Deep Learning Notes 9.4 & 9.5

Key Words : Infinitely Strong Prior; Variants of the Basic Convolution Function

1. Infinitely Strong Prior

1. **Prior Probability Distribution** is a probability distribution over the parameters of model that encodes our beliefs about what models are reasonable, **before we have seen any data**.

2. A **Weak Prior** is a prior distribution with **high entropy**, such as a Gaussian distribution with **high variance**. Such a prior allows **the data to move the parameters more or less freely**.

3. A **Strong Prior** has very **low entropy**, such as a Gaussian distribution with **low variance**. Such a prior **plays a more active role in determining where the parameters end up**.

4. An infinitely strong prior places **zero probability on some parameters** and says that these parameters values are completely forbidden, **regardless of how much support the data gives to those values**.

5. We can think of the use of convolution as introducing an infinitely strong prior probability distribution over parameters of a layer. This prior says that the function the layer should learn contains **only local interactions** and **is equivariant to translation**. Likewise, the use of pooling is an infinitely strong prior that **each unit should be invariant to small translations**.

6. Two insights we should clear: 1) convolution and pooling can cause **underfitting**; 2) we should **only** compare convolution models to other convolution modes in benchmarks of statistical learning performance.

2. Variants of the Basic Convolution Function

7. Although at many spatial locations, the convolution operation with a single kernel can only extract one kind of feature. So we usually mean an operation that consists of many applications of convolution **in parallel** when we refer to convolution in the context of neural networks.

8. The input is usually not just a grid of real values. Instead, it is a grid of vector-valued observations.

9. In a multilayer convolutional network, the input to the second layer is the output of the first layer, which usually has the output of many different convolutions at each position.

10. For example, we usually think of the input and output of the convolution as being **3-D tensors**, with one index into the different channels and two indices into the spatials coordinates of each channel, and we can use **4-D tensors**, with the fourth axis indexing different examples in the batch.

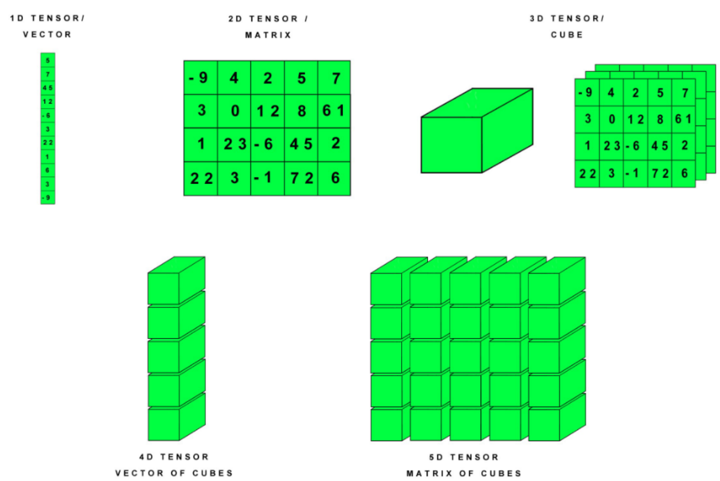

Figure 1: This picture is from hackernoon.com

11. These multi-channel operations are only commutative if each operation has **the same number of output channels as input channels**.

12. One essential feature of any convolutional network implementation is the ability to implicitly **zero-pad** the input **V** in order to make it wider, and **Zero-padding** the input allows us to control **the kernel width** and **the size of the output** independently. As we mentioned before, there are three types of convolution(i.e., valid, same, full).

13. Here are some useful pictures