19EC503 – ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

UNIT 5 - DEEP LEARNING

TOPIC: Bidirectional Recurrent Neural Networks

Bidirectional Recurrent Neural Network

Introduction

Recurrent Neural Networks (RNNs) are a particular class of neural networks that was created with the express purpose of processing sequential input, including speech, text, and time series data. RNNs process data as a sequence of vectors rather than feedforward neural networks, which process data as a fixed-length vector. Each vector is processed depending on the hidden state from the previous phase.

The network can store data from earlier steps in the sequence in a type of memory by computing the hidden state by taking into account both the current input and the hidden state from the previous phase. RNNs are thus well suited for jobs that call for knowledge of the context and connections among sequence elements.

Even though conventional RNNs can handle variable-length sequences, they sometimes have trouble with the vanishing gradient problem. Gradients during backpropagation become extremely small at this point, making it challenging for the network to learn from the data. Many RNN versions, such LSTMs, and GRUs, which use gating methods to regulate the flow of information and enhance learning, have been created to address this problem.

Bi-directional Recurrent Neural Network

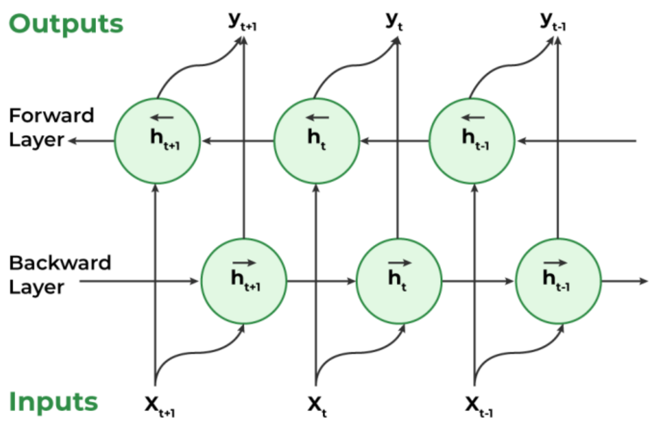
An architecture of a neural network called a bidirectional recurrent neural network (BRNN) is made to process sequential data. In order for the network to use information from both the past and future context in its predictions, BRNNs process input sequences in both the forward and backward directions. This is the main distinction between BRNNs and conventional recurrent neural networks.

A BRNN has two distinct recurrent hidden layers, one of which processes the input sequence forward and the other of which processes it backward. After that, the results from these hidden layers are collected and input into a prediction-making final layer. Any recurrent neural network cell, such as Long Short-Term Memory (LSTM) or Gated Recurrent Unit, can be used to create the recurrent hidden layers.

The BRNN functions similarly to conventional recurrent neural networks in the forward direction, updating the hidden state depending on the current input and the prior hidden state at each time step. The backward hidden layer, on the other hand, analyses the input sequence in the opposite manner, updating the hidden state based on the current input and the hidden state of the next time step.

Compared to conventional unidirectional recurrent neural networks, the accuracy of the BRNN is improved since it can process information in both directions and account for both past and future contexts. Because the two hidden layers can complement one another and give the final prediction layer more data, using two distinct hidden layers also offers a type of model regularisation.

In order to update the model parameters, the gradients are computed for both the forward and backward passes of the backpropagation through the time technique that is typically used to train BRNNs. The input sequence is processed by the BRNN in a single forward pass at inference time, and predictions are made based on the combined outputs of the two hidden layers. layers.



Bi-directional Recurrent Neural Network

Working of Bidirectional Recurrent Neural Network

1. Inputting a sequence: A sequence of data points, each represented as a vector with the same dimensionality, are fed into a BRNN. The sequence might have different lengths.
2. Dual Processing: Both the forward and backward directions are used to process the data. On the basis of the input at that step and the hidden state at step t-1, the hidden state at time step t is determined in the forward direction. The input at step t and the hidden state at step t+1 are used to calculate the hidden state at step t in a reverse way.
3. Computing the hidden state: A non-linear activation function on the weighted sum of the input and previous hidden state is used to calculate the hidden state at each step. This creates a memory mechanism that enables the network to remember data from earlier steps in the process.
4. Determining the output: A non-linear activation function is used to determine the output at each step from the weighted sum of the hidden state and a number of output weights. This output has two options: it can be the final output or input for another layer in the network.
5. Training: The network is trained through a supervised learning approach where the goal is to minimize the discrepancy between the predicted output and the actual output. The network adjusts its weights in the input-to-hidden and hidden-to-output connections during training through backpropagation.

Applications of Bidirectional Recurrent Neural Network

Bi-RNNs have been applied to various natural language processing (NLP) tasks, including:

1. Sentiment Analysis: By taking into account both the prior and subsequent context, BRNNs can be utilized to categorize the sentiment of a particular sentence.
2. Named Entity Recognition: By considering the context both before and after the stated thing, BRNNs can be utilized to identify those entities in a sentence.
3. Part-of-Speech Tagging: The classification of words in a phrase into their corresponding parts of speech, such as nouns, verbs, adjectives, etc., can be done using BRNNs.
4. Machine Translation: BRNNs can be used in encoder-decoder models for machine translation, where the decoder creates the target sentence and the encoder analyses the source sentence in both directions to capture its context.
5. Speech Recognition: When the input voice signal is processed in both directions to capture the contextual information, BRNNs can be used in automatic speech recognition systems.

Advantages of Bidirectional RNN

- Context from both past and future: With the ability to process sequential input both forward and backward, BRNNs provide a thorough grasp of the full context of a sequence. Because of this, BRNNs are effective at tasks like sentiment analysis and speech recognition.
- Enhanced accuracy: BRNNs frequently yield more precise answers since they take both historical and upcoming data into account.
- Efficient handling of variable-length sequences: When compared to conventional RNNs, which require padding to have a constant length, BRNNs are better equipped to handle variable-length sequences.
- Resilience to noise and irrelevant information: BRNNs may be resistant to noise and irrelevant data that are present in the data. This is so because both the forward and backward paths offer useful information that supports the predictions made by the network.
- Ability to handle sequential dependencies: BRNNs can capture long-term links between sequence pieces, making them extremely adept at handling complicated sequential dependencies.

Disadvantages of Bidirectional RNN

- Computational complexity: Given that they analyze data both forward and backward, BRNNs can be computationally expensive due to the increased amount of calculations needed.
- Long training time: BRNNs can also take a while to train because there are many parameters to optimize, especially when using huge datasets.
- Difficulty in parallelization: Due to the requirement for sequential processing in both the forward and backward directions, BRNNs can be challenging to parallelize.
- Overfitting: BRNNs are prone to overfitting since they include many parameters that might result in too complicated models, especially when trained on short datasets.
- Interpretability: Due to the processing of data in both forward and backward directions, BRNNs can be tricky to interpret since it can be difficult to comprehend what the model is doing and how it is producing predictions.