16/04/24 **Lecture 9: RNN**

**CNN**

• Specialized deep neural network for visual processing.

• Gains efficiency by restricting connectivity between hidden layers and sharing common weights between replicated structures.

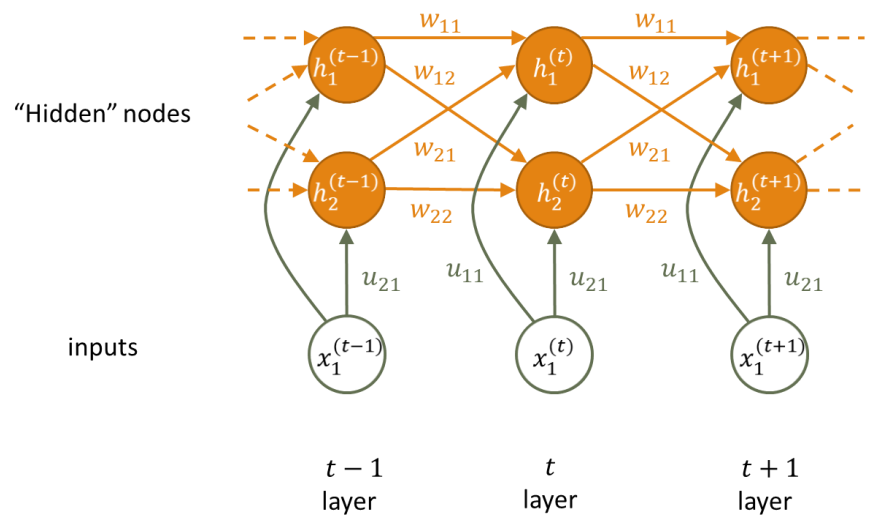
• Best known for processing 2D image data.

• Each input is processed independently in NNs and CNNs.

• For processing sequences or series of data, entire data must be fed into NNs and CNNs.

**RNN**

Recurrent neural networks (RNNs) process data sequences by iterating through elements, retaining information relative to previous experiences, similar to word-by-word processing in reading.



Input Layer:

- Receives input data.

- Transforms input into a format suitable for processing by the network.

Hidden Layer(s):

- Captures sequential information and dependencies within the input data.

- Utilizes recurrent connections to retain the memory of past inputs.

- Each unit in the hidden layer processes a time step of the input sequence.

Output Layer:

- Produces the output based on the information learned from the hidden layers.

- Typically applies an activation function to generate the final output.

- Output may be a prediction, classification, or any other relevant task based on the network's objective.

In essence, the input layer receives data, the hidden layer(s) process sequential information and retains memory, and the output layer generates the final output based on the learned information.

**RNN vs CNN**

Recurrent Neural Networks (RNNs) are best for sequential data like text, speech, or time series, remembering past information through recurrent connections. They process data sequentially, maintaining a hidden state. Convolutional Neural Networks (CNNs) are best for grid-like data like images, detecting local patterns and preserving spatial relationships. RNNs focus on understanding sequences and CNNs on local patterns.

**Intuition**

Imagine you're reading a story. As you progress through each sentence, you understand the current sentence based not only on its own words but also on the context provided by the sentences you've read before.

Now, think of an RNN as a storyteller with memory. It reads the story one word at a time, and at each word, it updates its understanding of the story so far. This understanding is stored in its memory, which it carries forward to comprehend the next word.

Here's how it works step by step:

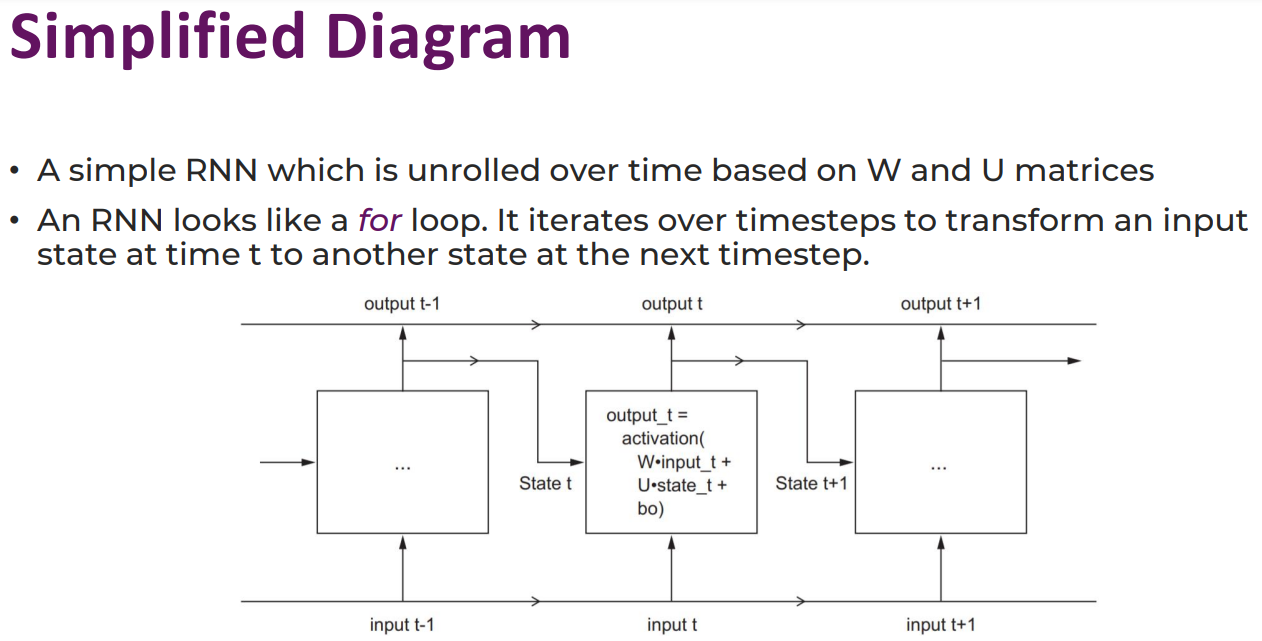
1. \*\*Reading Word by Word:\*\* The RNN starts with the first word of the story.

