UE21CS343BB2

Topics in Deep Learning

Dr. Shylaja S S
Director of Cloud Computing & Big Data (CCBD), Centre for Data Sciences & Applied Machine Learning (CDSAML)
Department of Computer Science and Engineering
shylaja.sharath@pes.edu

Ack:Divya K,
Teaching Assistant

Overview of lecture

- Why do we need BRNNs?
- Introduction to BRNNs
- BRNN Architecture
- Advantages of BRNNs
- Disadvantages of BRNNs

Why do we need BRNNs?

- In a traditional RNN architecture, information flows only in one direction - from the input to the hidden layers and then to the output. This can hinder the network's ability to capture bidirectional dependencies in the data as only past context is considered.

- Let's take an example of Named Entity Recognition, which is an NLP task that involves extracting and identifying essential information from text used in applications like chatbots:

  "I love Amazon, it's a great website."

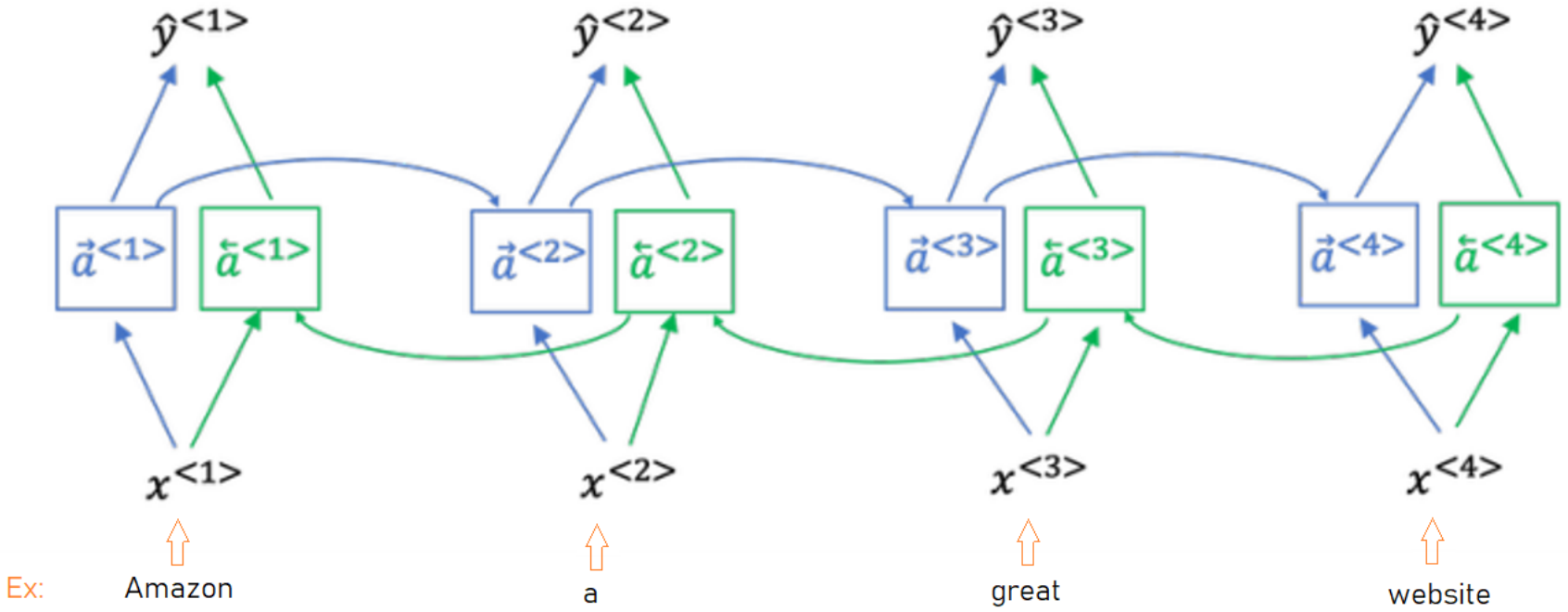
  "I love Amazon, it's a beautiful river."

- Here, the entity is Amazon. It can be annotated as ORG(organization) or LOC(location). This leads to ambiguity unless we consider the future context. In such scenarios, we can make use of Bidirectional RNNs.

Introduction to BRNNs

- In a Bidirectional RNN(BRNN) architecture, the input sequence is processed in both forward and backward directions, starting from the beginning and the end of the sequence simultaneously.
- This allows the network to capture both past and future context of the current time step, resulting in a more comprehensive representation of the input sequence.
- BRNNs consist of two separate RNNs: one that processes the sequence in the forward direction (from the beginning to the end) and another that processes the input sequence in the backward direction(from the end to the beginning). The outputs of both RNNs are then combined (usually by concatenation) to produce the output.

BRNN Architecture



- Here, x<i> represents the input and y<i> represents the output at each timestep i.
- The basic idea of BRNNs is to connect two hidden layers of opposite directions to the same output. By this structure, the output layer can get information from past and future states.
- The blocks can be RNN/LSTM/GRU, bidirectional LSTMs are commonly used.

