Recurrent Neural Network (RNN)

- Designed for **sequential data** (text, time series, speech).
- At each step, output depends on **current input + previous hidden state** (memory).
- Problem: struggles with **long-term dependencies** because of **vanishing/exploding gradients**.

□ Example: Predicting the next word in *“The sun rises in the ___”* → RNN remembers recent context (“sun rises in the”) but forgets very old context.

Long Short-Term Memory (LSTM)

- A special kind of RNN that solves the **vanishing gradient** problem.
- Has **memory cells** and **three gates**:
  - **Forget Gate** → decides what to discard.
  - **Input Gate** → decides what new info to store.
  - **Output Gate** → decides what info to use for prediction.
- Can **remember information for long periods**, making it good for translation, speech recognition, etc.

□ Example: In *“Archie lived in China for 13 years … He is fluent in ___”* → LSTM remembers “China” from long ago → predicts “Chinese.”

Gated Recurrent Unit (GRU)

- A **simplified version of LSTM** with **fewer gates** → faster training.
- Has **two gates**:
  - **Update Gate** → controls what to keep/discard.
  - **Reset Gate** → controls how much past info to forget.
- Only one hidden state (no separate cell state like LSTM).
- Often performs **as well as LSTM** but with less computation.

Example: Works like LSTM but lighter → good choice when you want **speed + efficiency**.

□ In short:

- **RNN** → remembers short-term context, suffers from vanishing gradient.
- **LSTM** → fixes this with memory cells & 3 gates, good for long-term dependencies.
- **GRU** → simpler, faster, uses 2 gates, similar performance to LSTM.
