19EC503 – ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

UNIT 5 - DEEP LEARNING

TOPIC: Recursive Neural Network

Recursive Neural Network

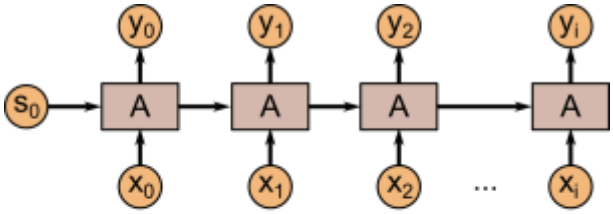
Recursive Neural Networks are a type of neural network architecture that is specially designed to process hierarchical structures and capture dependencies within recursively structured data. Unlike traditional feedforward neural networks (RNNs), Recursive Neural Networks or RvNN can efficiently handle tree-structured inputs which makes them suitable for tasks involving nested and hierarchical relationships.

What is RvNN?

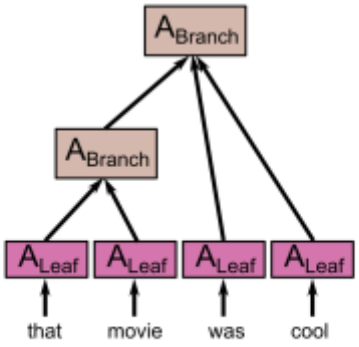
From the family of Neural networks, RvNN is a special Deep Learning which has the ability to operate on structured input data like parse trees in natural language processing or molecular structures in chemistry. The network processes the input recursively, combining information from child nodes to form representations for parent nodes. RvNN mainly used in some NLP tasks like sentiment analysis etc. by processing data which is in the format of parse tree. RvNN processes parse trees by assigning vectors to each word or subphrase based on the information from its children which allows the network to capture hierarchical relationships and dependencies within the sentence.

Working Principals of RvNN

Some of the key-working principals of RvNN is discussed below:



Recurrent Neural Net



Recursive Neural Net

1. Recursive Structure Handling: RvNN is designed to handle recursive structures which means it can naturally process hierarchical relationships in data by combining information from child nodes to form representations for parent nodes.
2. Parameter Sharing: RvNN often uses shared parameters across different levels of the hierarchy which enables the model to generalize well and learn from various parts of the input structure.
3. Tree Traversal: RvNN traverses the tree structure in a bottom-up or top-down manner by simultaneously updating node representations based on the information gathered from their children.
4. Composition Function: The composition function in RvNN combines information from child nodes to create a representation for the parent node. This function is crucial in capturing the hierarchical relationships within the data.

A recursive network is only a recurrent network generalization. In a recurrent network, weights are exchanged (and dimensionality stays constant) over the sequence and, in a test cycle, you can see a list of varying lengths then you will find in train times while working with position-dependent weights. For the same reason, the weights are distributed in a recursive network (and dimensionality stays constant).

Recursive Neural Networks (RvNNs)