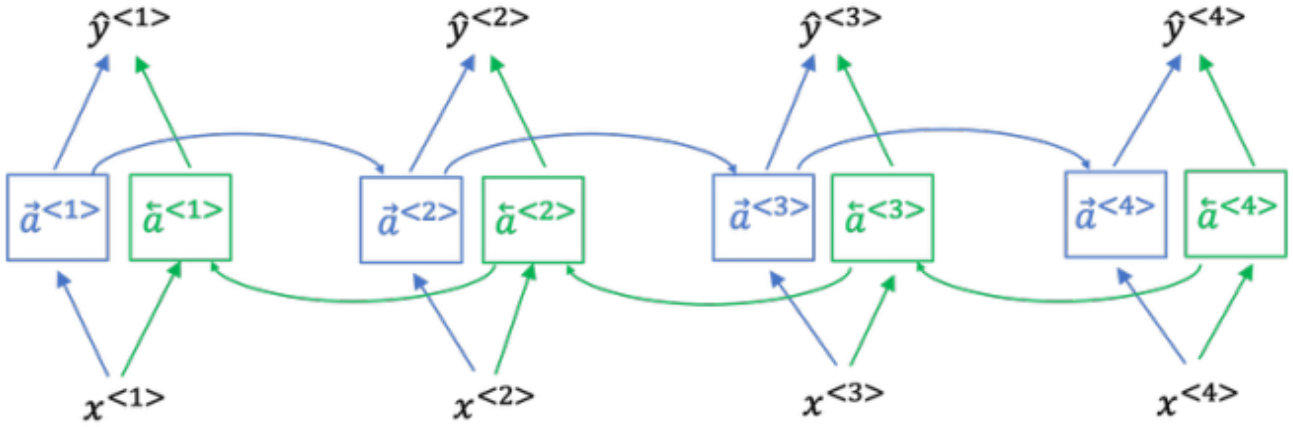
BRNN Architecture

Let's look at the mathematical equations:

$$\vec{a}^{<t>} = f(W_f \vec{a}^{<t-1>} + U_f x^{<t>} + b_f)$$

$$\overleftarrow{a}^{<t>} = f(W_b \overleftarrow{a}^{<t+1>} + U_b x^{<t>} + b_b)$$

$$\hat{y}^{<t>} = g(W_y[\vec{a}^{<t>}, \overleftarrow{a}^{<t>}] + b_y)$$



Here, f and g represent activation functions,
$\vec{a}^{\langle t \rangle}$ represent the hidden state of forward RNN, $\overleftarrow{a}^{\langle t \rangle}$ represent the hidden state of backward RNN
W_f, U_f and b_f represent the weight matrices and bias of the forward RNN,
W_b, U_b and b_b represent the weight matrices and bias of the backward RNN,
$[\vec{a}^{\langle t \rangle}, \overleftarrow{a}^{\langle t \rangle}]$ represent the concatenation of forward and backward outputs

Advantages of BRNNs

- **Capture context:** BRNNs excel in tasks requiring an understanding of both past and future information within a sequence, such as Named Entity Recognition (NER) and sentiment analysis.
- **Capture long-term dependencies:** BRNNs are effective at capturing long-term dependencies in sequential data, making them suitable for tasks like machine translation and time series forecasting.
- **Improved accuracy:** By processing input in both directions, BRNNs can better capture the context and meaning of words or phrases, leading to improved accuracy in tasks such as sentiment analysis, speech recognition, and machine translation.
- **Flexibility:** BRNNs can be used for a wide range of tasks, including sequence classification, sequence prediction, and sequence-to-sequence mapping.
- **Robustness to Input Variability:** BRNNs are robust to variations in input sequences. By considering both forward and backward contexts, they can handle sequences of varying lengths and effectively process inputs of different modalities, such as text, or audio.

Disadvantages of BRNNs

- **Higher Computational Complexity:** Training and inference with BRNNs are more computationally intensive compared to unidirectional RNNs because BRNNs process input sequences in both forward and backward directions, resulting in increased computational requirements and longer training times.
- **Memory Consumption:** BRNNs require more memory compared to unidirectional RNNs due to the need to store activations for both forward and backward passes.
- **Difficulty in Real-Time Applications:** BRNNs are not suitable for real-time applications that require low latency processing, such as online prediction or streaming data analysis as they need the entire sequence of data before making predictions.
- **Potential Overfitting:** Due to their increased complexity, BRNNs can suffer from overfitting, particularly when working with small datasets.

Acknowledgements

https://www.deeplearning.ai/courses/deep-learning-specialization/

https://youtube.com/playlist?list=PLKnIA16_RmvYuZauWaPlRTC54KxSNLtNn&si=a2L-j8rAkG15EWKY

UE21CS343BB2

Topics in Deep Learning

Dr. Shylaja S S
Director of Cloud Computing & Big Data (CCBD), Centre for Data Sciences & Applied Machine Learning (CDSAML)
Department of Computer Science and Engineering
shylaja.sharath@pes.edu

Ack:Divya K,
Teaching Assistant