Recurrent Neural Networks (RNNs)

Hossein Javidnia

Problem

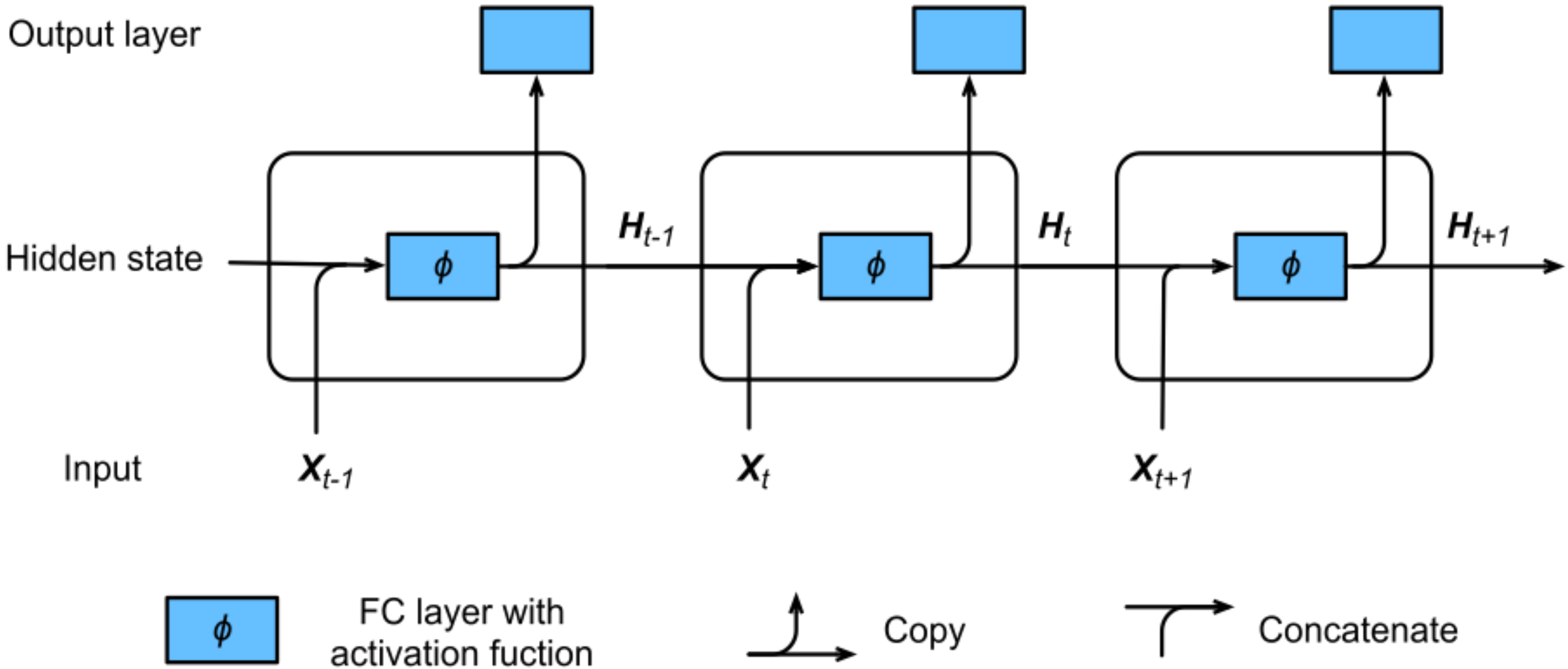
- One of the most difficult type of data to handle and forecast is sequential data.
- Sequential data is different from other types of data in the sense that while all the features of a typical dataset can be assumed to be order-independent, this cannot be assumed for a sequential dataset.
- To handle such type of data, the concept of Recurrent Neural Networks (RNNs) was proven to be effective.

Idea

- The Recurrent Neural Network consists of multiple fixed activation function units, one for each time step.
- Each unit has an internal state which is called the hidden state of the unit. This hidden state signifies the past knowledge that that the network currently holds at a given time step.
- This hidden state is updated at every time step to signify the change in the knowledge of the network about the past.

Steps

- Take input the previous hidden state vector and the current input vector.
- Element-wise multiplication of the hidden state vector by the hidden state weights and similarly perform the element wise multiplication of the current input vector and the current input weights.
- Perform the vector addition of the two parameterized vectors and then calculate the element-wise hyperbolic tangent to generate the new hidden state vector.


Output layer
Hidden state
Input
ϕ
H_{t-1}
H_t
H_{t+1}
X_{t-1}
X_t
X_{t+1}
ϕ
FC layer with activation fuction
Copy
Concatenate

Notes

- Unlike the multilayer perceptron, here we save the hidden variable $\boldsymbol{H}_{t-1}$ from the previous timestep and introduce a new weight parameter, to describe how to use the hidden variable of the previous timestep in the current timestep.

- Specifically, the calculation of the hidden variable of the current timestep is determined by the input of the current timestep together with the hidden variable of the previous timestep.

Notes

- From the relationship between hidden variables $\boldsymbol{H}_t$ and $\boldsymbol{H}_{t-1}$ of adjacent timesteps, we know that those variables captured and retained the sequence's historical information up to the current timestep, just like the state or memory of the neural network's current timestep.

