RNN (Recurrent Neural Networks) [used in NLP]

- is a type of Nural Network designed to process sequences of data (like text, speech, or time series) by incorporating an internal "memory" of previous inputs. Unlike traditional neural Networks, which treat inputs independently, the RNN uses information from prior steps to influence the current output

- How they work: The core mechanism that gives RNN's their memory is the hidden state and Feedback loops

1) sequential processing: RNN process data one element (word, fractal, etc) at a time, in order. The output at a specific time step depends on both the current input and information retained from the previous time steps

2) The hidden state (memory) at each step, the network updates an internal hidden state vector. This hidden state acts as a summary or "memory" of all the sequence data the network has processed so far

3) Feedback loop and weigh sharing: The crutrial part is that this updated hidden state is fed back into the network as a input for the next time step. The same set of weights and biases are used (shared) across all time steps, allowing the network to learn generalized patterns across sequences regardless of their position!

4) Contextual understanding: This looping mechanism allows the network to build a contextual understanding of the entire sentence. for example when predicting the next word in a sentence, the RNN uses the history of the previous words to make an informed prediction, rather than just using the single current word

Example: To predict the word "red" in sequence "Apple is ..."

- Timestep 1: when the RNN processes "Apple" it stores a representation in its memory (hidden state) a numarical vector
- Timestep 2: when the RNN processes "is" it recalls "Apple" from its memory and understands the full context "Apple is" to better predict that "red" is likley next word

- Note: Storing of "Apple" happens when hidden state vector is updated by combining the input vector for "Apple" with previous hidden state using weighted matrix + activation function (learned) (non-linear)

- Types: to solve RNN issues like not remembering information in long sequences (gradient vanishing)

- LSTM (long short term memory): uses a system of gates (input forget and output) to control which info in the memory cell is kept or discarded allowing to capture long term dependencies effectively
- GRU (Gated Reccurent unit): A simpler more efficient alternitive too LSTM with only 2 gates (reset and update) that also solves the long term dependecy problem. the update gate controls how much of the input should be combined with existing memory cell, reset gate decides how much of past memory to forget before considering new input.