**Assignment:-2 Gen\_AI**

**Assignment Tasks**

**Task 1: Conceptual Questions**

**Q-1** What is the difference between RNN and LSTM?

**Ans:-**

**RNN (Recurrent Neural Network):** Remembers short-term information; struggles with long sequences due to vanishing gradient problem.

**LSTM (Long Short-Term Memory):** A special kind of RNN that remembers information for longer periods using gates (input, forget, output) to control memory flow.

LSTM = Improved RNN for long-term memory.

**Q-2** What is the vanishing gradient problem, and how does LSTM solve it?

**Ans:-**

**Vanishing Gradient Problem:** In deep networks (like RNNs), during training, gradients become very small as they backpropagate, causing the network to stop learning long-term patterns.

**How LSTM Solves It:** LSTM uses special gates and a cell state that allow important information to flow unchanged, preventing gradients from vanishing and helping it learn long-term dependencies.

**Q-3** Explain the purpose of the Encoder-Decoder architecture.

**Ans:-**

Encoder-Decoder Architecture Purpose: It is used to transform one sequence into another.

* Encoder: Understands and compresses the input sequence into a context vector.
* Decoder: Uses that context to generate the output sequence.

**Q-4** In a sequence-to-sequence model, what are the roles of the encoder

and decoder?

**Ans:-**

Encoder: Converts the input sequence into a fixed-size context that captures its meaning.

Decoder: Uses that context to generate the output sequence, one step at a time.

Encoder = Understands input, Decoder = Produces output.

**Q-5** How is attention different from a basic encoder-decoder model?

**Ans:-**

Focuses on different parts of the input at each decoding step, using multiple context vectors. Looks at important words dynamically, not just one fixed summary.

**Task 2: Sequence-to-Sequence Data Flow**

**Q-1** Draw or describe the data flow in an encoder-decoder model using

RNN/LSTM.

**Ans:-**

Clearly label:

● Input sequence: The English sentence “hello how are you” is cleaned (lowercase, no punctuation), tokenized (e.g., [2, 50, 30, 10, 3] for , “hello,” “how,” “are you,” ), and padded to a fixed length (e.g., 6). Each token is turned into a 256-dimensional vector via an embedding layer, giving a sequence of vectors [e\_2, e\_50, e\_30, e\_10, e\_3, e\_0].

● Hidden states: The encoder LSTM processes these embeddings one by one. For each token, it updates a hidden state h\_t and cell state c\_t (both 512-dimensional, per your lstm\_units). By the end, we have a sequence of hidden states [h\_1, h\_2, h\_3, h\_4, h\_5] and final states h\_T, c\_T. Your code also applies self-attention, comparing hidden states to create a weighted encoder context.

● Context vector: Instead of just using h\_T, attention creates a dynamic context vector for each decoder step. The decoder’s current hidden state h\_dec\_t is compared (via dot product) to all encoder hidden states [h\_1, h\_2, ...], producing attention weights. These weights are used to combine the encoder’s hidden states into a context vector, focusing on relevant input words (e.g., “hello” for “bonjour”).

● Output sequence: The decoder starts with a token and the encoder’s h\_T, c\_T. For each step, it embeds the current token, processes it with an LSTM to get h\_dec\_t, computes a context vector via attention, concatenates them, and predicts the next French word (e.g., “bonjour”) using a dense layer. This repeats until or a max length, producing a sequence like “bonjour comment vas-tu”.

**Task 8: Model Performance Discussion**

**Q-1** What are the challenges in training sequence-to-sequence models?

**Ans:-**

It's hard because long sentences can be forgotten, and the model needs lots of data. It may also struggle to align words correctly between languages.

**Q-2** What does a “bad” translation look like? Why might it happen?

**Ans:-**

A bad translation might repeat words, miss important parts, or make no sense. This happens when the model doesn't learn properly or guesses wrong.

**Q-3** How can the model be improved further?

**Ans:-**

Use attention to help focus on the input, add more data, try beam search for better output, or switch to a Transformer model for better results.