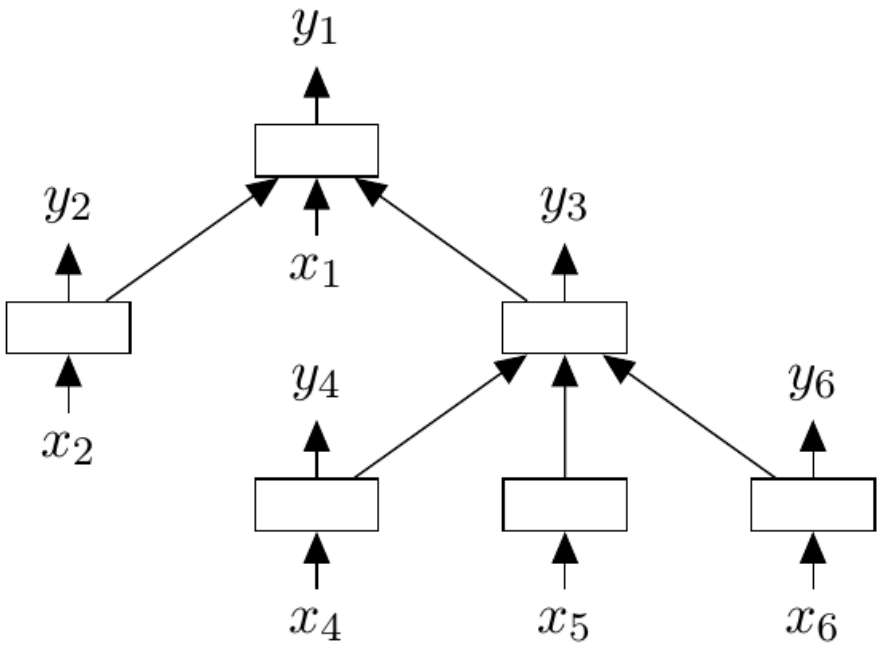
In order to understand Recurrent Neural Networks (RNN), it is first necessary to understand the working principle of a feedforward network. In short, we can say that it is a structure that produces output by applying some mathematical operations to the information coming to the neurons on the layers.

The information received in the Feedforward working structure is only processed forward. In this structure, an output value is obtained by passing the input data through the network. The error is obtained by comparing the obtained output value with the correct values. The weight values on the network are changed depending on the error, and in this way, a model that can give the most accurate result is created.

Non-linear adaptive models that can learn in-depth and structured information are called Recursive Neural Networks (RvNNs). RvNNs were effective in natural language processing for learning sequences and structures of the trees, primarily phrases, and sentences based on word embedding.

RvNN is the connections between neurons are established in directed cycles. These models have however not yet been universally recognized. The key explanation for this is its underlying ambiguity. Not only for being highly complex structures for information retrieval but also because of a costly computational learning period.

RvNN is more of a hierarchy, where the input series actually is without time aspects, but the input must be hierarchically interpreted in a tree-type manner.



Recursive Neural Networks Architecture

The children of each parent node are just a node like that node. RvNNs comprise a class of architectures that can work with structured input. The network looks at a series of inputs, each time at x1, x2… and prints the results of each of these inputs.

This means that the output depends on the number of neurons in each layer of the network and the number of connections between them. The simplest form of a RvNNs, the vanilla RNG, resembles a regular neural network. Each layer contains a loop that allows the model to transfer the results of previous neurons from another layer.

Schematically, RvNN layer uses a loop to iterate through a timestamp sequence while maintaining an internal state that encodes all the information about that timestamp it has seen so far.

This allows us to create recurring models without having to make difficult configuration decisions. We can use the same parameters as input to perform the same tasks at both the input and the hidden level to generate output, but we can also define different parameters for the output position (e.g. the number of inputs and outputs) for user-defined behavior. This can be used in a variety of ways, such as a single layer, multiple layers, or a combination of layers.

The performance, robustness, and scalability of RvNNs are super exciting compared to other types of artificial neural networks since neural networks consist of a series of connections between nodes in a directed graph or a sequence of connected nodes.

Difference between RvNN and RNN

RNN and RvNN both belong to the family of Neural Network, but they differ significantly. Some of the differences are discussed below:

Differencing facts	RNN	RvNN
Data Structure	It is designed for sequential data like time series or sequences of words.	It is specially designed to tailor for hierarchical data structures like trees.
Processing Mechanism	It processes sequences by maintaining hidden states which capture temporal dependencies.	It processes hierarchical structures by recursively combining information from child nodes.
Applications	Commonly used in tasks like natural language processing, speech recognition and time series prediction.	Commonly used in applications which involves hierarchical relationships like parsing in NLP, analyzing molecular structures or image segmentation.
Topology	It has linear topology where information flows sequentially through time steps.	It has a tree-like or hierarchical topology which allows them to capture nested relationships within the data.