on the input sequence—one in the forward direction and the other in the backward direction. The outputs of these two RNNs are combined, providing a richer context for each time step.

Advantages:

Enhanced Context: Bi-directional RNNs can utilise both past and future context, leading to improved performance in tasks where the entire sequence is available (e.g., machine translation, speech recognition).

When to Use:

Sequence Prediction: When the task involves predicting an output sequence from an input sequence (e.g., language modelling).

Contextual Understanding: When the entire sequence is available, and understanding both past and future context is beneficial.

3.Hybrid Architecture:

Question: What is a hybrid architecture in the context of sequence modelling? Provide examples of how combining RNNs with other deep learning models can enhance performance.

Answer:

Hybrid Architecture: A hybrid architecture in sequence modelling combines RNNs with other types of neural networks, such as Convolutional Neural Networks (CNNs) or Transformer models, to leverage the strengths of each model.

Examples:

CNN-RNN Hybrid: In tasks like video classification or sentence classification, CNNs can be used to extract spatial features (e.g., from images or text), which are then passed to an RNN to capture temporal dependencies.

RNN-Transformer Hybrid: Transformers are powerful for capturing long-range dependencies and parallel processing. Combining them with RNNs can help handle very long sequences or improve performance on tasks where both local and global context are important.

4.Types of RNN:

Question: List down types of RNN model and explain their structures and differences with RNN.

Answer:

Types of RNN:

Vanilla RNN:

Structure: A simple RNN where each neuron's output is fed back into the network at the next time step.

Challenges: Suffers from vanishing/exploding gradients, making it difficult to learn long-term dependencies.

Long Short-Term Memory (LSTM):

Structure: LSTMs introduce memory cells and gates (input, forget, and output gates) that control the flow of information, allowing the network to maintain information over long periods.

Differences: LSTMs are specifically designed to overcome the vanishing gradient problem, making them more effective at learning long-term dependencies.

Gated Recurrent Unit (GRU):

Structure: GRUs simplify LSTMs by combining the input and forget gates into a single update gate, reducing the number of parameters.

Differences: GRUs are computationally more efficient than LSTMs and often perform similarly, making them a popular alternative.

Bidirectional RNN:

Structure: Consists of two RNNs running in opposite directions (forward and backward), with their outputs combined to provide context from both past and future inputs**.**

Differences: Unlike traditional RNNs, which process data in one direction, bidirectional RNNs can capture information from both directions in the sequence.

Deep RNN (Stacked RNN):

Structure: Multiple RNN layers are stacked on top of each other, where the output of one layer becomes the input to the next.

Differences: Deep RNNs can model more complex patterns by capturing information at multiple levels of abstraction.

These different types of RNNs provide various advantages and are chosen based on the specific requirements of the sequence modelling task at hand.