Week 6 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 we were introduced to a specific type of Neural Network called Convolutional Neural Network (CNN). It is one of the most popular networks for processing grid-like data, example image data or time series data.

When we were introduced to feed forward neural networks - the idea was that all the neurons in the earlier layer connected to all the neuron in the next layer. When we have high dimensional input, for example, image was a thousand by thousand pixels - the fully connected neural networks will have too many connections that has too many parameters, which are expensive to learn and very difficult to learn as well from data. CNN models try to reduce the number of parameters.

- Localization: To make to computationally more efficient, we can force the neuron to have a small number of connections. We also want to limit those connections to local input. In CNN model or convolution, we try to connect a set of x that are next to each other to a neural, there's no common neurons between far away inputs.
- Weight Sharing: This can be a considered as applying a filter or kernel that is really a set of weights to many position in the input data.
- Pooling to handle distortions: CNN architecture also have pooling layers outside convolution layers, which further dramatically reduce the number of parameters. The idea is if data, the two images are very similar, one has just shifted from the other by a little bit, then this pooling operation can handle that.

Advantages of CNN over a fully connected neural networks are the following.

- It have sparse interconnection
- Parameter sharing can reduce number of parameter and can handle translation invariance

The core concepts of Convolution and Pooling were explained using 1-d convolution which essentially applies to 2-d or 3-2 convolution.

Key concepts and terms -

- Filter (also called Kernel)

- Padding: The idea of padding is to add extra values (usually zeroes) outside the input data boundary to ensure convolution can be applied to elements near the boundary.
- Stride: Determines how often we apply the filter operations over the input.
- Spatial Extent

💡 Formula for calculating Output of Convolution

- $W_2 = \frac{W_1 - F + 2P}{S} + 1$
- $H_2 = \frac{H_1 - F + 2P}{S} + 1$
- $D_2 = K$

💡 Formula for calculating Output of Pooling

- $W_2 = \frac{W_1 - F}{S} + 1$
- $H_2 = \frac{H_1 - F}{S} + 1$
- $D_2 = D_1$

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 Convolutional Neural Networks - the core concepts of Convolution, Pooling etc..

If I have to recommend an improvement, perhaps there could be a walkthrough on how to calculate the feature map/kernel from beginning to end for a CNN. Also introducing an example using PyTorch could help give an understanding to the students on how to use the libraries that are already available.

Although I think the exercises will help students get that understanding.

Any additional topics do you suggest adding?

As I've mentioned above, some code examples of using PyTorch with a walk through of different layers will really help us as students.

I found the chapter and the case studies really interesting some of those case studies could've been included in the video lectures, the one on Covid as an example.

💡 I think the course CS 410 Text Information Systems is very useful for understanding more on this subject

If you are given a dataset of 100 X-ray images, would you still use CNN models? If so, which architecture would you try? And why?

In my opinion it may not be feasible for building a new CNN model, if there are pre-trained models available which work on similar data - we can consider using those models.

If image quality is high - and we still want to apply CNN models, we need to avoid overfitting. Perhaps Google's inception can be used to avoid that as it has a dropout mechanism. It is also known to have high levels of accuracy.

What about you are given 1000 images, which CNN models would you try?

In this scenario, I would consider using a CNN model of moderate complexity, such as ResNet-18.

1000 images is a higher dataset, but I still don't think it is large enough. I think it would be a better idea to create our own architecture instead of using the CNN models that were described in this chapter. We can play with the number of layers and perhaps baseline against AlexNet or VGG 19.

> ***References for above***
> Lecture Slides
> Chapter
> https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8