2. \*\*Understanding the Context:\*\* It considers the current word and combines it with its memory of the story up to that point to understand the context.

3. \*\*Updating Memory:\*\* The RNN updates its memory based on the current word and the context it has understood.

4. \*\*Moving Forward:\*\* With the updated memory, it moves on to the next word, repeating the process until it reaches the end of the story.

In essence, an RNN learns to understand sequences by processing them step by step, updating its internal memory as it goes along. This ability to remember past information makes RNNs powerful for tasks involving sequences, like language translation, sentiment analysis, or time series prediction.

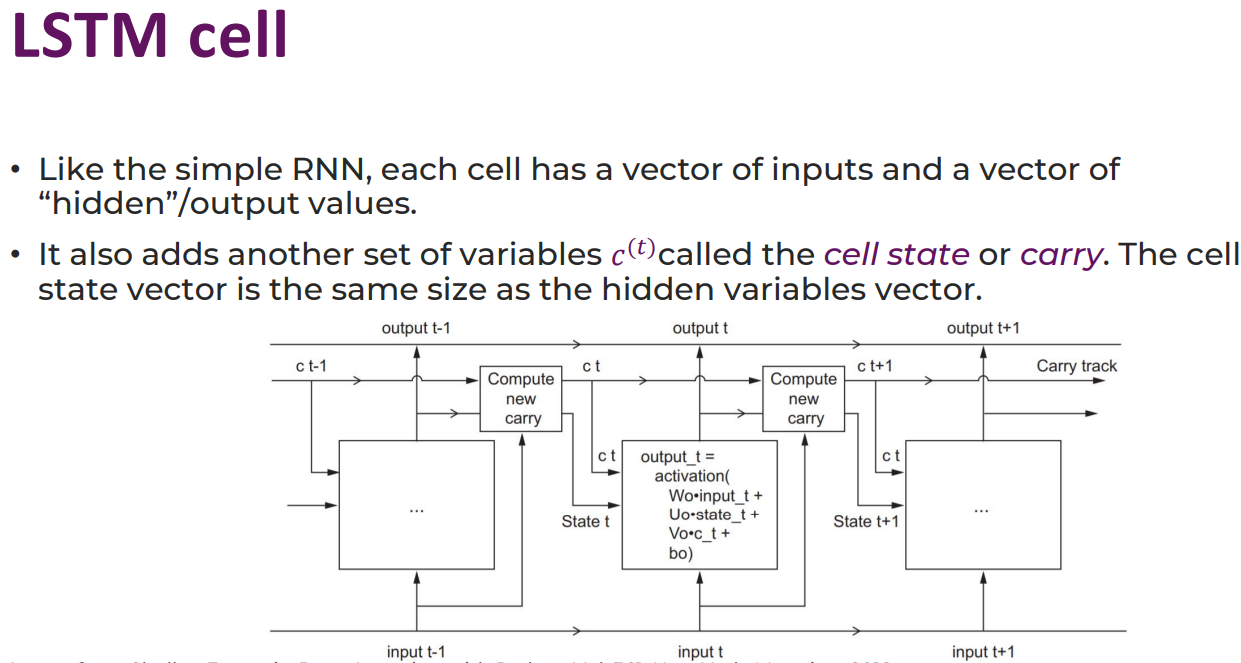


**Issues with RNN**

Repetition of identical stages in networks can cause numerical instabilities, making training difficult. Complex networks like the Long Short-Term Memory (LSTM) network mitigate these issues.

**LSTM**

LSTM networks are a more complex version of simple RNNs, using cells as terminology, with a single cell representing a layer or timestep in a simple RNN.



**Applications**

• Document and time-series classification: Natural language processing and identifying article topics or book authors.

• Sentiment analysis: Classifying positive or negative sentiment in tweets or movie reviews.

• Time series forecasting: Predicting future weather based on recent weather data.