Week 7 Reflections

Course	CS 598 - Deep Learning for Healthcare
Submitted By	Rohit Bhagwat (rohitb2@illinois.edu)

Questions

What are the main messages you learned from this chapter?

This week was focussed around learning the basics of RNN (Recurrent Neural Networks).

Recurrent Neural Networks (RNN) are a family of deep learning models for sequential data such as longitudinal patient records and time-series data. RNN is very natural to model variable-length sequences such as longitudinal EHR data, time-series like Electroencephalogram (EEG) and Electrocardiogram (ECG), and text sequences like clinical notes.

Some key concepts that were introduced during the video lectures and also explained in detail of the chapter include -

- RNN Fundamentals: Fundamental concept of RNN is that they have a sequential memory
- Backpropagation Through Time (BPTT)
- Long-ShortTerm Memory Networks (LSTM)
- Gated Recurrent Unit (GRU)
- Bidirectional RNN
- Sequence-to-Sequence RNN

Which part do you want to improve in this chapter?

In the lectures as well as in the chapter, we are introduced to the concepts very well. The author and the professor walked through the idea RNNs and the core concepts very well.

I did struggle with completing assignments, perhaps a walkthrough of an example of PyTorch would prove to be useful for students.

Although I think the exercises will help students get that understanding, it just needs more research.

What related resources (book, paper, blog, link) do you recommend your classmates to checkout?

Some great references, that also proved useful when working on the Homeworks -

Anyone Can Learn To Code an LSTM-RNN in Python (Part 1: RNN)
Summary: I learn best with toy code that I can play with. This tutorial teaches Recurrent Neural Networks via a very simple toy example, a short python implementation. Chinese Translation Korean Translation I'll tweet out (Part 2: LSTM) when it's complete at @iamtrask.
https://iamtrask.github.io/2015/11/15/anyone-can-code-lstm/

https://towardsdatascience.com/illustrated-guide-to-recurrent-neural-networks-79e5eb8049c9
https://towardsdatascience.com/illustrated-guide-to-recurrent-neural-networks-79e5eb8049c9

Understanding LSTM Networks
Posted on August 27, 2015 Humans don't start their thinking from scratch every second. As you read this essay, you understand each word based on your understanding of previous words. You don't throw everything away and start thinking from scratch again. Your thoughts have persistence.
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

https://www.youtube.com/channel/UCPjarHmDBblQ1k8ZVGvLGMA/videos
https://www.youtube.com/channel/UCPjarHmDBblQ1k8ZVGvLGMA/videos

What is the main difference between RNN and bidirectional RNN?

RNNs process input sequences in a specific order

Bidirectional RNNs on the other hand allows, depend on both previous and future events for producing the output.

They are simply two RNNs:

- One processes input sequences in a forward direction
- Another one processes in a backward direction

The output is computed based on the concatenation of the hidden states from both RNNs.

This makes bidirectional RNNs useful when looking at sequences, but less useful at looking at streaming data where the future data is not available yet.

What is the main difference between LSTM and GRU?

Here're some key pointers on both LSTM and GRU as explained in the lectures and the chapters -

Long Short-Term Memory (LSTM)

- The primary goal is to overcome the vanishing gradient problem
- Introduces a new structure called the *cell state*, which is designed to remember useful information and forget unnecessary information over time
- The key idea to control what information to remember or forget is through various gates

Gated Recurrent Unit (GRU)

- Compared with LSTM, the GRU uses a similar gating mechanism to learn long-term dependencies and alleviate the vanishing gradient problem
- GRU removes the cell state in LSTM and uses the hidden state h(t) serve as both the cell state and output state
- GRU has only two gates: *a reset gate r*, and *an update gate z*

A GRU is a simpler version of LSTM.

GRU uses a similar gating mechanism to learn long-term dependencies and alleviate the vanishing gradient problem, it does not have a cell state as LSTM. In a GRU, h(t) serves as both the cell state and the output state.