**1. Explain the Basic Architecture of an RNN Cell**

A **Recurrent Neural Network (RNN)** is a neural network designed for **sequential data** where past information influences future predictions. Unlike traditional feedforward networks, an RNN has **loops** to retain information from previous time steps.

**Architecture:**

* **Input (xₜ):** Feature vector at time step **t**.
* **Hidden state (hₜ):** Stores memory from previous time steps.
* **Output (yₜ):** Prediction at time step **t**.
* **Weight matrices (W, U, V):** Learnable parameters.

**Mathematical Representation:**

ht=tanh⁡(Wxxt+Whht−1+b)h\_t = \tanh(W\_x x\_t + W\_h h\_{t-1} + b)ht​=tanh(Wx​xt​+Wh​ht−1​+b) yt=σ(Vht)y\_t = \sigma(V h\_t)yt​=σ(Vht​)

Where:

* Wx,Wh,VW\_x, W\_h, VWx​,Wh​,V = Weight matrices
* bbb = Bias
* tanh⁡\tanhtanh = Activation function
* σ\sigmaσ = Output activation (e.g., softmax for classification)

**Uses:** Language modeling, time series forecasting, speech recognition.

**Challenges:** Vanishing gradients, short-term memory.

**2. Explain Backpropagation Through Time (BPTT)**

**BPTT** is an extension of backpropagation for **RNNs**. Since RNNs process **sequences**, errors are propagated **through time steps**.

**Steps:**

1. **Forward Pass:** Compute hidden states hth\_tht​ and outputs yty\_tyt​.
2. **Loss Calculation:** Compare predicted yty\_tyt​ with actual values.
3. **Backward Pass:** Compute gradients of loss with respect to parameters **over all time steps**.
4. **Weight Update:** Update weights using **gradient descent**.

**Advantages:** Efficient training for sequence data.  
**Disadvantages:** **Vanishing gradients** make learning long-term dependencies difficult.

**3. Explain Vanishing and Exploding Gradients**

**Vanishing Gradient Problem:**

* In deep RNNs, **gradients shrink exponentially** during backpropagation.
* Earlier layers get **almost zero updates**, causing the network to **forget long-term dependencies**.

**Exploding Gradient Problem:**

* If weights are too large, gradients **explode** exponentially, leading to **unstable training**.

**Solutions:**

* **LSTMs & GRUs** (handle vanishing gradients with gates).
* **Gradient Clipping** (prevents exploding gradients).
* **Batch Normalization** (stabilizes training).

**4. Explain Long Short-Term Memory (LSTM)**

**LSTMs** solve the **vanishing gradient problem** by using **gates** to regulate memory flow.

**Key Components:**

* **Cell state (CtC\_tCt​)**: Stores long-term memory.
* **Forget Gate (ftf\_tft​)**: Decides what to discard.
* **Input Gate (iti\_tit​)**: Decides what to store.
* **Output Gate (oto\_tot​)**: Controls what to output.

**Advantages:**

* Remembers long-term dependencies.
* Solves vanishing gradients.

**Disadvantages:**

* Computationally expensive.

**5. Explain Gated Recurrent Unit (GRU)**

**GRUs** simplify LSTMs by **removing the cell state** and using only **two gates**:

**Key Components:**

* **Reset Gate (rtr\_trt​)**: Controls how much past memory to forget.
* **Update Gate (ztz\_tzt​)**: Controls how much new information to keep.

**Advantages:**

* **Faster than LSTMs** (fewer parameters).
* **Performs well on NLP & time-series tasks**.

**Disadvantages:**

* Less control over long-term memory than LSTMs.

**6. Explain Peephole LSTM**

Peephole LSTMs **modify standard LSTMs** by allowing gates to **look at the previous cell state**.

**New Equations (Forget Gate Example):**

ft=σ(Wf[ht−1,xt,Ct−1]+bf)f\_t = \sigma(W\_f [h\_{t-1}, x\_t, C\_{t-1}] + b\_f)ft​=σ(Wf​[ht−1​,xt​,Ct−1​]+bf​)

**Benefit:**

* Better handling of **timing-sensitive** sequences.

**Drawback:**

* More parameters → Slower training.

**7. Bidirectional RNNs (BiRNNs)**

**BiRNNs** process data **in both directions** (forward & backward).

**Architecture:**

ht→=f(Wxxt+Whht−1→)h\_t^{\rightarrow} = f(W\_x x\_t + W\_h h\_{t-1}^{\rightarrow})ht→​=f(Wx​xt​+Wh​ht−1→​) ht←=f(Wxxt+Whht+1←)h\_t^{\leftarrow} = f(W\_x x\_t + W\_h h\_{t+1}^{\leftarrow})ht←​=f(Wx​xt​+Wh​ht+1←​) ht=[ht→,ht←]h\_t = [h\_t^{\rightarrow}, h\_t^{\leftarrow}]ht​=[ht→​,ht←​]

**Advantages:**

* Captures **past & future** context.
* Useful in **speech recognition, NLP**.

**Disadvantages:**

* Doubles computational cost.

**8. Explain the Gates of LSTM with Equations**

| **Gate** | **Function** | **Equation** |
| --- | --- | --- |
| Forget | Discards old memory | ft=σ(Wf[ht−1,xt]+bf)f\_t = \sigma(W\_f [h\_{t-1}, x\_t] + b\_f)ft​=σ(Wf​[ht−1​,xt​]+bf​) |
| Input | Stores new info | it=σ(Wi[ht−1,xt]+bi)i\_t = \sigma(W\_i [h\_{t-1}, x\_t] + b\_i)it​=σ(Wi​[ht−1​,xt​]+bi​) |
| Cell | Updates memory | Ct=ftCt−1+itCt~C\_t = f\_t C\_{t-1} + i\_t \tilde{C\_t}Ct​=ft​Ct−1​+it​Ct​~​ |
| Output | Controls output | ot=σ(Wo[ht−1,xt]+bo)o\_t = \sigma(W\_o [h\_{t-1}, x\_t] + b\_o)ot​=σ(Wo​[ht−1​,xt​]+bo​) |
| Hidden | Final state | ht=ottanh⁡(Ct)h\_t = o\_t \tanh(C\_t)ht​=ot​tanh(Ct​) |

**9. Explain BiLSTM**

**BiLSTM = BiRNN + LSTM**

* Processes sequences **both forward & backward** using **LSTM units**.

**Applications:**

* **Named Entity Recognition (NER)**
* **Machine Translation**

**10. Explain BiGRU**

**BiGRU = BiRNN + GRU**

* Uses **GRU units** instead of LSTMs.
* **Faster** than BiLSTMs but less expressive.

**Applications:**

* **Speech processing**
* **Sentiment analysis**