EECS 182 Deep Neural Networks
Spring 2023 Anant Sahai

Note 5

1 Initialization

1.1 Insight

From the earlier discussions about ReLU, we know that given a ReLU network, the elbow (corner) can be found at $c = -\frac{b}{w}$. Let $w \sim N(0,1)$ and $b \sim N(0,1)$. The distribution of the elbow turns out to be Cauchy.

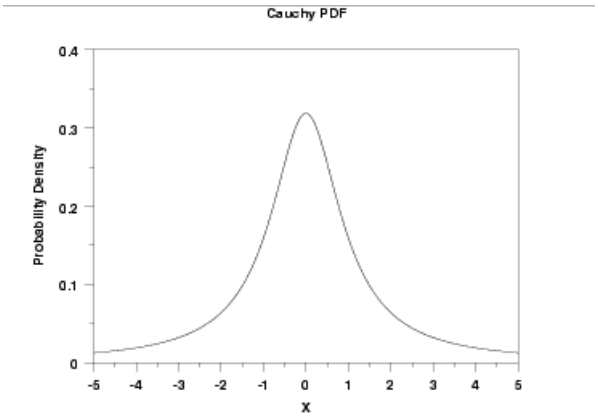

Figure 1: Cauchy Probability Density Function

The PDF of the corner c:

$$f(c) = \frac{1}{\pi(1+c^2)} \tag{1}$$

If we try to find the expectation of the corner, it turns out $E[C]$ is actually undefined due to it diverging:

$$E[C] = \int_{-\infty}^{\infty} \frac{1 \cdot c}{\pi(1+c^2)} dc \rightarrow \text{diverges} \tag{2}$$

Why do we care about initialization?

- It's desirable for the input to our ReLU layers to be centered near 0 and to be close to where the "action" is, otherwise poor initialization can lead to dead ReLU nodes from the beginning. Dead ReLUs are problematic because they make all the computations with the weights and biases, but are always less than 0, leading to wasted computations.
- Additionally, dead ReLUs are usually unable to recover, leading to a portion of our network being unable to learn further.

We want to achieve a distribution of $N(0,1)$ at initialization, the outputs of the layer will be standardized. So the main idea is that we want our inputs to have a distribution of $N(0,1)$ at initialization.

Current Basic Folk Wisdom:

- Use whatever worked on a related problem. (e.g., pre-trained network, literature review, etc)
- Random initialization using Gaussian $N(0, \sigma^2)$.
  * Xavier Initialization: $\sigma^2 = \frac{1}{d}$, where d is the fan-in of this unit.
  * $Var(\sum_{i=1}^{d} X_i) = dk$ (X_i are independent, has $mean = 0$ and $var = k$). Note that all variances add up to 1.

1.2 Weight Initialization

1.2.1 Xavier Initialization

Xavier Initialization aims to initialize weights such that the variance across every layer is the same. Let d be the fan-in (number of inputs of the layer) of the target layer.

Xavier Initialization

$$w_{1,i} \overset{\text{iid}}{\sim} N\left(0, \frac{1}{d}\right) \tag{3}$$

Problem: This doesn't work well for ReLU!

Why? If the previous layer was ReLU, we expect half of the outputs to be 0 on average. We really only have $\frac{d}{2}$ inputs, causing our the sum of layer's variances to be $\frac{1}{2}$ instead of 1.

1.2.2 He Initialization

In order to fix the variance issue from Xavier initialization, we simply multiply our variance by 2 such that the sum of the layer's variances equals 1. Another way to see why He Initialization's variance is $\frac{2}{d}$ instead of $\frac{1}{d}$: When initializing ReLU weights, we want our ReLU elbows to be close to where the action is. In other words, we want the ReLUs to actually produce non-linearities, so our model can emulate nonlinear phenomena. If this is the case, about half of our ReLUs should be in the off-state, so the actual fan-in for ReLUs is halved, which is why there's a two in the numerator of $\frac{2}{d}$.

He Initialization

$$w_{1,i} \overset{\text{iid}}{\sim} N\left(0, \frac{2}{d}\right) \tag{4}$$

We use He Initialization when the previous layer is ReLU, in order to achieve our target $N(0,1)$ initialization distribution.

1.3 Bias Initialization

There are typically four approaches that are used to initialize biases:

(a) **Treat fan-in as $d + 1$ and use Xavier Initialization**

This treats the bias as a weight in order to use a weight initialization technique, which is why we use $d + 1$ instead of d for the fan-in.

(b) **Just use bias $b = 0$**

This is essentially letting our optimizer initialize the bias in the first few steps.

(c) **Just use $b = 0.01$**

Empirically, people have seen that this sometimes performs better than $b = 0$.

(d) **Just use any small, random number**

This claims that it doesn't matter what small, random number we select, since our optimizer will quickly change it anyways.

