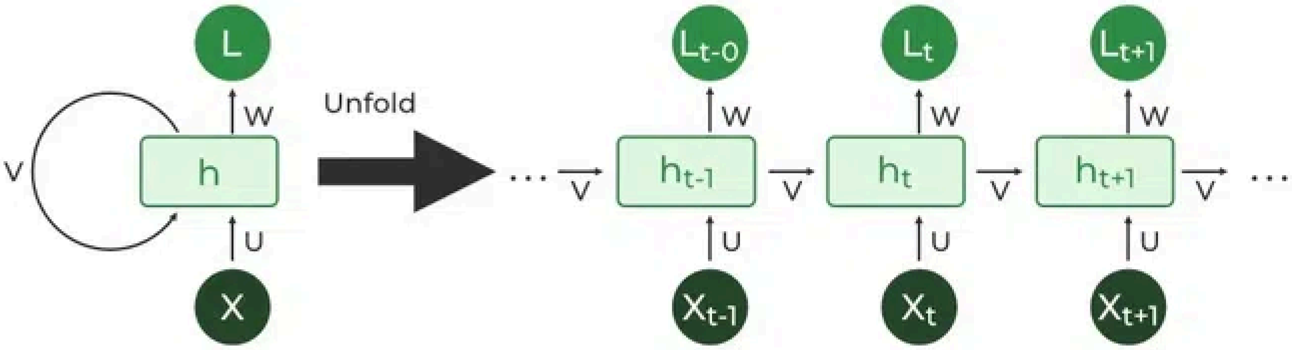
Recurrent Neural Network Notes by Soham

2024-07-25

1. Connection of Hidden Layers in RNNs

In RNNs, the hidden layers are connected not only to the next layer but also to themselves through time. This is different from traditional ANNs, where layers are only connected sequentially (feedforward).



- **Temporal Connection**: Each hidden layer at time step t receives input from both the current input x_t and the hidden state from the previous time step h_{t-1}.
- **Weight Sharing**: The weights used for these connections (both the input-to-hidden and hidden-to-hidden) are shared across all time steps.

2. Weights in Forward Propagation

During forward propagation in an RNN:

- **Input-to-Hidden Weights (W_{ih})**: These weights are used to connect the input x_t to the hidden state h_t.
- **Hidden-to-Hidden Weights (W_{hh})**: These weights (often referred to as W_{rec}) connect the hidden state from the previous time step h_{t-1} to the current hidden state h_t.
- These weights are the same across all time steps. This weight sharing is a fundamental aspect of RNNs, enabling them to generalize across different time steps.

3. Backpropagation Through Time (BPTT)

When updating weights in RNNs through Backpropagation Through Time (BPTT):

- **Shared Weights**: The weights W_{ih} and W_{hh} are shared across all time steps during both forward and backward propagation.
- **Gradients**: Gradients are calculated for each time step and accumulated to update the shared weights. Therefore, the weights W_{rec} (or W_{hh}) receive contributions from each time step's gradients during the backpropagation process.

Detailed Explanation with Equations

Forward Propagation

For each time step t:

$$h_t = \sigma(W_{ih} x_t + W_{hh} h_{t-1} + b_h)$$

where: - h_t is the hidden state at time step t. - x_t is the input at time step t. - W_{ih} are the weights from the input to the hidden state. - W_{hh} (or W_{rec}) are the recurrent weights connecting the previous hidden state to the current hidden state. - b_h is the bias term. - σ is the activation function (e.g., tanh, ReLU).

Backpropagation Through Time (BPTT)

During BPTT, we unroll the RNN through time and calculate the gradients for each time step: 1. Compute the loss L at the final time step. 2. Propagate the gradients backward through each time step. 3. Accumulate the gradients for the shared weights.

For each time step t:

$$\frac{\partial L}{\partial W_{ih}} = \sum_t \frac{\partial L_t}{\partial h_t} \frac{\partial h_t}{\partial W_{ih}}$$

$$\frac{\partial L}{\partial W_{hh}} = \sum_t \frac{\partial L_t}{\partial h_t} \frac{\partial h_t}{\partial W_{hh}}$$

These gradients are then used to update the weights:

$$W_{ih} \leftarrow W_{ih} - \eta \frac{\partial L}{\partial W_{ih}}$$

$$W_{hh} \leftarrow W_{hh} - \eta \frac{\partial L}{\partial W_{hh}}$$

where η is the learning rate.

Vanishing Gradient Problem

- The vanishing gradient problem occurs when gradients become very small as they are propagated backward through many time steps, causing the model to learn very slowly or not at all for early time steps.

- To mitigate this, techniques such as Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) are used.

-

Summary

- In RNNs, hidden layers are connected temporally to themselves across time steps, and the same weights are used for these connections at each time step during forward propagation.
- The recurrent weights W_{rec} (or W_{hh}) are the same across all time steps and are updated using gradients accumulated from all time steps during backpropagation.
- The updation of the weights both input and recurrent weights take place in the very end of the backpropogation.
- Be aware of the vanishing gradient problem and consider advanced RNN architectures like LSTM and GRU to address it.

This weight-sharing mechanism is crucial for the RNN's ability to learn temporal patterns and dependencies in sequential data.