BackPropagation Through Time

- The backpropagation algorithm applied to unrolled (unfolded) RNN is called BackPropagation Through Time (BPTT).
- RNN considers its history. The current state depends on the input as well as the previous states.
- Thus to update the weights, the gradient of loss function at the particular time step t depends not only on the input but also on the **gradients of previous states at all the previous time steps**.
- The total error is given by the summation of the errors at all the time steps.

Problems

- Although the basic Recurrent Neural Network is effective, it can suffer from a significant problem. For deep networks, The Back-Propagation process can lead to the following issues:

- **Vanishing Gradients:** This occurs when the gradients become very small and tend towards zero.
- **Exploding Gradients:** This occurs when the gradients become too large due to back-propagation.

Solutions

- Long Short Term Memory (LSTM) Networks
- Gated Recurrent Unit (GRU) Networks

LSTM

- Consider the following example:

The color of sky is _____

- The RNN needs not remember what was said before this, or what was its meaning, all they need to know is that in most cases the sky is blue.

LSTM

- Now, consider the following example:

I spent 20 long years working for the underprivileged kids in Spain. I then moved to Africa.
I can speak fluent _____

- We can understand that since the author has worked in Spain for 20 years, it is very likely that he may possess a good command over Spanish. But, to make a proper prediction, the RNN needs to remember this context. The relevant information may be separated from the point where it is needed, by a huge load of ***irrelevant data***. This is where a Recurrent Neural Network fails!

LSTM to the Rescue!

- When we arrange our calendar for the day, we prioritize our appointments. If in case, we need to make some space for anything important we know which meeting could be cancelled to accommodate a possible meeting.

- Turns out that an RNN doesn't do so. In order to add a new information, it transforms the existing information completely by applying a function. Because of this, the entire information is modified, i. e. there is no consideration for *'important'* information and *'not so important'* information.

- With LSTMs, the information flows through a mechanism known as cell states. This way, LSTMs can selectively remember or forget things. The information at a particular cell state has three different dependencies.

LSTM

- Just because of this property of LSTMs, where they do not manipulate the entire information but rather modify them slightly, they are able to *forget* and *remember* things selectively.
- A typical LSTM network is comprised of different memory blocks called **cells.**
- There are two states that are being transferred to the next cell; the **cell state** and the **hidden state**.

LSTM

- The memory blocks are responsible for remembering things and manipulations to this memory is done through three major mechanisms, called **gates.**

- Input gate, Forget gate and output gate

LSTM Gates

- The input gate is responsible for the addition of information to the cell state.
- A forget gate is responsible for removing information from the cell state. The information that is no longer required for the LSTM to understand things or the information that is of less importance is removed via multiplication of a filter.
- The job of selecting useful information from the current cell state and showing it out as an output is done via the output gate.

GRU

- Similar concept as LSTM
- The GRU operates using a **reset gate** and an **update gate**. The reset gate sits between the previous activation and the next candidate activation to forget previous state, and the update gate decides how much of the candidate activation to use in updating the cell state.

LSTM and GRU Similarity

- Both LSTMs and GRUs are ability to keep memory/state from previous activations rather than replacing the entire activation like a vanilla RNN, allowing them to remember features for a long time and allowing backpropagation to happen through multiple bounded nonlinearities, which reduces the likelihood of the vanishing gradient.

LSTM and GRU Differences

- LSTMs control the exposure of memory content (cell state) while GRUs expose the entire cell state to other units in the network. The LSTM unit has separate input and forget gates, while the GRU performs both operations together using its reset gate.

- GRUs are faster to train as compared to LSTMs due to the fewer number of weights and parameters to update during training.

One Hot Encoding

- Generally, Encoding means representing each piece of data in a way that the computer can understand.
- There are different ways of encoding such as **Label Encoding**, or **One Hot Encoding**.

Label Encoding

- Let’s assume we’re working with categorical data, like cats and dogs.
- Looking at the name of label encoding, you might be able to guess that it encodes labels, where label is just a category (i.e. cat or dog).
- Encode just means giving them a number to represent that category (1 for cat and 2 for dog).
- By giving each category a number, the computer now knows how to represent them, since the computer knows how to work with numbers.

Problem of Label Encoding

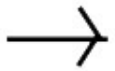
- The computer is programmed to treat higher numbers as higher numbers! It will naturally give the higher numbers higher weights.

Problem of Label Encoding

- Imagine if you had 3 categories of foods: apples, chicken, and broccoli.
- Using label encoding, you would assign each of these a number to categorize them: apple = 1, chicken = 2, and broccoli = 3.
- Now, if your model internally needs to calculate the average across categories, it might do 1+3 = 4/2 = 2. This means that according to your model, the average of apple and broccoli together is chicken!
- That’s a disaster.

Label Encoding

Food Name	Categorical #	Calories
Apple	1	95
Chicken	2	231
Broccoli	3	50



One Hot Encoding

Apple	Chicken	Broccoli	Calories
1	0	0	95
0	1	0	231
0	0	1	50

Solution

- Rather than labeling things as a number starting from 1 and then increasing for each category, we'll go for more of a binary style of categorizing.