Robeir Samir George 1901823

Assignment 3 (Part2)

2. Why are long-term dependencies difficult to learn in a RNN?

Solution:

In theory, RNN's are capable of handling such "long-term dependencies." A human could carefully pick parameters for them to solve toy problems of this form. Sadly, in practice, recurrent neural network doesn't seem to be able to learn them. This problem is called Vanishing gradient problem. The neural network updates the weight using the gradient descent algorithm. The gradients grow smaller when the network progress down to lower layers. The gradients will stay constant meaning there is no space for improvement. The model learns from a change in the gradient. This change affects the network's output. However, if the difference in the gradients is very small network will not learn anything and so no difference in the output. Therefore, a network facing a vanishing gradient problem cannot converge towards a good solution.

3. When would the use of Gated Recurrent Units (GRU) be more efficient than vanilla RNNs?

Solution:

Both LSTMs and GRUs can 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.

4. What are the advantage and disadvantage of Truncated Backpropagation Through Time (TBTT)?

Solution:

Truncated Backpropagation Through Time, or TBPTT, is a modified version of the BPTT training algorithm for recurrent neural networks where the sequence is processed one timestep at a time and periodically (k1 timesteps) the BPTT update is performed back for a fixed number of timesteps (k2 timesteps).

Truncated BPTT is a closely related method. It processes the sequence one timestep at a time, and every k1 timesteps, it runs BPTT for k2 timesteps, so a parameter update can be cheap if k2 is small. Consequently, its hidden states have been exposed to many timesteps and so may contain useful information about the far past, which would be opportunistically exploited.

The key feature of this algorithm is that the next backward pass is not performed until time step t + h0; in the intervening time the history of network input, network state, and target values are saved in the history buffer, but no processing is performed on this data.